Deep Quantum Circuit Simulations of Low-energy Nuclear States

Ang Li[1,2], Alessandro Baroni[1], Ionel Stetcu[3] and Travis S. Humble[1*]

[1]Quantum Science Center, Oak Ridge National Laboratory, One Bethel Valley Road, Oak Ridge, 37831, Tennessee, USA.
[2]Physical and Computational Sciences Directorate, Pacific Northwest National Laboratory, 902 Battelle Blvd, Richland, 99354, Washington, USA.
[3]Theoretical Division, Los Alamos National Laboratory, P.O. Box 1663, MS B283, Los Alamos, 87545, New Mexico, USA.

*Corresponding author(s). E-mail(s): humblets@ornl.gov;
Contributing authors: ang.li@pnnl.gov; baronia@ornl.gov;
stetcu@lanl.gov;

Abstract

Numerical simulation is an important method for verifying the quantum circuits used to simulate low-energy nuclear states. However, real-world applications of quantum computing for nuclear theory often generate deep quantum circuits that place demanding memory and processing requirements on conventional simulation methods. Here, we present advances in high-performance numerical simulations of deep quantum circuits to efficiently verify the accuracy of low-energy nuclear physics applications. Our approach employs several novel methods for accelerating the numerical simulation including 1- and 2-qubit gate fusion techniques as well as management of simulated mid-circuit measurements to verify state preparation circuits. We test these methods across a variety of high-performance computing systems and our results show that circuits up to 21 qubits and more than 115,000,000 gates can be efficiently simulated.

Keywords: quantum computing, numerical simulation

1 Introduction

Quantum computing offers many opportunities to explore the complex interactions of many-body nuclear physics [1, 2]. This includes enabling efficient methods for modeling quantum physical processes as well as new techniques to simulate their outcomes [3]. Recent discoveries in preparing quantum states to model nuclear physics have shown the efficacy of these methods for studying both structure and dynamics [4–7]. In the low-energy regime, this includes calculating binding energies [8], quadrupole moments [9], and electromagnetic transition rates [10]. With early validation at small scales, quantum simulation methods for low-energy nuclear physics are expected to solve larger systems that are currently inaccessible to conventional approaches and thus expand computing capabilities for scientific discovery [11]

A powerful approach to these forms of digital quantum simulation rely on quantum circuits as sequences of gates to prepare and transform a quantum state under a defined Hamiltonian [12]. Accurate implementations of these quantum circuits are used to estimate observable outcomes that describe the prepared state [13]. The structure of the quantum circuits for quantum simulation are often derived from first-principles models of the underlying many-body problem that are then translated into exact or approximate operations [14]. For example, unitary coupled cluster (UCC) theory has been used recently to solve nuclear shell models by constructing quantum circuits that first encode fermionic fields in a spin representation and then apply unitary transformations to approximate the ground state [6, 15]. By transforming these physics-driven quantum circuits into the spin representation, the subsequent unitary operator decomposition can be executed using a quantum computer [16, 17]. Quantum circuits, therefore, represent the functionality of software programs that run on quantum computing hardware to perform these types of nuclear physics simulations.

An important step in designing the quantum simulation circuits is verification, which checks the correctness of the quantum circuit construction and the outputs observed during execution. Numerical simulation is a direct approach to verify the circuit transformations that can be used to check the prepared quantum state as well as the calculation of statistics and diagnostics [18]. However, numerical simulation of quantum circuits is challenged by the exponential memory requirements with respect to system size for representing quantum states. Additionally, the probabilistic nature of quantum circuit execution often generates a combinatorial expansion of possible circuit outcomes that must be tracked. Formally, computational complexity arguments suggest that conventional numerical methods are intractable for simulating arbitrary quantum circuits, and it is widely anticipated that the applicability of numerical simulation is upper bounded by circuit width and depth, which is a concern for verification of quantum circuits. The continued development of numerical simulation techniques to accelerate and optimize such calculations is an area of active research within the quantum computing community.

The challenges for numerical simulation extend immediately to applications in nuclear physics, where deep quantum circuits spanning many qubits are often required to create highly accurate quantum states of the many-body problem. The case of deep quantum circuits is challenging for numerical simulation because these circuits are constructed from long sequences of unitary transformation, i.e., gates, to ensure accurate preparation of the intended quantum state [6, 15]. Recent advances in using mid-circuit measurements as part of more efficient state preparation methods has helped to reduce circuit depth [16], but this approach requires numerical simulation methods to manage the probabilistic outcomes generated by such measurements.

