# PruningQC: Boosting the Quantum Computation Fidelity by Pruning Redundant Gates

Fang Qi
Tulane University
NEW ORLEANS, Louisiana, USA
fqi2@tulane.edu

Yongshan Ding
Yale University
New Haven, Connecticut, USA
yongshan.ding@yale.edu

Victor Bankston
Tulane University
New Orleans, Louisiana, USA
lbanksto@tulane.edu

Ji Liu
Argonne National Laboratory
Lemont, Illinois, USA
ji.liu@anl.gov

Lu Peng
Tulane University
NEW ORLEANS, Louisiana, USA
lpeng3@tulane.edu

## Abstract

Quantum computing offers promise for solving problems beyond classical capabilities, but its effectiveness is hindered by noise and high error rates in superconducting quantum computers. These issues, exacerbated by nearest-neighbor connectivity, necessitate extensive routing, leading to decoherence and multiple swap operations.

We introduce PruningQC, a novel method that optimizes circuit efficiency by removing redundant gates in measurement subsetting, reducing noise, and minimizing circuit depth. By selectively measuring specific qubits, PruningQC identifies and eliminates redundant gates without affecting outcomes, thereby enhancing quantum computing performance. Evaluations on three IBM 127-qubit machines show that PruningQC improves success rates by an average of 17.5 times over full measurement methods and 2.1 times over partial measurement techniques. This improvement is due to around 50% reduction in circuit depth and a significant decrease in gate counts. PruningQC can also be integrated with quantum error correction strategies, advancing quantum computing efficiency.

## CCS Concepts

• **Computer systems organization** → *Quantum Computing*.

## Keywords

Quantum Computing, Compilation, Fidelity, Redundant Gates

**ACM Reference Format:**
Fang Qi, Yongshan Ding, Victor Bankston, Ji Liu, and Lu Peng. 2025. PruningQC: Boosting the Quantum Computation Fidelity by Pruning Redundant Gates. In *Great Lakes Symposium on VLSI 2025 (GLSVLSI '25), June 30–July 02, 2025, New Orleans, LA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3716368.3735232


*GLSVLSI '25, New Orleans, LA, USA*

ACM ISBN 979-8-4007-1496-2/25/06
https://doi.org/10.1145/3716368.3735232

## 1 Introduction

Quantum computing holds promise for addressing challenges unattainable by classical methods [23], yet superconducting quantum computers face significant limitations due to low fidelity and noisy operational environments [6]. These limitations are exacerbated by the need for extensive routing operations, leading to deeper circuit designs that degrade performance [22, 25].

To address these challenges, researchers have explored strategies to enhance performance, such as developing robust quantum chips, introducing optimized routing paths [25], and dividing circuits into sub-circuits [24]. However, our research focuses on an underexplored area: pruning redundant gates to reduce circuit depth and improve computational efficiency. By selectively measuring only a subset of qubits, PruningQC identifies and eliminates redundant gates, significantly reducing cross-talk, decoherence errors, and the need for swap operations. This approach enhances fidelity without altering the logical outcomes of the circuits.

We validated PruningQC using well-known quantum benchmarks on three 127-qubit IBM quantum computers, showing a 17.5x improvement in success rate over the Baseline and a 50% reduction in circuit depth. The main contributions of our work are:

- A novel method for reducing quantum circuit depth, distinct from traditional strategies with high overhead.
- Identification of redundant gates in measurement subsetting, improving execution performance while maintaining logical equivalence.
- A comprehensive compilation platform for pruning redundant gates, achieving significant efficiency improvements with minimal overhead.

## 2 Background and Related Work

Quantum computing utilizes qubits, which can exist in superposition states, enabling quantum algorithms to be represented as sequences of unitary gates that evolve the system's state. However, current devices are prone to errors such as crosstalk, gate inaccuracies, and qubit relaxation ($T1$, $T2$) [18], which degrade the fidelity of computations as circuit depth increases. Consequently, optimizing the compilation process is crucial to minimize these errors and enhance overall system performance.

