INDUSTRIAL AND SYSTEMS ENGINEERING

ISE

Early Exploration of a Flexible Framework for Efficient Quantum Linear Solvers in Power Systems

MUQING ZHENG[1,2], YOUSU CHEN[2], XIU YANG[1], AND ANG LI[2]

[1]Pacific Northwest National Laboratory, Richland, WA, 99352, USA

[2]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA

ISE Technical Report 24T-010

LEHIGH
UNIVERSITY

Early Exploration of a Flexible Framework for Efficient Quantum Linear Solvers in Power Systems

Muqing Zheng*†, Yousu Chen*, Xiu Yang†, Ang Li*
* Pacific Northwest National Laboratory, Richland, WA, 99352, USA
†Lehigh University, Bethlehem, PA 18015, USA

***Abstract*—The rapid integration of renewable energy resources presents formidable challenges in managing power grids. While advanced computing and machine learning techniques offer some solutions for accelerating grid modeling and simulation, there remain complex problems that classical computers cannot effectively address. Quantum computing, a promising technology, has the potential to fundamentally transform how we manage power systems, especially in scenarios with a higher proportion of renewable energy sources. One critical aspect is solving large-scale linear systems of equations, crucial for power system applications like power flow analysis, for which the Harrow-Hassidim-Lloyd (HHL) algorithm is a well-known quantum solution. However, HHL quantum circuits often exhibit excessive depth, making them impractical for current Noisy-Intermediate-Scale-Quantum (NISQ) devices. In this paper, we introduce a versatile framework, powered by NWQSim, that bridges the gap between power system applications and quantum linear solvers available in Qiskit. This framework empowers researchers to efficiently explore power system applications using quantum linear solvers. Through innovative gate fusion strategies, reduced circuit depth, and GPU acceleration, our simulator significantly enhances resource efficiency. Power flow case studies have demonstrated up to a eight-fold speedup compared to Qiskit Aer, all while maintaining comparable levels of accuracy.**

***Index Terms*—quantum computing, Harrow-Hassidim-Lloyd, high-performance computing, grid analytics**

I. Introduction

As nations around the world unite in the battle against climate change, clean energy and decarbonization have taken center stage. For perspective, in 2021, one-fifth of the electricity in the U.S. came from renewable sources. With eyes set on a greener future, America is aiming for 100% clean electricity by 2035 and zero net emissions by 2050. California has a pathway to achieve carbon neutral by 2045 [1]. Europe is on a similar trajectory, with numerous countries launching strategies to cut their greenhouse gas emissions significantly.

The energy landscape is indeed transforming. The shift is evident with the growing reliance on renewable sources like wind and solar, advanced energy-consuming devices, and other cutting-edge technologies. Take California as an example: there's been a rapid increase in the use of wind and solar energy. By 2021, 90% of its power came from these sources. And in a milestone moment on May 8, 2022, this number surged past 100%. This kind of changes has brought significant challenges to grid management as it demands a much faster, quicker responses to a much complex, dynamic, and with uncertainty that has not been seen before.

High-performance-computing, as an emerging technique, can help power system engineers to gain faster computational speeds. However, there are still problems in power systems, particularly those of large scale and high granularity, are intractable to classical cluster machines due to their computational complexity, data communication overhead, or both. Some examples are listed below:

- Stochastic Optimization for Operational Planning with Renewables: Given the variability and uncertainty of renewable energy sources like wind and solar, planning for their optimal integration and dispatch involves solving high-dimensional stochastic optimization problems.
- Very Large-Scale Contingency Analysis: Assessing the grid's resilience to numerous possible contingencies (like multiple line outages) requires analyzing a combinatorial number of scenarios. It's a problem that can quickly grow beyond the capabilities of classical computing clusters.
- Security Constrained Unit Commitment: Determining which power plants should be turned on or off to meet the anticipated electricity demand, while also ensuring the grid remains stable and secure against potential disturbances.

Classical cluster machines, while powerful, have limits in terms of processing capability and data communication bandwidth. Emerging computing paradigms, like quantum computing, might offer pathways to tackle some of these intractable problems in the future.