Here, we show how numerical simulations of deep quantum circuits can be used to efficiently verify the accuracy of low-energy nuclear physics applications. Our results demonstrate the performance scaling of quantum circuit simulations using novel methods to accommodate mid-circuit measurement integrated into deep circuit constructions. A principal approach is to leverage the right integration of GPU-accelerated numerical simulation methods. The leading contributions of this work are the development of mid-circuit measurement techniques to amplify the probability of observable outcomes; the improved efficiency of simulating mid-circuit measurement by avoiding repeated circuit sampling, and the improved efficiency of deep circuit numerical simulations using gate fusion.

The remainder is organized as follows. Section 2 introduces background on the theoretical model for the second quantized Hamiltonian, the quantum algorithms for simulating the corresponding states, and methods and notation for numerical simulation of these quantum circuits. Section 3 specifies the algorithm for state preparation of low-energy nuclear states. Details about the numerical simulation methods are provided in Section 4 with results presented in Section 5. Concluding remarks are offered in Section 6.

2 Background

2.1 Theoretical Model and Qubit Mapping

The nucleons inside the atomic nucleus interact via two-, three-, and so on body forces, with a clear hierarchy, in which two-body interactions dominate, followed by the many-body forces whose contributions diminish with increasing the number of bodies involved. A natural way to take into account the above constraint is given by chiral effective field theory (χEFT). In χEFT the symmetries of the underlying theory of strong interactions of quarks and gluons QCD are used to write down Lagrangians that describe interactions between nucleons and nucleons and pions and each interaction term is multiplied by a low energy constant that is fixed using some of the available experimental data. Interactions developed in this framework successfully predict nuclear binding energies up to $A = 48$ [19–21].

In second quantization, the Hamiltonian governing the static and dynamics properties of the nuclear system can be written as

$$H = \sum_{i,j=1,N_s} t_{ij} a_i^\dagger a_j + \frac{1}{2} \sum_{ij,kl} V_{ij,kl} a_i^\dagger a_j^\dagger a_l a_k + \cdots , \tag{1}$$

where N_s is the number of single particle states included in the Hilbert space, t_{ij} and $V_{ij; kl}$ are the one- and two-body matrix elements of the nucleon-nucleon interaction, while $a_i^\dagger$ (a_i) are the creation (annihilation) operators for state i. Given that the mean field is a good first approximation, we have allowed a general one-body term in Eq. (1).

Using realistic full-space inter-nucleon interactions, like the ones rigorously constructed through χEFT, would not be feasible in the near future due to hardware limitations. Hence, a less demanding testing ground for quantum algorithms has been identified in simpler models like the Lipkin-Meshkov-Glick model [7] or the phenomenological shell model [6, 15]. In particular the shell model offers the advantage that it has some of the complexity of realistic nucleon-nucleon interactions in a small model space, even though it was found to produce very entangled states [6]. The simulations presented in this paper were performed using the Cohen-Kurath interaction [22] in the p shell, where active the space is restricted to six proton and neutron states with $j = 1/2$ and $j = 3/2$, assuming a ^{4}He inert core. We would like to emphasize that while an excellent testing ground for quantum algorithms and even hardware benchmarks, shell model calculations in larger model spaces is not our final goal but rather an intermediary step for the time being.

The first step in the process of implementing various algorithms on quantum hardware and/or numerical simulators is to map this interaction to the qubit space, and different mappings could have advantages over the others, although the advantage is not always clear nor easy to quantify. The Jordan-Wigner (JW) [23] or occupation mapping is the most intuitive second quantization scheme. In this case, the creation and annihilation operators are given by

$$a_i^\dagger = \frac{1}{2} \left(\prod_{j=0}^{i-1} Z_j \right) (X_i - iY_i), \tag{2}$$

$$a_i = \frac{1}{2} \left(\prod_{j=0}^{i-1} Z_j \right) (X_i + iY_i), \tag{3}$$

where the X_i, Y_i and Z_i are the Pauli matrices acting on qubit i. Thus, each single-particle state is associated with a single qubit, and in order to describe a system with N_s, the number of qubits required is N_s. If the qubit is measured in state $|0\rangle$, the state is unoccupied, while a measurement in state $|1\rangle$ represents an occupied state.