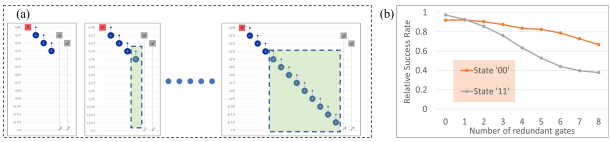


**Figure 1: (a) The Partial Measurement Circuits of the GHZ-13 algorithm featuring various redundant gates highlighted within the green box. The leftmost logic circuit depicts a pruned partial measurement circuit, while the rightmost logic circuit represents the original partial measurement circuit without pruning. (b) Relative success rate changes corresponding to redundant gates increase. Specifically, the probabilities of states '00' and '11' increase from 50% to 100%, and these adjustments are documented as relative success rates.**

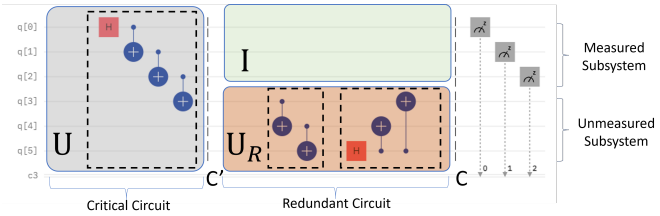


**Figure 2: An illustrative example of a logic circuit highlighting the equivalence achieved by pruning redundant circuits. The GHZ logic circuit is segmented into critical and redundant subcircuits with measurements. Dashed boxes indicate exemplary circuits.**

Recent work has focused on mitigating these errors through various software techniques, including measurement error reduction [4, 5], crosstalk mitigation [3], qubit allocation optimization [10, 11], and routing path optimization [20]. Measurement subsetting has also been proposed to address measurement error crosstalk [1], where local circuits with different measurement subsets are executed and updated the global circuit, resulting in boosted performance. Building on these efforts, PruningQC introduces a novel approach to reducing circuit depth by eliminating redundant gates in partial measurement circuits, thereby enhancing quantum computation fidelity. Recent methods like qubit tracing QuTracer [14] involve fine-grained division of circuits to achieve reduced circuit depth and boost performance. However, QuTracer introduces significant overhead when performing circuit cutting with partial measurements. For partial measurements involving more than two qubits, QuTracer separates the circuit into multiple partial measurement copies, each requiring circuit cutting. To execute circuit cutting for each partial measurement, QuTracer performs numerous individual circuit executions, each with different Pauli measurements. Each Pauli measurement circuit execution requires hundreds to thousands of additional trials. The number of individual Pauli measurement circuits grows exponentially with the number of measured qubits in the partial measurement circuits, leading to substantial computational overhead. This overhead is further compounded by the heavy computational demands of pre- and post-execution processes, making it challenging to implement QuTracer efficiently on current quantum hardware. In contrast, PruningQC maintains a lower overhead than the original partial measurement method, resulting in significantly reduced computation time due to lower circuit depth.

## 3 Problem and Motivation

During the partial measurement experiments, we identified redundant gates in the original work [1]. As explained in Sec. 4.2, these redundant gates can be removed without changing the logical outcomes of the circuit. This approach is based on the well-known causal cone method, derived from the concept of the light cone in physics [8]. The causal cone analysis helps determine which gates and associated errors contribute to the measurements, a technique extensively used in quantum computing simulations [16].

We conducted experiments using a partial measurement circuit from the GHZ-13 algorithm, progressively removing redundant gates, as shown in Fig. 1. Contrary to expectations, we found that the computing performance improved as more redundant gates were removed. Further investigation revealed that unwanted hardware errors were the primary cause of the performance issues in the original setup.

Removing redundant gates enhances performance by mitigating several types of errors. First, redundant gates increase circuit depth, leading to more relaxation errors ($T1$ and $T2$ errors). Second, these gates can introduce crosstalk, creating physical errors that affect the critical path and compromise measurement accuracy. Third, fewer redundant gates reduce the need for swap operations, which are used to move qubits to suitable physical locations during execution. Swap operations can introduce errors, including the movement of critical qubits along the swap path, additional crosstalk, and relaxation errors due to increased waiting time. Notably, these swap errors are often invisible at the logical level.

To address these challenges, it is crucial to develop a partial measurement-aware compiler that actively reduces redundant gates, thereby avoiding unwanted hardware errors and enhancing computational performance.