The basic information unit in quantum computing is called qubits. An n-qubit quantum state can be described as a 2^n-size complex vector in Hilbert space. Quantum state has a unique feature: at most 2^n number of distinguishable, i.e., orthogonal, quantum states can form a valid n-qubit quantum state. This property is called superposition. Another property of quantum states is entanglement. It is when several quantum states are intricately intertwined so that any transformation on one state will affect all the other states. Both are important sources of quantum advantage.

The Pacific Northwest National Laboratory (PNNL) is operated by Battelle for the U.S. Department of Energy (DOE) under Contract DE-AC05-76RL01830. This work was supported by DOE Office of Electricity through its Advanced Grid Modeling (AGM) program, DOE Office of Basic Energy Science (BES), National Science Foundation CAREER, and Defense Advanced Research Projects Agency. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a DOE Office of Science User Facility located at Lawrence Berkeley National Laboratory. Emails: yousu.chen|ang.li@pnnl.gov, and muz219|xiy518@lehigh.edu.

Quantum computing holds great promise for power systems, even though some details remain unclear. Many experts are excited to start exploring its application before it is perfected, as shown in [2] [3], [4]. However, there are still gaps that need to be bridged to enable power system engineers, who typically do not have a quantum computing background, to evaluate how to use quantum algorithms to accelerate their applications. To address this, we introduce a flexible framework, empowered by Northwest Quantum Simulator (NWQSim) [5], [6], to bridge the gaps between power system engineers and quantum techniques.

Building upon this potential of quantum computing for power systems, it's crucial to understand the tools that can help harness this technology effectively. Quantum solvers, for instance, leverage the principles of quantum mechanics to address specific types of problems more efficiently than classical computers. Some of the main quantum solvers include:

- Harrow-Hassidim-Lloyd (HHL) Algorithm: the HHL algorithm is used for solving linear systems of equations efficiently on a quantum computer [7],
- Variational Quantum Eigensolver (VQE): VQE is a quantum solver used for finding the ground state energy of quantum systems,
- Quantum Approximate Optimization Algorithm (QAOA): QAOA is a quantum solver designed for solving combinatorial optimization problems, such as MaxCut problems,
- Quantum Annealing (QA): QA currently solves the problems that can be reformulated to Ising models and quadratic unconstrained binary optimization (QUBO) problems.

This paper start with integrating the HHL solver into the framework due to its ability to handle linear system and its connection to power systems. HHL has the potential to solve complex linear equations with exponential speedup compared to classical solvers, which could enable a significant computational improvement for many power system applications, such as power flow, short-circuit analysis, contingency analysis, and voltage stability analysis.

Nevertheless, applying the HHL algorithm directly to power system problems presents several challenges. First and foremost, the quantum circuits for the HHL algorithm tend to be very deep, even for small-scale problems. To illustrate, consider the HHL circuit that only solves a random 2-by-2 linear system, depicted in Figure 1. It already has 120 one-qubit gates and 90 two-qubit gates [8]. Furthermore, the small experiment illustrated in Figure 2 shows that significant errors persist even in a circuit with one-qubit gate and a two-qubit gate. In essence, given the current noise level in NISQ (Noisy Intermediate-Scale Quantum) devices, obtaining meaningful results for such circuits is difficult. To the best of our knowledge, a 4-by-4 system is the largest one that can be accurately solved on an actual quantum computer in existing literature [9], which falls short of the requirements for practical power system applications.

Additionally, HHL circuits have certain requirements on the qubit connectivity. This is not an issue for trapped-ion quantum computers as in [9], but it is challenging for superconducting quantum computers offered by companies like IBM and Google. Lastly, there is a lack of software tools for developing and deploying quantum algorithms for power system applications, especially for non-quantum-background researchers and engineers.