The Jordan-Wigner mapping is very simple and intuitive, but it has a couple of disadvantages. First, it describes a Fock space of dimension 2^{N_s} that contains particle numbers from zero to N_s. For particle-conserving Hamiltonians as the ones governing the nuclear many-body system, only one subspace

with a fixed particle number is active at one time. Moreover, for the nuclear problem, the JW mapping is even less efficient given that the Hamiltonian concurrently preserves the proton and neutron particle numbers. Let us take an example considering N_p protons and N_n neutrons, each in N_s states. In this case, for large N_s, and using the Stirling approximation [24], the number of many-body states with N_p protons and N_p is

$$
\tilde{N} \approx \left(\frac{N_s^2}{(N_s - N_p)(N_s - N_n)} \right)^{N_s}
$$
$$
\times (N_s - N_p)^{N_p}(N_s - N_n)^{N_n} e^{-N_p - N_n}, \tag{4}
$$

which is much smaller than the dimension of the Fock space 2^{2N_s} represented in the Jordan Wigner mapping. Second, in order to represent state i one must include $i - 1$ operators acting on the previous qubits, and hence, the number of one-qubit gates and entanglement gates in general is quite large. This poses challenges for applications on current and near-term devices.

While less transparent, other more efficient mappings within the second quantization framework exist. One alternative is the Bravyi-Kitaev mapping [25], where the qubits store partial sums of occupation numbers, requiring the number of states to be powers of 2. This mapping can be as efficient, and in many cases even more efficient in quantum chemistry calculations of ground states of molecular systems [26, 27]. In the parity representation, the nth qubit stores the sum of the parity of the first n modes [28, 29]. However, in applications to quantum chemistry, this scheme was not found to be particularly useful [29].

A first quantization approach would be more efficient in terms of the number of qubits required to describe the system. In this case, any particular one single particle state is encoded as a fixed combination of many qubit states, with the number of required qubits being given by $\approx \log_2(N_s)$. In each case, to describe each particle one needs the same number of qubits, so that the total number of qubits would be $N_q \approx n \log_2(N_s)$, with n the number of particles. It is clear that for this mapping, the scaling is more advantageous as $n \log_2(N_s) \ll N_s$ for $n \ll N_s$. The main disadvantage of this mapping, however, is that the many-body system is not automatically antisymmetric for a system of fermions. For a large number of particles this is especially challenging. However, a recent quantum algorithm [30] was proposed for antisymmetrizing identical fermions with a gate complexity of $O(\log^c n \log_2 \log_2 N_s)$ and a circuit size of $O(n \log^c n \log_2 N_s)$. The value of c depends on the choice of the specific algorithm.

The advantages of each mapping scheme are not obvious. One possible middle ground between the second and the first quantization mapping could be the scheme proposed in Ref. [31], in which one preserves the antisymmetrization of the basis states, thus eliminating the need for a complicated antisymmetrization first step. In addition, while the main justification for the work in Ref. [31] is particle-number conservation, other symmetries can be considered as well.

Thus, this approach is perfectly fitted for the shell model. In an occupation basis, the Hamiltonian in Eq. (1) becomes

$$H = \sum_{\alpha,\beta} \langle\alpha|H|\beta\rangle|\alpha\rangle\langle\beta|, \tag{5}$$

where $|\alpha\rangle$, $|\beta\rangle$ are many-body states that preserve not only the particle number, but also the projection of the total angular momentum in the M scheme. For the $j - j$ coupling, the many-body states $|\alpha\rangle$ and $|\beta\rangle$ have good total angular momentum; in this case, the matrix H is block diagonal, as $[H, J^2] = 0$. In the $j - j$ scheme the matrix that needs to be diagonalized is less sparse than in M scheme, but the dimension is smaller. While it is unclear how the trade off between the dimension that needs to be solved would map into quantum hardware, it is clear that many of the tools [32–35] developed for large-scale shell model calculations can be adopted for porting the same type of problems on quantum hardware.

Basis states generated in M or $j - j$ schemes are assigned to a certain combination of qubits. A similar mapping was used in Ref. [36] to solve for the deuteron in relative coordinates in a harmonic oscillator basis. In Table 1, we compare the number of qubits necessary to represent a nuclear system with N_p protons and N_n neutrons for Jordan-Wigner mapping (N_q^{JW}) and using encoding of the basis states into combinations of qubits (N_q^{SM}), with a fixed projection of the spin M_{tot}. It is immediately clear that while it might be more advantageous to encode the shell model basis, the number of Pauli strings in this mapping ($N_{\mathrm{Pauli}}^{\mathrm{SM}}$) quickly surpasses the number of Pauli strings necessary for the JW mapping ($N_{\mathrm{Pauli}}^{\mathrm{JW}}$). Thus, the advantage of using the qubit-efficient enconding quickly goes away in particular for the *sd* shell model space, where the number of single-particle states included in the calculation increases to 12 for each species, assuming an inter ^{16}O core, and using the "universal" SD interaction [37, 38]. Note that in Table 1 we used an arbitrary order for encoding, and not the one based on the Gray code that could potentially be more efficient [36]. Nevertheless, we do not expect the picture presented in Table 1 to dramatically change.