## 4 PruningQC Design

### 4.1 PruningQC Objectives

In quantum computing, increased qubit usage and circuit length are directly associated with higher noise levels and error rates. Partial measurement circuits often contain redundant gates, which can degrade performance by unnecessarily increasing circuit depth and introducing additional sources of error. Removing these redundant gates can lead to substantial improvements in computational accuracy. For instance, in our primary experiment, eliminating redundant CNOT gates in a GHZ-13 benchmark circuit resulted in a 47% increase in the success rate.

PruningQC is a framework designed for partial measurement-aware compilation. It actively identifies and eliminates redundant gates, optimizing the circuit for execution. Furthermore, it enables the development of tailored compilation strategies for the pruned

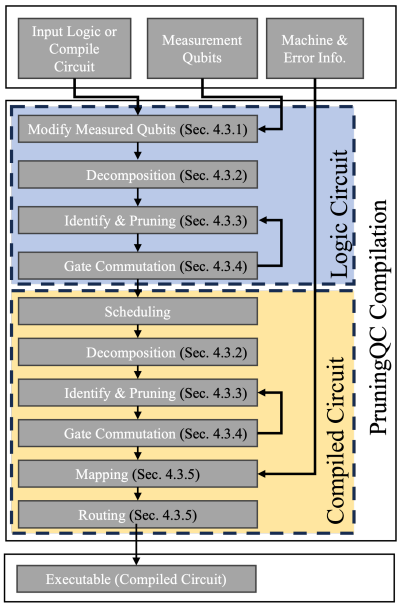


**Figure 3: PruningQC compilation workflow**

circuits, significantly boosting computational performance and reliability.

## 4.2 Pruning Equivalent Proof

We demonstrate that removing redundant gates from a circuit does not alter the measurement outcomes. As shown in Fig. 2, if a circuit $C$ contains redundant gates $R$, the modified circuit $C'$ (with $R$ removed) produces identical measurement statistics.

This equivalence is maintained since the redundant gates act only on unmeasured qubits. Fig. 2 illustrates this with a GHZ logic circuit, where removing redundant circuits yields identical outputs. These results are consistent with other proof methods, such as the Lieb-Robinson bound and non-signaling quantum theorem [9, 26].

## 4.3 PruningQC Compilation

PruningQC offers a comprehensive compilation strategy designed to enhance the computational performance of any given algorithm. The compilation workflow, illustrated in Fig. 3, accepts a logic or compiled circuit, a designated set of measurement qubits, and the noise profile of the machine as inputs. Subsequent steps involve applying pruning-related compilation optimizations at both the logic circuit and the compiled circuit levels, aiming to identify and eliminate as many redundant gates as possible. The ensuing subsections will provide detailed descriptions of the key components utilized in the PruningQC approach, delineating each step's role and contribution to optimizing the overall circuit efficiency.

*4.3.1 Modify Measured Qubits.* During this phase, adjustments are made to the input circuit to choose the optimal measurement gate sets, extracting more redundant gates. This module plays a pivotal role in crafting localized sub-circuits, each with a distinct set of partial measurement gates. These sub-circuits facilitate the post-execution Bayesian update algorithm, enhancing fidelity [1], and help aggregate results from smaller qubit clusters to reconstruct the original larger outcomes [24].

**Algorithm 1** Identify & Pruning Redundant Gates

**Require:** Circuit $C$ with a mix of measured and unmeasured qubits; qubit $q$
**Ensure:** Optimized circuit $C$ with redundant gates removed
Initialize qubit status for each qubit in $C$ to either measured ($M$) or unmeasured ($Unm$)
**for** Traverse through each gate in $C$ in reverse order **do**
  **if** Gate is a single-qubit operation and acts on an unmeasured qubit **then**
    Designate the gate as redundant
  **else if** Gate is a two-qubit operation **then**
    Determine the qubits involved in the operation
    **if** Both qubits are unmeasured **then**
      Designate the gate as redundant
    **else**
      Update the status of any involved unmeasured qubits to critical, indicating their role in the computation
    **end if**
  **end if**
**end for**
Remove gates marked as redundant from $C$
**return** $C$

*4.3.2 Decomposition.* This model operates at both the logical circuit and compiled circuit levels. Essentially, this stage processes the input circuit and generates an equivalent circuit composed solely of 1-qubit and 2-qubit operations.