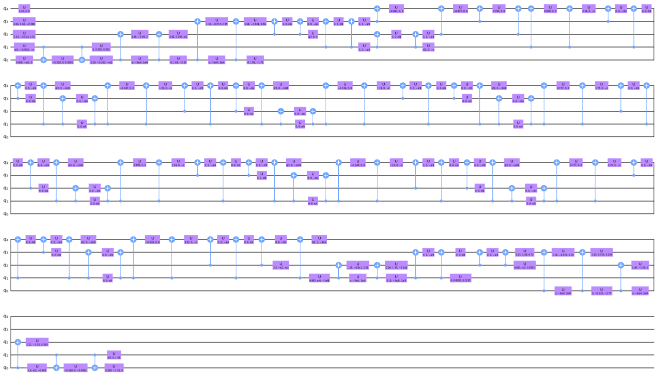

Fig. 1. An example of the *transpiled* HHL circuit that solves a random 2-by-2 linear system. It is generated by the package in [8] and transpiled into one- and two-qubit basis gates.

Given all these challenges, the authors propose a flexible simulator-based framework designed for tackling power system problems. This framework leverages quantum simulators, which are dedicated to the evaluation of quantum circuits on classical quantum computers. The use of simulators can facilitate feasibility studies in this area, targeting more realistic problem settings and making it more accessible to a broader range of researchers. Once fault-tolerant quantum computers are available, researchers can then migrate their existing work to the real hardware, further advancing the field.

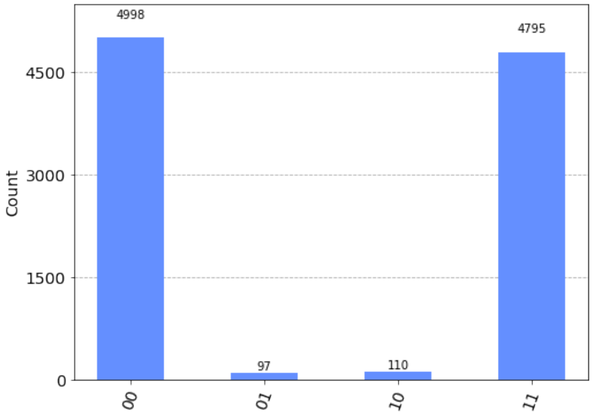

Fig. 2. 10,000 measurements of bell state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ prepared from initial state $|00\rangle$ using a Hadamard gate and a Controlled-NOT gate on IBM' superconducting quantum computer ibmq_kolkata. The noise in the quantum computer causes the actual results deviate from the ideal case where 01 and 10 should not appear and the counts for 00 and 11 are almost equal in the measurements.

The remainder of this paper is organized as follows: Section II introduces the flexible framework along with its key components and setup. Section III explains the details of applying the HHL algorithm to linear systems in power flow problems. Section IV illustrates the use of our framework on three power flow problems. Finally, Section V concludes the paper with a discussion of future work.

II. Introduction to the Framework

The motivation of this work is to provide power system engineers with a versatile platform for testing power system applications by seamlessly combining classical and quantum algorithms within a hybrid simulation framework. This hybrid simulation framework allows users to connect with various existing power system applications with different quantum simulators via the interfaces that facilitate the smooth exchange of data necessary for the successful execution of hybrid simulations. The ultimate goal is to equip power system professionals with the tools and environment they need to explore and advance quantum computing in power system applications.

A. The proposed hybrid framework

As illustrated in Figure 3, our conceptual framework serves as a bridge, enhancing the communication between a range of quantum algorithms, such as HHL and variational quantum algorithm (VQA), and external simulation tools compatible with diverse languages and platforms. Our current focus lies in the integration of the HHL algorithm with the power flow functions in MatPower [10]. This integration necessitates some data preprocessing to transform the data into a format suitable for HHL. Subsequently, we are poised to extend our framework by integrating other quantum algorithms, such as VQA, to further enrich our hybrid simulation capabilities.