3 Algorithm Design

3.1 Projection Algorithm for State Preparation

Several algorithms are now available for preparing selected states on quantum hardware [16, 39–42]. A projection algorithm like the one introduced in Ref. [16] could be better suited to the nuclear physics problems, as it can be adapted to take into account even limited information about the nuclear spectrum.

All projection algorithms use a series of time evolutions and measurements of one or more ancilla qubits connected to the system in order to project on a desired state. To project the ground state, in Ref. [39], Ge et al. apply

Table 1: Comparison interaction mapping and number of qubits for several p and sd shell nuclei. $M_{tot} = 0$ for an even number of particles, $M_{tot} = 1/2$ for an odd number of particles.

Shell	N_p	N_n	N_q^{JW}	$N_{\text{Pauli}}^{\text{JW}}$	N_q^{SM}	$N_{\text{Pauli}}^{\text{SM}}$
p	1	2	12	975	5	528
p	2	2	12	975	6	2,072
p	1	3	12	975	5	488
p	2	3	12	975	6	2,080
p	3	3	12	975	7	7,936
sd	1	2	24	12,869	7	8,252
sd	1	3	24	12,869	9	131,321
sd	2	2	24	12,869	10	523,720

$\cos^M(\tilde{H}\pi/2)$ on a trial state, with $\tilde{H}$ shifted and re-scaled Hamitlonian H and M an integer. Thus, if M is large enough, the amplitude of any arbitrary state with energy E is reduced by $\cos^M(E\pi/2)$, ensuring the suppression of all but the state at $E = 0$. In a very simplistic approach, applying this type of filtering reduces to time evolution with constant time $t = \pi/2$ and measuring M ancilla states. A more efficient implementation based on $\log_2(M)$ ancilla qubits is presented in Ref. [39] and its implementation is discussed in Appendix A. In contrast, the Rodeo algorithm [41] exploits the fact that the probability of finding a state with energy E given by

$$P_N = \prod_{n=1}^{N} \cos^2\left[(E_{\text{target}} - E)\frac{t_n}{2}\right] \tag{6}$$

is suppressed by a factor $1/4^N$ if $E \neq E_{\text{target}}$ for a Gaussian distribution of times t_n and large N, with N the number of measurements. In the proposed implementation, the Rodeo algorithm is based on N controlled time evolution with the Hamiltonian governing the dynamics and normal distributed times.

Neither of the two algorithms briefly presented above uses any information about the system one wants to describe. It is well understood in imaginary time evolution approaches in classical calculations that in order to filter out contributions from undesired states, the evolution time is of the order of $1/\Delta$, where Δ is the gap between the target state and the first state with non-zero overlap with the trial state. A direct application on quantum computers of the imaginary time evolution is cumbersome, but the same general lessons can be applied using real-time evolution. In Ref. [16], we introduced a projection algorithm based on real-time evolution and using a set of optimized times and small phases. The optimization is based on the approximate knowledge of the many-body spectrum and the general assumption regarding the overlap of the eigenstates with the trial state. Since this approach was much more efficient in state preparation than other projection-based algorithms [16] even for small

gaps, in this work we use this procedure to construct the ground state of a many-body system.

In the simplest implementation of the energy filtering algorithm of Ref. [16], a single ancilla qubit is attached to the collection of qubits describing the physical system. The trial state $|\psi_0\rangle$ is usually a Hartree-Fock Slater determinant, which can be trivially represented on a quantum computer, and the ancilla qubit a is set to state $|0\rangle$. Then, one performs a series of time evolutions with times t_i followed by a measurement of the qubit a. The algorithm is successful if all measurements of the ancilla qubit produce state $|0\rangle$. At step i one produces the state $|\psi_i\rangle$ from state $|\psi_{i-1}\rangle$ as follows

$$
\begin{aligned}
|\psi_i\rangle &= \exp\left[-i(\tilde{H}t_i + \delta_i)Y_a\right]|\psi_{i-1}\rangle \otimes |0\rangle \\
&= \cos\left(\tilde{H}t_i + \delta_i\right)|\psi_{i-1}\rangle \otimes |0\rangle + \sin\left(\tilde{H}t_i + \delta_i\right)|\psi_{i-1}\rangle \otimes |1\rangle.
\end{aligned}
\tag{7}
$$