At the logical circuit level, the decomposition model transforms the circuit to exclusively use 1 and 2-qubit gates that are originally present in the circuit. This model eliminates the presence of gates involving three or more qubits, thereby enabling the compiler to commence searching for redundant gates at the logical circuit level. It also facilitates the subsequent gate commutation model in identifying equivalent gates within the same 2-qubit gate library.

At the compiled circuit level, the decomposition model breaks down the gates into those supported by the target machine and applies optimizations aimed at identifying opportunities to cancel or combine gates. This approach not only simplifies the circuit but also enhances its efficiency by reducing unnecessary operations.

*4.3.3 Identify & Pruning Redundant Gates.* Central to the compilation process, this module is tasked with pinpointing and eliminating redundant gates, yielding a circuit that's both functionally equivalent and more concise. As illustrated in Fig. 2, certain properties characterize these redundant gates, aiding their identification. Specifically, the gate must be unitary and should only interact with unmeasured qubit(s). Moreover, the gate cannot propagate its state through a two-qubit path leading to the measurement gate. The procedure, elaborated in Algorithm 1, sequentially examines each gate, assessing their dependencies with respect to the measure gates based on the aforementioned criteria. To explain briefly, upon ingestion of a circuit with only select qubits being measured, the algorithm initializes a dictionary variable, $Q_{\text{status}}$, encapsulating the

measurement status of every qubit. A qubit slated for measurement is marked $M$; otherwise, it is tagged $Unm$, which could change to $Critical$ once a path is established between itself and a measured qubit. Then, the algorithm examines all the gates in the circuit in reverse order, analyzing them for redundancy. As depicted in Fig. 2, the algorithm systematically evaluates each gate, earmarking those that operate exclusively on unmeasured qubits devoid of any connection to measurement gates as redundant. Ultimately, these extra gates are removed from the circuit. It is important to note that the complexity of Algorithm 1 is $O(G)$, where $G$ represents the count of gates, as in the worst case, all gates might be deemed redundant and each gate is examined once and labeled as such.

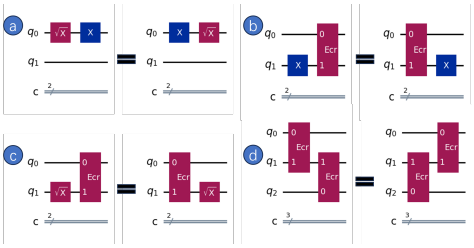

**Figure 4: Commutation gate pairs of native support gates on IBM_Kyoto 127-qubit machine.**

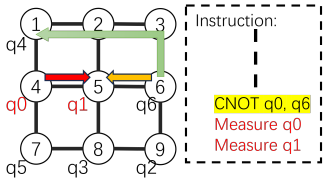


**Figure 5: Partial Measurement-aware Routing. The left shows the mapping of virtual qubits with physical qubits before executing the redundant CNOT gate between $q0$ and $q6$, highlighted on the right-hand instruction list. The three colored arrows indicate three types of possible methods for relocating $q0$ and $q6$ to adjacent positions.**

*4.3.4 Gate Commutation.* The gate commutation model is designed to identify equivalent gate patterns that permit the reordering of gate sequences, thereby presenting additional opportunities to eliminate redundant gates and further reduce circuit depth. As depicted in Fig. 4, we illustrate commutation gate pairs for the basis gates natively supported by the IBM_Kyoto 127-qubit machines. These equivalent pairs are utilized to check for gate commutation at the compiled circuit level. Each of the four subfigures presents a pair of equivalent gate pairs that can interchange with each other. Using Fig. 4.b as an example, when implementing the pruning technique, assume the first qubit is being measured and the second qubit is not. In this scenario, the configuration on the right-hand side results in only one ECR gate and a circuit depth of 1. Conversely, the circuit on the left exhibits a circuit depth of 2 and a higher likelihood of introducing noise, potentially impacting output performance.