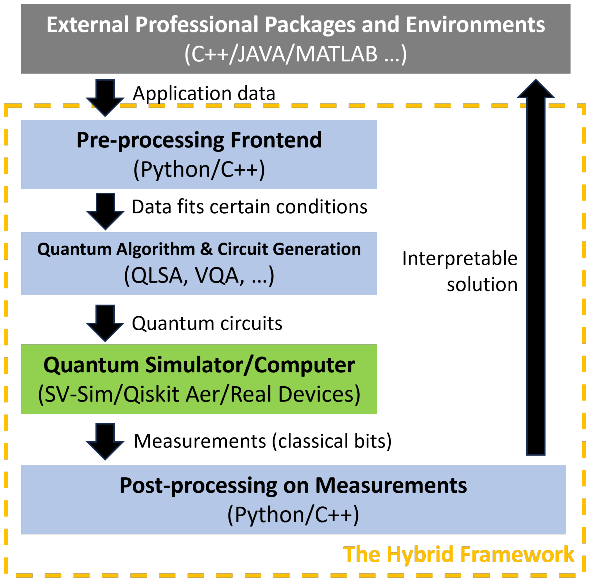
External Professional Packages and Environments
(C++/JAVA/MATLAB ...)
Application data
Pre-processing Frontend
(Python/C++)
Data fits certain conditions
Quantum Algorithm & Circuit Generation
(QLSA, VQA, ...)
Interpretable solution
Quantum circuits
Quantum Simulator/Computer
(SV-Sim/Qiskit Aer/Real Devices)
Measurements (classical bits)
Post-processing on Measurements
(Python/C++)
The Hybrid Framework

Fig. 3. The conceptual view of the proposed framework.

B. Existing software stack choices

On the classical side of the framework, as a starting point, we have integrated MatPower due to its widespread use in power system problems in academia. For the quantum computing component of our framework, we have selected Qiskit [11] developed by IBM, from among other tools such as Cirq from Google [12], and Q# from Microsoft [13]. Qiskit offers several advantages that make it a preferred choice. Foremost is the large and active community supporting Qiskit as an open-source python library. This helps lower the potential difficulties on the development of the framework. Additionally, Qiskit provides a wide range of implementations for popular quantum algorithms, which will assist the future integration of additional quantum algorithms into the framework. Notice that Qiskit has provided a complete VQA routine, but its recent update removed its HHL module. To address this, we have adapted a Qiskit-based HHL package in [8] and incorporated it into the framework.

Although Qiskit has provided quantum simulators in its Aer submodule, their efficiency falls short when dealing with complex circuits like those in the HHL algorithm. To overcome this limitation, our preferred simulator is the SV-Sim simulator in NWQSim. It is a C++ CPU/GPU statevector simulator includes various front-end supports such as C++, Qiskit, native Python, and Q#. When compared to the statevector simulators in Qiskit, Cirq, and Q#, SV-Sim exhibits exceptional performance in the benchmarking tests in [6]. The speed comparisons between simulators in Qiskit Aer and NWQSim on power flow use cases are also shown in Section IV.

C. Embedded techniques in SV-Sim simulator

The majority of time consumption during HHL circuit simulation is attributed to the process of simulation and measurement. Therefore, it's crucial to highlight several techniques employed by SV-Sim (NWQSim) to optimize computation time for deep circuits. The most significant time sink is the unitary evolution of the state. For SV-Sim, a noticeable improvement over the simulators in Aer (from Qiskit) and qsim (from Cirq) is the specialized gate implementation. While other simulators use general unitary gates for all one-qubit and two-qubit gates, SV-Sim offers specialized computation for dozens of supported basis gates and specific architectures of CPUs and GPUs. One instance of such specialized computation is gate fusion. As illustrated in Figure 4, gate fusion merges several applicable gates into a single gate based on four distinct strategies. This substantially reduces both the gate counts and the depth of the circuits. Taking the 210-gate circuit shown in Figure 1 as an example, the actual circuit executed in SV-Sim only has 67 gates. While gate fusion is limited to certain basis gates, it does not necessitate the matrix multiplications. Thus, the overall runtime can be lowered significantly [6].