Knowledge of the spectrum, even approximative, is not required, but is useful. For example, the same algorithm can be used to project on quantum numbers. While projection on symmetries has been introduced before [43, 44], the algorithm based on Eq. (7) could be more efficient as a large number of quantum numbers can be eliminated at each iteration [16]. Moreover, since the projection on targeted quantum numbers usually increases the gap that controls the algorithm, the projection can become more effective with smaller propagation times, akin to classical calculations.

In the investigations presented in this paper, we did not perform symmetry projection. In turn, we have only projected the ground state, assumed at zero energy. To better understand the algorithm in Eq. (7), and the optimization procedure, we start with the trial vector decomposed in an orthogonal basis

$$
|\psi_0\rangle = \sum_\alpha C_\alpha |\alpha\rangle,
\tag{8}
$$

where $|\alpha\rangle$ are the (unknown) eigenstates of $\tilde{H}$ ($\tilde{H}|\alpha\rangle = E_\alpha|\alpha\rangle$), and C_α (unknown) complex coefficients. Time evolving the system using Eq. (7) produces the state

$$
\begin{aligned}
|\psi_1\rangle &= \exp\left[-i(\tilde{H}t_1 + \delta_1)Y_a\right]|\psi_0\rangle \otimes |0\rangle \\
&= \sum_\alpha C_\alpha|\alpha\rangle \otimes \left[\cos\left(E_\alpha t_1 + \delta_1\right)|0\rangle + \sin\left(E_\alpha t_1 + \delta_1\right)|1\rangle\right].
\end{aligned}
\tag{9}
$$

Since the ground state is shifted so that $E_0 = 0$, if the phases δ_i are kept small, measuring the ancilla qubit in state $|0\rangle$ will enhance the amplitude of the ground state with respect to the other states. If the gap is known from classical calculations, one can always take the time $t_1 = \pi/(2\Delta)$ and $\delta_1 = 0$, so that after the first measurement, the state at the gap is exactly removed from physical state (without the ancilla qubit). In the case the spectrum is known,

Table 2: A summary of numerical methods for simulating quantum circuits categorized by memory cost for qubits (Mem(Q)), memory cost for gates (Mem(G)), computational cost for qubits (Comp(Q)), and computational cost for gates (Comp(G)). Sparsity identifies if sparsity in the representation can be utilized, while Noise identifies if noise can be simulated.

Approach	Mem(Q)	Mem(G)	Comp(Q)	Comp(G)	Sparsity	Noise	Ref
State Vector	High	Low	High	Low	No	No	[46]
Density Matrix	High	Low	High	Low	No	Yes	[47]
Decision Diagram	Medium	Medium	Medium	High	Yes	No	[48]
Tensor Network	Medium	High	Medium	High	Yes	No	[49]
Stabilizer	Low	Low	Low	Low	Yes	No	[50]
Device Simulation	High	High	High	High	No	Yes	[51]

one can continue like in the case of projection on quantum numbers. However, for the more general case when the spectrum is not known, it was found that additional exponentially shorter times ($t_i = t_{i-1}/2$) are a reasonable choice to remove higher lying excitations [16], with a total evolution time approaching $2t_1$. If the suppression of the unwanted states is deemed unsatisfactory, one can repeat the same procedure, while running the circuit in Fig. 12 of Ref. [45] will provide information about the gap and other states present in the trial state, if desired.

The probability to produce the desired state is given by the probability to find that state into the trial state in the case of all phases zero. Using Eq. (9), the amplitude of each state after N measurements of the ancilla (all giving state $|0\rangle$) becomes $C'_\alpha = \prod_{i=1}^{N} \cos(E_\alpha t_i + \delta_i)C_\alpha$. To optimize the times and phases in order to speed up the calculation, one can start with the exponentially distributed times, assuming some reasonable overlap with the ground state, and random overlaps for the remaining states, and then maximize the final overlap for the final state. This optimization procedure can be classically performed, and general properties regarding the nuclear spectra can be used in the case when the spectrum is not *a priori* known. If non-zero phases are included in the optimization, they are generally small, and reduce the probability of success for the targeted state by $\prod_{i=0}^{N} \cos^2(\delta_i)$.