Thus, this model actively searches through the last few gates for all qubits to identify matching gate commutation patterns for modification. The rearranged circuit is then resubmitted to the previous model for further identification of redundant gates and undergoes this process iteratively $n$ times. This systematic approach ensures the optimization of the quantum circuit by minimizing noise and improving overall computational efficiency.

*4.3.5 Partial Measurement-aware Mapping & Routing.* Our PruningQC also implements a tailored mapping and routing strategy for the pruned partial measurement compiled circuit. PruningQC reverses the compiled circuit and makes several mapping attempts based on the readout errors on the physical chip. These attempts select different combinations of low readout errors for the partial measurement gates. The options with relatively low swap counts and circuit depth are then chosen.

The routing strategy is illustrated in Figure 5, showing how different swap paths affect the outcome of pruned partial measurement circuits. The red arrow indicates six CNOT errors that would be engraved into the measurement results and should be avoided. The yellow and green paths are preferable for different topologies. In the example shown, the green path is optimal since the swap errors do not affect the measurement, and the path is relatively short in the mesh topology. However, for hexagonal or ring topologies, the yellow path would be a better choice.

## 5 Experiment Setup

Experiments were conducted on three 127-qubit IBM superconducting quantum machines (IBM_Kyoto, IBM_Brisbane, IBM_Osaka), all equipped with Eagle r3 processors supporting ECR, ID, RZ, SX, and X gates.

We used Qiskit [7] for implementing PruningQC, with Noise-Aware Sabre [13] as the baseline compiler (optimization level 2). JigSaw [1] was used for partial measurement comparisons, applying the same optimization as the Baseline for consistency.

We evaluated PruningQC using benchmarks like Bernstein-Vazirani (BV), Greenberger-Horne-Zeilinger (GHZ), Deutsch-Jozsa (DJ), Hidden Shift (HS), Quantum approximate optimization algorithm (QAOA), and Variational Quantum Eigensolver (VQE) from previous studies [12, 13, 17, 21]. These benchmarks, characterized by low success rates and high circuit depths, were chosen to demonstrate PruningQC's effectiveness in improving computational results.

Experiments were conducted with 20K to 68K trials, evenly split between global execution and partial measurements. PruningQC was applied during partial measurement trials to demonstrate performance improvements at the same or reduced computational cost.

Quantum computation performance was evaluated using the following metrics: **Probability of Successful Trial (PST)** [15, 17, 21]: The ratio of correct trials to total trials, with relative PST indicating improvement over the Baseline. **Hellinger Fidelity** [2, 8]: Measures the similarity between the output distribution and a noise-free distribution, serving as a benchmark for evaluating distribution outputs. **Inference Strength (IST)** [15, 19]: The ratio of correct output probability to the most frequent incorrect output, with higher IST indicating better performance. **Circuit Depth Reduction**: Comparison of circuit depths before and after pruning

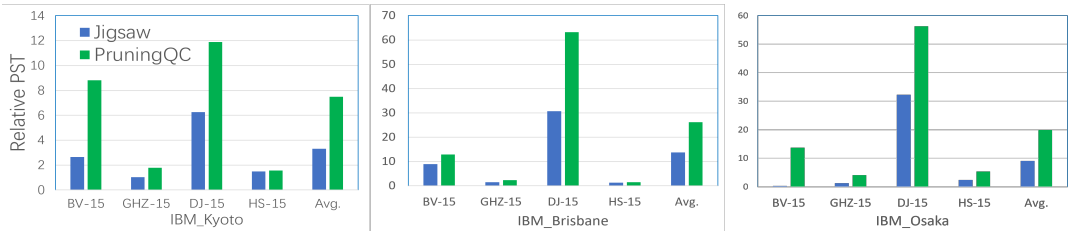


**Figure 6: PruningQC vs. JigSaw: Relative PST improvement across various benchmarks on IBM 127-qubit machines.**

to assess error reduction. **Gate Reduction**: Analysis of gate count reduction to highlight efficiency improvements achieved through pruning.