Another source of time inefficiency stems from latency arising from communication between multiple computational units during parallel computing. On one hand, SV-Sim employs the so-called "PGAS-based SHMEM" communication model,

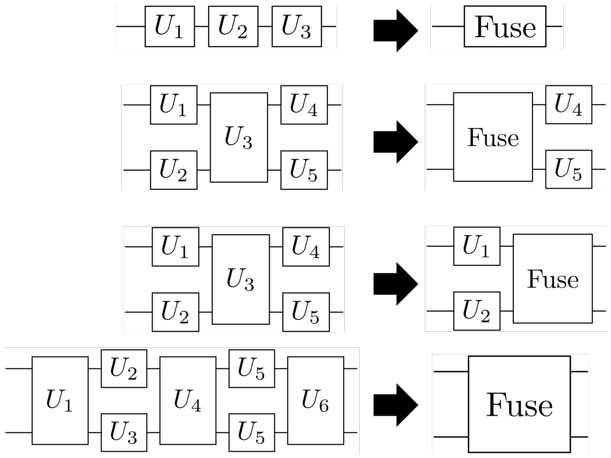

Fig. 4. Four possible gate fusion strategies for *applicable gates*. The individual gates on the left-hand side are combined into one fused gate on the right-hand side. The priorities among four strategies is from top to the bottom: first, one-qubit gates are fused; then two different fusion between one-qubit and two-qubit gates happen; and finally, two-qubit gates are fused together.

which provides enhanced support for software governing direct links between GPUs compared to the traditional MPI routine. On the other hand, researchers have also developed a novel method to minimize unnecessary data migration [6].

III. Power Flow Use Case

Power flow is a fundamental power system application. In this paper, our primary focus centers on the Newton-Raphson based AC power flow, a widely used technique for power system analysis.

A. Procedures for applying HHL algorithm on practical problems

Let's consider the linear system $\mathbf{Ax} = \mathbf{b}$ in an AC power flow problem. The condition number of the matrix $\mathbf{A}$ is very crucial in HHL. Firstly, the running time of HHL has a quadratic dependency on the condition number, which is notably worse than the state-of-the-art classical method like factorization methods or conjugate gradient. Likewise, a large condition number requires higher number of qubits for quantum phase estimation (QPE). Because QPE governs the accuracy of the final output and the sizes of state and gates in QPE are determined by 2^n where n is the number of qubits, any incremental in qubit will be very challenging for quantum simulators. To mitigate this difficulty, a preconditioner is used for system with relatively high condition number in our routine. The current implementation utilizes the Gauss-Seidel (GS) preconditioner due to its efficiency. But it does not preserve the Hermiticity of the original matrix. In Section IV, a case is provided to show the use of GS preconditioner can improve the accuracy of the final solution.

The complete procedures used to solve the linear system from the power flow problems are following.

Pratical HHL procedures

Goal:

Providing $\mathbf{A}$ and $\mathbf{b}$ in a linear system $\mathbf{Ax} = \mathbf{b}$, use HHL to solve for solution vector $\mathbf{x}$.

Process:

1) First, normalize $\mathbf{b}$ to get $|\mathbf{b}\rangle$ and record $\|\mathbf{b}\|$, then check the properties of matrix $\mathbf{A}$:
   a) if the condition number is too high, use a preconditioner
   b) if the dimension is not a power of 2, expand it to the nearest power of 2
   c) if the matrix is not Hermitian (e.g., due to the preconditioner), make it Hermitian:
   $$\begin{bmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^\dagger & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ |\mathbf{x}\rangle \end{bmatrix} = \begin{bmatrix} |\mathbf{b}\rangle \\ \mathbf{0} \end{bmatrix}.$$
2) Generate the HHL circuit as in Figure 5.
3) Compute $|\mathbf{x}\rangle$ by measuring the final state $|\vec{\psi}\rangle$ from the circuit.
   - If a simulator is used, the statevector of $|\psi\rangle$ can be read directly.
   - if a real quantum computer is used, the statevector of $|\psi\rangle$ has to be reconstructed from state tomography.
4) Retrieve the solution vector $\mathbf{x}$ from $|\mathbf{x}\rangle$.
   a) Compute
   $$\|\mathbf{x}\| = |\lambda_0| \cdot P(\text{measure ancilla and get } 1),$$
   where $|\lambda_0| \leq |\lambda_i|$ for all $i \neq 0$, and λ_i are eigenvalues of the matrix input in the circuit.
   b) Finally, $\mathbf{x} = \|\mathbf{x}\| \cdot \|\mathbf{b}\| \cdot |\mathbf{x}\rangle$.
5) Output $\mathbf{x}$.