3.2 Numerical Simulation of Quantum Circuit

Generally speaking, there are multiple ways to numerically simulate the quantum circuits generated from a quantum algorithm, such as the projection algorithm described. These include state-vector [46, 52], density-matrix [47, 53], tensor-network [49, 54], decision diagrams [55, 56], stabilizer [50, 57], and device-level simulation such as pulse-based simulation [58]. Some of their features are summarized in Table 2, with respect to the cost of system memory and computation when scaling qubits (Q) and gates (G), as well as the ability to leverage sparsity in the representation and incorporating noise effect.

As discussed, the circuit for the time evolution and projection algorithm can be quite deep which is far beyond the capability and coherence time of current NISQ devices. For algorithm verification purposes without inspecting

noise effects, in this work we mainly focus on state-vector simulation given its resilience to circuit depth. In the following, we briefly introduce the fundamentals of state-vector representation and the general ways of performing state-vector based numerical simulation in a classical machine.

State-Vector Representation: A pure quantum state in a mathematical formulation corresponds to a vector in the Hilbert space. A quantum system in a superposition state thus can be represented as a linear combination of the (orthonormal) eigenstates according to some basis:

$$|\Psi(t)\rangle = \sum_s C_s(t)|\Phi_s\rangle$$

The coefficient C_s which corresponds to a particular eigenstate is a complex number thus allowing interference effects among states. Evolution of the quantum states is time dependent, governed by the evolution operators defined in the quantum circuit. State transitions are triggered by evolution operators of which each represents a gate. Through the gate sequence of the circuit, the quantum system evolves toward an objective state of interest, where measurement can be repeatedly applied for sampling the state. The squares of the coefficients sum-up to 1. The simulation approach for pure state is to use an array with complex numbers to represent the coefficients C_s, known as the state-vector. The size of the array depends on the number of eigenstates in the system, which scales as 2^n with respect to the number of qubits n. To ensure numerical accuracy, double-precision is necessary. The system is evolved by applying the gates, each denoting a unitary matrix that describes how the coefficients of certain eigenstates need to be adjusted. Once a gate is applied, the system transits to the next state:

$$|\Psi\rangle \rightarrow U|\Psi\rangle \tag{10}$$

The state-vector approach describes the fundamental evolution of a quantum system in an ideal scenario without noise impact, thus is widely used for quantum algorithm verification, which is the purpose of this work.

State-Vector Simulation: State-vector based simulation is to simulate the operations of applying a series of unitary operators $U_{m-1} \cdots U_1 U_0$ to the state-vector $|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i|i\rangle$ that describes the state of a quantum system. n is the number of qubits and m is the number of operations or gates. Here, a complex-valued double-precision floating-point vector $\vec{\alpha}$ of size 2^n is used to store the coefficients α_i, which costs 16×2^n bytes of memory in a classical computer. U_i with $i \in [0, m-1]$ is a 2×2 (for one-qubit gate) or 4×4 (for two-qubit gate) complex matrix. It has been shown that an arbitrary quantum circuit can be decomposed into 1-qubit and 2-qubit gates [59]. In fact, almost all real quantum devices execute 1-qubit or 2-qubit basis gates internally. For example, IBMQ adopts 1-qubit gate X, SX, RZ, ID and 2-qubit gate CX as the

basis gates; Rigetti uses 1-qubit gate `RX`, `RZ` and 2-qubit gate `CZ` for internal execution. Multi-qubit gates are decomposed into 1-qubit and 2-qubit gates.

To apply a gate U, the operation is $|\psi\rangle \to U|\psi\rangle$. For 1-qubit U applying on qubit q in a quantum register, $\vec{\alpha}$ is updated through the following expression where $s_i = \lfloor i/2^q \rfloor 2^{q+1} + (i\%2^q)$ for every integer $i \in [0, 2^{n-1} - 1]$:

$$\begin{bmatrix} \alpha_{s_i} \\ \alpha_{s_i+2^q} \end{bmatrix} \to U_{2\times 2} \cdot \begin{bmatrix} \alpha_{s_i} \\ \alpha_{s_i+2^q} \end{bmatrix} \tag{11}$$