## 6 Results and Sensitivity Studies

### 6.1 Probability of Successful Trials and Inference Strength

We evaluated PruningQC against Baseline and Jigsaw strategies on IBM 127-qubit machines, showing consistent PST improvements. PruningQC achieved an average relative PST of 7.85x over the Baseline, peaking at 56x, while Jigsaw averaged 3.5x with a maximum of 30x, as shown in Fig. 6. PruningQC consistently outperformed Jigsaw across all benchmarks, notably improving PST in cases where baseline PST was below 10%. When comparing the fidelity, PruningQC achieves 10% higher fidelity compared to Baseline and 6.7% with respect to Jigsaw. We also compare IST values for PruningQC and Jigsaw. PruningQC enhanced IST by 8.76x to 25.86x, compared to Jigsaw's 3.21x to 8.8x improvement. PruningQC consistently outperformed Jigsaw in both PST and IST metrics.

### 6.2 Circuit Depth Reduction

Fig. 7 illustrates substantial circuit depth reductions with PruningQC, achieving approximately 50% reduction compared to Baseline and Jigsaw. This reduction minimizes the likelihood of relaxation errors, leading to enhanced PST and IST.

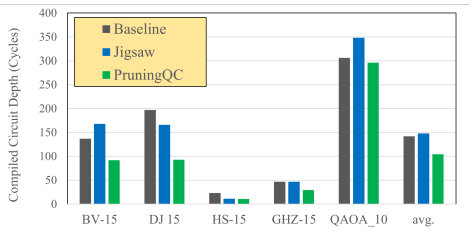


**Figure 7: Compiled circuit depth for Baseline, Jigsaw, and PruningQC on IBM_Kyoto machine.**

### 6.3 Gate Reduction

Fig. 8 shows gate count reductions with PruningQC, including significant decreases in 1-qubit, 2-qubit, and communication gates. PruningQC averaged 82.9 gates for 1-qubit operations, compared to 172.9 (Jigsaw) and 233 (baseline). For 2-qubit gates, PruningQC averaged 17.24 gates, significantly lower than Jigsaw and the Baseline. The reduction in gate count contributes to the improved PST, especially for the BV and DJ benchmarks. We examined the effect of varying partial measurement qubits using the BV-19 algorithm. PruningQC consistently outperformed Jigsaw across all configurations, demonstrating its robustness in enhancing PST.

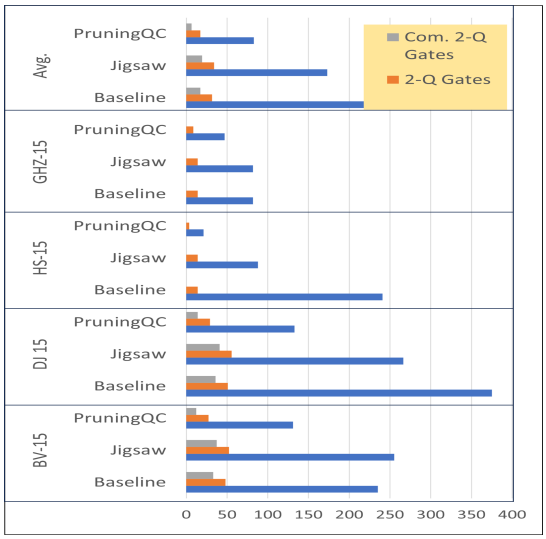


**Figure 8: Compiled Circuit Gate Counts for Baseline, Jigsaw, and PruningQC on IBM_Kyoto machine.**

## 7 Scalability Study

### 7.1 Complexity Study

The PruningQC framework involves three key stages: generating compiled circuits with pruned redundant gates, executing these circuits on a quantum machine, and applying a post-execution reconstruction algorithm on a classical computer. The compilation stage incurs a minor overhead with linear complexity $O(G)$, where $G$ is the number of gates, with fewer than 10 repetitions. During execution, PruningQC requires similar or less quantum computation time compared to Jigsaw and Baseline, due to its focus on Partial Measurement Circuits (PMCs) with reduced circuit depth. The post-execution Bayesian Reconstruction algorithm, shared with Jigsaw, maintains linear complexity in time and memory, ensuring that PruningQC is comparable or superior in computational overhead to other approaches.