B. Resources estimation

Following the settings in [8] and after computing the first step in Section III-A (expand and Hermitize the matrix $\mathbf{A}$), the estimated total number of qubits used, n_{total}, is based on the following equation.

$$n_{\text{total}} = n_{\text{data}} + n_{\text{QPE}} + n_{\text{neg val}}$$

where

$$\begin{cases} n_{\text{data}} & = \log_2(\text{number of rows or columns of matrix}) \\ n_{\text{QPE}} & = \max(n_{\text{data}} + 1, \lceil \log_2(\kappa + 1) \rceil) \\ n_{\text{neg val}} & = \begin{cases} 1 & \text{if the input matrix has negative eigenvalues} \\ 0 & \text{else} \end{cases} \end{cases}$$

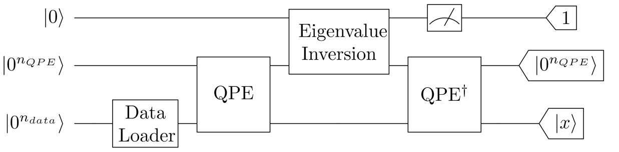

Fig. 5. The conceptual circuit of HHL

and κ is the condition number of matrix $\mathbf{A}$.

IV. Numerical experiments

To demonstrate the capabilities of our framework, we selected the IEEE 14-bus and IEEE 30-bus systems [14], which are readily available through the Matpower package. These systems serve as our initial testbed, allowing us to showcase the framework's functionality.

Experiments were conducted on Perlmutter supercomputer using a single NVIDIA A100 GPU. We will usemultiple GPUs to accommodate the increased computational demands for future large test cases. In our study, HHL circuits for both the 14-bus and 30-bus cases were executed on both Qiskit Aer and the SV-Sim simulator. The detailed results are presented in Table I. A highlight is the impact of the GS preconditioner on the 30-bus case. It can reduce the condition number of the 30-bus system from 492.5 to 109.3. This reduction had the effect of lowering the value of n_{qpe} from 10 to 8, as elaborated in the entry marked "30-buses*" in the table..

Although GS preconditioner evidently improved the accuracy of the final solution, it came at the cost of increased circuit generation time and thus simulation time. That was because the preconditioner broke the Hermiticity of the original matrix, resulting in deeper circuits. Additionally, SV-Sim consistently outperformed Qiskit Aer, achieving simulation speeds up to 8 times faster. Notably, the running time of the C++ segment within SV-Sim was 1.7s, 36.2s, and 34.5s for the three cases, respectively. This observation indicated that the actual backend computation time for each circuit ranged from $1/150$ to $1/65$ of the time recorded in Python. This discrepancy can be attributed to the substantial time overhead involved in casting data in between C++ and Qiskit. Unfortunately, since it is difficult to obtain the actual computation time from Qiskit Aer, the authors could not make a direct comparison on simulation time without type-casting.

V. Conclusion and future work

This paper presents a flexible hybrid framework, offering expedited problem-solving through the incorporation of quantum algorithms while integrating with existing power system tools. The framework is based on an advanced quantum simulator that significantly minimize resource-intensive communications and GPU-based parsing through the implementation of specialized computations. Use case with power flow studies of IEEE test systems showed up to 8 times speedup compared to Qiskit Aer with similar levels of accuracy. This framework has potential to empower power system engineers, even those without a quantum computing background, to explore and advance quantum computing and its applications in power system domain.