Regarding 2-qubit unitary gate U applying on qubit p and q (assuming $p < q$ without losing generality), $\vec{\alpha}$ is updated through:

$$\begin{bmatrix} \alpha_{s_i} \\ \alpha_{s_i+2^p} \\ \alpha_{s_i+2^q} \\ \alpha_{s_i+2^p+2^q} \end{bmatrix} \to U_{4\times 4} \cdot \begin{bmatrix} \alpha_{s_i} \\ \alpha_{s_i+2^p} \\ \alpha_{s_i+2^q} \\ \alpha_{s_i+2^p+2^q} \end{bmatrix} \tag{12}$$

where $s_i = \lfloor \lfloor i/2^p \rfloor / 2^{q-p-1} \rfloor 2^{q+1} + (\lfloor i/2^p \rfloor \% 2^{q-p-1}) 2^{p+1} + (i\%2^p)$ for every integer $i \in [0, 2^{n-2} - 1]$.

To summarize, state-vector based quantum numerical simulation is to perform a sequence of 2×2 or 4×4 operations Eq. (11) and Eq. (12) over the large state-vector coefficient array of complex numbers. Note, although quantum gates without noise are all unitary operations, Eq. (11) and Eq. (12) can be more general and do not necessarily require U to be unitary, which offers the great opportunities of gate fusion, as will be discussed in Section 4.2.

4 Numerical Simulator Design

Our state-vector numerical simulator for low-energy nuclear state-preparation is developed from our previous work SV-Sim [46] of the NWQSim package. Considering the deep circuits and the unique amplification based algorithm through mid-circuit ancilla measurement, in this work, we propose two techniques for efficient simulation. We first introduce the simulator framework and then focus on the two techniques.

4.1 Simulator Framework

Fig. 1 illustrates the SV-Sim framework [46]. It offers both C++ and Python interfaces to support quantum programming environments such as Qiskit [60], Q# [61], QCOR [62], as well as quantum intermediate representations (IR) such as QASM [63] and QIR [64]. The backends include CPUs, GPUs, Xeon-Phis, and multi-node heterogeneous HPC clusters.

In terms of single-device backend implementation, the original SV-Sim harvests performance of heterogeneous accelerators such as GPUs through two major strategies: *homogeneous execution*, where GPU-side polymorphism is realized through a device functional pointer approach [46], so that all the
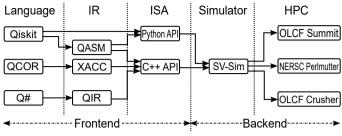
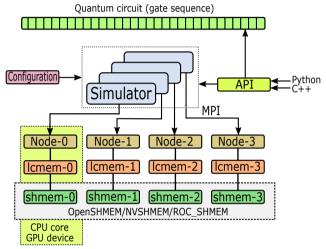**Fig. 1**: Simulation Framework.

Fig. 2: Architecture for the scaling-out of the SV-Sim simulator.

gate operations on the GPU side can be merged into a single GPU kernel, avoiding kernel creation, context switching, and data movement overhead. *Gate-specialized implementation*, where the speciality of the gate matrices, including sparsity and identity, are exploited. The new simulator designed in this work still benefits from homogeneous execution, but we could not use gate-specialization anymore due to gate fusion, as will be seen in Section 4.2.

In terms of the cluster backend implementation, SV-Sim mainly performs the communication through the shared-memory (SHMEM) model and inter-faces, such as NVSHMEM and OpenSHMEM. MPI is also used when necessary. Fig. 2 shows the architecture for the cluster implementation. Each SHMEM or MPI process owns a simulator object. Therefore, rather than using a single simulator instance to manage the multi-devices, each of them operates on a single CPU or GPU. The state-vector coefficients are evenly distributed among the devices. SHMEM/MPI are used for inter-device communication.

In terms of the frontend design, the state preparation algorithm described in Section 3 is implemented in Qiskit. Although SV-Sim directly supports

Table 3: Gates defined in IBM OpenQASM [63].