## 7.2 Scalability on Circuit Depth and Gates

Fig. 9 illustrates PruningQC's performance in reducing circuit depth and gate counts compared to Baseline and Jigsaw, using the DJ benchmark on the IBM_Kyoto machine. PruningQC consistently achieves a 50% reduction in circuit depth and significantly lowers gate counts, with a 63.8% reduction in 1-qubit gates and a 50% reduction in 2-qubit gates relative to Baseline and Jigsaw. These results suggest that PruningQC scales effectively for larger computations and is applicable to a wider range of algorithms.

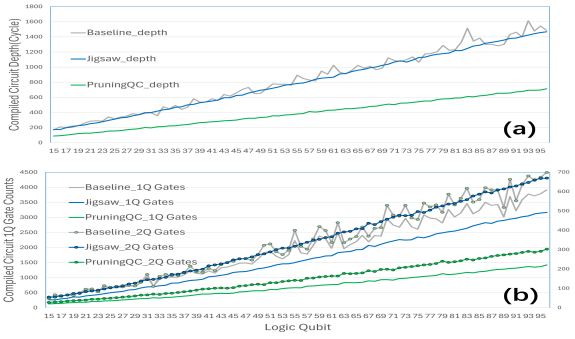


**Figure 9: Scalability analysis of PruningQC vs. Baseline and Jigsaw using the DJ algorithm on IBM_Kyoto machine. (a) Circuit Depth Comparison. (b) Gate Count Analysis: 1-qubit gates (left y-axis) and 2-qubit gates (right y-axis).**

## 8 Conclusion

PruningQC, a novel quantum compilation framework, enhances computational fidelity and efficiency by eliminating redundant gates and implementing partial measurement-aware compilation. Evaluations on three 127-qubit IBM machines using established benchmarks show significant improvements in computation success rates, along with reduced circuit depth and gate counts. These enhancements address key challenges in quantum computing, such as noise and error propagation, and extend the capabilities of quantum hardware by minimizing circuit complexity and mitigating decoherence and operational errors.

## Acknowledgments

This work was supported in part by the National Science Foundation (NSF) under Grant OIA-2019511. Dr. Ji Liu acknowledges support from the DOE-SC Office of Advanced Scientific Computing Research MACH-Q project under contract DE-AC02-06CH11357.

## References

[1] Poulami Das, Swamit Tannu, and Moinuddin Qureshi. 2021. Jigsaw: Boosting fidelity of nisq programs via measurement subsetting. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*. 937–949.

[2] Poulami Das, Swamit S Tannu, Prashant J Nair, and Moinuddin Qureshi. 2019. A case for multi-programming quantum computers. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*. 291–303.

[3] Yongshan Ding, Pranav Gokhale, Sophia Fuhui Lin, Richard Rines, Thomas Propson, and Frederic T Chong. 2020. Systematic Crosstalk Mitigation for Superconducting Qubits via Frequency-Aware Compilation. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 201–214.

[4] Lena Funcke, Tobias Hartung, Karl Jansen, Stefan Kühn, Paolo Stornati, and Xiaoyang Wang. 2022. Measurement error mitigation in quantum computers through classical bit-flip correction. *Physical Review A* 105, 6 (2022), 062404.

[5] Pranav Gokhale, Olivia Angiuli, Yongshan Ding, Kaiwen Gui, Teague Tomesh, Martin Suchara, Margaret Martonosi, and Frederic T Chong. 2020. Optimization of Simultaneous Measurement for Variational Quantum Eigensolver Applications. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 379–390.

[6] Akel Hashim, Ravi K Naik, Alexis Morvan, Jean-Loup Ville, Bradley Mitchell, John Mark Kreikebaum, Marc Davis, Ethan Smith, Costin Iancu, Kevin P O'Brien, et al. 2020. Randomized compiling for scalable quantum computing on a noisy superconducting quantum processor. *arXiv preprint arXiv:2010.00215* (2020).

[7] IBM. [n. d.]. Open-Source Quantum Development. https://qiskit.org/. Retrieved on 04-16-2021.

[8] Zhi-Xiang Jin and Shao-Ming Fei. 2018. Quantifying quantum coherence and nonclassical correlation based on Hellinger distance. *Physical Review A* 97, 6 (2018), 062342.