TABLE I
Experiment results on 14-bus case, 30-bus case, and 30-bus case with GS preconditioner (30-bus*)

	14-bus	30-bus	30-bus*
Matrix Size	13 × 13	29 × 29	29 × 29
Condition Number	119.2	492.5	109.3
n_{total}	13	16	15
- n_{data}	4	5	6
- n_{QPE}	8	10	8
Circuit Generation Time	88s	30.5 min	37.5 min
Qiskit Aer Simu. Time	27s	31.3 min	37.5 min
- Error $\lVert \mathbf{x}_{hhl} - \mathbf{x}_{true} \rVert_2$	$1.97 \cdot 10^{-3}$	$1.18 \cdot 10^{-3}$	$6.35 \cdot 10^{-4}$
SV-Sim Simu. Time	13s	3.9min	4.7 min
- Error $\lVert \mathbf{x}_{hhl} - \mathbf{x}_{true} \rVert_2$	$1.97 \cdot 10^{-3}$	$1.18 \cdot 10^{-3}$	$6.35 \cdot 10^{-4}$

This paper shows an initial exploration with a focus on HHL linear algorithm in power system applications. As we look ahead, our vision includes several key improvements to further enhance the framework's capabilities. First, a C++ implementation of QPE and HHL circuit generation based on NWQSim would greatly reduce the time overhead for circuit generation and type-casting data between Qiskit and C++. With this new implementation in place, it will be feasible for running large-scale linear problems with higher number of qubits and more GPUs for scalability testing. Furthermore, authors will include more solvers into the framework, such as various variational quantum solvers, and extend it to a wide ranges of power system applications.

References

[1] Edison International. Countdown to 2045: Realizing California's Pathway to Net Zero September 2023. IEEE Electrification Magazine. https://download.newsroom.edison.com/create_memory_file/?f_id=6508e6633d63325f2e763f1b&content_verified=True.

[2] Zhou Y, Tang Z, Nikmehr N, et al. Quantum computing in power systems. iEnergy, 2022, 1(2): 170-187. https://doi.org/10.23919/IEN.2022.0021.

[3] Golestan, Saeed, M. R. Habibi, SY Mousazadeh Mousavi, Josep M. Guerrero, and Juan C. Vasquez. "Quantum computation in power systems: An overview of recent advances" Energy Reports 9 (2023): 584-596.

[4] Chen, Y., Stein, S., Li, A., and Huang, Z. H. (2023, July). Is It Coming Soon to Power Systems: Quantum Computing and Its Early Exploration. In 2023 IEEE Power & Energy Society General Meeting (PESGM) (pp. 1-5). IEEE.

[5] Li, A., Subasi, O., Yang, X. & Krishnamoorthy, S. Density Matrix Quantum Circuit Simulation via the BSP Machine on Modern GPU Clusters. *Proceedings Of The International Conference For High Performance Computing, Networking, Storage And Analysis.* (2020), https://doi.org/10.1109/SC41405.2020.00017. Available at https://github.com/pnnl/NWQ-Sim

[6] Li, A. et al. SV-Sim: Scalable PGAS-Based State Vector Simulation of Quantum Circuits. *Proceedings Of The International Conference For High Performance Computing, Networking, Storage And Analysis.* (2021), https://doi.org/10.1145/3458817.3476169. Available at https://github.com/pnnl/NWQ-Sim

[7] Harrow, A., Hassidim, A. & Lloyd, S. Quantum Algorithm for Linear Systems of Equations. *Phys. Rev. Lett.*. **103**, 150502 (2009,10), https://link.aps.org/doi/10.1103/PhysRevLett.103.150502

[8] Vazquez, A. quantum_linear_solvers. *GitHub Repository.* (2022), https://github.com/anedumla/quantum_linear_solvers

[9] Yalovetzky, R., Minssen, P., Herman, D. & Pistoia, M. Hybrid HHL with Dynamic Quantum Circuits on Real Hardware. *ArXiv Preprint ArXiv:2110.15958.* (2023).

[10] Zimmerman, R., Murillo-Sánchez, C. & Thomas, R. MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education. *IEEE Transactions On Power Systems.* **26**, 12-19 (2011)

[11] Qiskit contributors Qiskit: An Open-source Framework for Quantum Computing. (2023)

[12] Google Cirq. Available at: https://github.com/quantumlib/Cirq

[13] Microsoft Q#. Available at: https://github.com/microsoft/qsharp-language

[14] Power Systems Test Case Archive. Available at: https://labs.ece.uw.edu/pstca/