[9] Tomotaka Kuwahara, Tan Van Vu, and Keiji Saito. 2024. Effective light cone and digital quantum simulation of interacting bosons. *Nature Communications* 15, 1 (2024), 2520.

[10] Travis LeCompte, Fang Qi, Xu Yuan, Nian-Feng Tzeng, M Hassan Najafi, and Lu Peng. 2023. Graph neural network assisted quantum compilation for qubit allocation. In *Proceedings of the Great Lakes Symposium on VLSI 2023*. 415–419.

[11] Travis LeCompte, Fang Qi, Xu Yuan, Nian-Feng Tzeng, M Hassan Najafi, and Lu Peng. 2023. Machine-learning-based qubit allocation for error reduction in quantum circuits. *IEEE Transactions on Quantum Engineering* 4 (2023), 1–14.

[12] Ang Li, Samuel Stein, Sriram Krishnamoorthy, and James Ang. 2020. Qasmbench: A low-level qasm benchmark suite for nisq evaluation and simulation. *arXiv preprint arXiv:2005.13018* (2020).

[13] Gushu Li, Yufei Ding, and Yuan Xie. 2019. Tackling the qubit mapping problem for NISQ-era quantum devices. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. 1001–1014.

[14] Peiyi Li, Ji Liu, Alvin Gonzales, Zain Hamid Saleem, Huiyang Zhou, and Paul Hovland. 2024. QuTracer: Mitigating Quantum Gate and Measurement Errors by Tracing Subsets of Qubits. *arXiv preprint arXiv:2404.19712* (2024).

[15] Ji Liu and Huiyang Zhou. 2020. Reliability modeling of nisq-era quantum computers. In *2020 IEEE international symposium on workload characterization (IISWC)*. IEEE, 94–105.

[16] Angus Lowe, Max Hunter Gordon, Piotr Czarnik, Andrew Arrasmith, Patrick J Coles, and Lukasz Cincio. 2021. Unified approach to data-driven quantum error mitigation. *Physical Review Research* 3, 3 (2021), 033098.

[17] Prakash Murali, Jonathan M Baker, Ali Javadi-Abhari, Frederic T Chong, and Margaret Martonosi. 2019. Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. 1015–1029.

[18] Michael A Nielsen and Isaac Chuang. 2002. *Quantum computation and quantum information*. American Association of Physics Teachers.

[19] Tirthak Patel and Devesh Tiwari. 2020. Veritas: accurately estimating the correct output on noisy intermediate-scale quantum computers. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1–16.

[20] Fang Qi, Xin Fu, Xu Yuan, Nian-Feng Tzeng, and Lu Peng. 2024. A New Routing Strategy to Improve Success Rates of Quantum Computers. In *Proceedings of the Great Lakes Symposium on VLSI 2024*. 546–550.

[21] Fang Qi, Kaitlin N Smith, Travis LeCompte, Nian-feng Tzeng, Xu Yuan, Frederic T Chong, and Lu Peng. 2023. Quantum Vulnerability Analysis to Guide Robust Quantum Computing System Design. *IEEE Transactions on Quantum Engineering* (2023).

[22] Salonik Resch, Swamit Tannu, Ulya R Karpuzcu, and Moinuddin Qureshi. 2020. A day in the life of a quantum error. *IEEE Computer Architecture Letters* 20, 1 (2020), 13–16.

[23] Peter W Shor. 1999. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review* 41, 2 (1999), 303–332.

[24] Wei Tang, Teague Tomesh, Martin Suchara, Jeffrey Larson, and Margaret Martonosi. 2021. Cutqc: using small quantum computers for large quantum circuit evaluations. In *Proceedings of the 26th ACM International conference on architectural support for programming languages and operating systems*. 473–486.

[25] Swamit S Tannu and Moinuddin K Qureshi. 2019. Not all qubits are created equal: a case for variability-aware policies for NISQ-era quantum computers. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. 987–999.

[26] Minh C Tran, Andrew Y Guo, Christopher L Baldwin, Adam Ehrenberg, Alexey V Gorshkov, and Andrew Lucas. 2021. Lieb-Robinson light cone for power-law interactions. *Physical review letters* 127, 16 (2021), 160401.