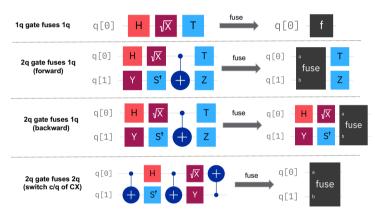| Gates	Meaning	Gates	Meaning
U3	3 parameter 2 pulse 1-qubit	CY	Controlled Y
U2	2 parameter 1 pulse 1-qubit	SWAP	Swap gate
U1	1 parameter 0 pulse 1-qubit	CH	Controlled H
CX	Controlled-NOT	CCX	Toffoli gate
ID	Idle gate or identity	CSWAP	Fredkin gate
X	Pauli-X bit flip	CRX	Controlled RX rotation
Y	Pauli-Y bit and phase flip	CRY	Controlled RY rotation
Z	Pauli-Z phase flip	CRZ	Controlled RZ rotation
H	Hadamard	CU1	Controlled phase rotation
S	sqrt(Z) phase	CU3	Controlled U3
SDG	Conjugate of sqrt(Z)	RXX	2-qubit XX rotation
T	sqrt(S) phase	RZZ	2-qubit ZZ rotation
TDG	Conjugate of sqrt(S)	RCCX	Relative-phase CXX
RX	X-axis rotation	RC3X	Relative-phase 3-controlled X
RY	Y-axis rotation	C3X	3-controlled X
RZ	Z-axis rotation	C3XSQRTX	3-controlled sqrt(X)
CZ	Controlled phase	C4X	4-controlled X

Fig. 3: 1-qubit and 2-qubit gate fusion operations for state-vector simulation.

Qiskit through the Python-API, given the extremely deep circuit, the parsing of the Qiskit gate object and performing gate conversion in Python can lead to considerable overhead. Consequently, we use the QASM frontend. Table 3 lists the gates defined by QASM [63]. Given such gate support, and the gate fusion technique to be presented, when generating QASM, no transpilation optimization is needed in Qiskit, which also improves performance.

4.2 Gate Fusion

The first challenge for the numerical simulation, as mentioned, is the deep circuit. We propose gate fusion to merge gates and shrink circuit depth. Given it is numerical simulations, we are not limited by the basis gates of a real device. We thus propose the following fusion operations:
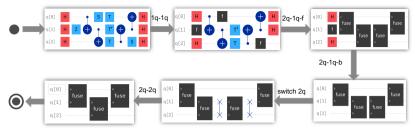
Fig. 4: Proposed gate fusion strategy.

1-qubit gates fuse 1-qubit gates: This implies that all consecutive 1-qubit gates applied on the same qubit can be merged into a single unitary gate (see Fig. 3). We implement a general 1-qubit gate labeled as `C1` in the simulator to perform the merged unitary gate.

2-qubit gates fuse 2-qubit gates: Similarly, we can merge consecutive 2-qubit gates over the same qubit pairs together into a single 2-qubit unitary gate. This includes the conditions of switching the control and target qubit of `CX` gates, as shown in Fig. 3. We realize a general 2-qubit gate labeled as `C2` in the simulator to run this gate.

2-qubit gates fuse 1-qubit gates: For state-vector simulation, it is also feasible for a 1-qubit gate to concatenate with a 1-qubit identity gate, forming a 2-qubit gate, and thus can be merged with another 2-qubit gate of the same qubit pair. This is similar to the 2-qubit gate "absorbs" a 1-qubit gate. Depending on the order of the 1-and 2-qubit gate, a forward or a backward fusion can be established, see Fig. 3. We use `C2` to execute the fused 2-qubit gate.

To effectively explore all fusion opportunities, we propose the following strategy to compose these fusion operations, shown in Fig. 4. We start by first merging consecutive 1-qubit gates applying to the same qubit. Then, we conduct forward and backward absorption for the 2-qubit gates to fuse the surrounding 1-qubit gate. After that, we add `SWAP` gates to ensure all 2-qubit gates are having the same partial order that the first qubit index is smaller than the second, i.e., `U(b,a)`→`SWAP(a,b)U(a,b)SWAP(a,b)`. This will facilitate 2-qubit 2-qubit fusion. We finally run 2-qubit fusion to obtain the ultimate circuit for simulation. We show in Section 5 about the efficiency of gate fusion.

4.3 Mid-Circuit Measurement Assertion

As discussed in Section 3.1, the projection algorithm works by asserting the ancilla qubit measuring $|0\rangle$, which will progressively enhance the amplitude of the ground state with respect to alternative states. When all ancilla qubits testing $|0\rangle$, the ground state is well prepared, which can be sampled. Listing 1 shows the structure of the QASM circuit generated from the projection algorithm. $q[0]$ is the ancilla qubit, which is measured (Line 11) and recycled (Line 13) repeatedly in the middle of the circuit. After the execution, from the

```
1 OPENQASM 2.0;
2 include "qelib1.inc";
3 qreg q[21];
4 creg c[11];
5 creg r[20];
6 //======= filtering step-1 =======
7 ry(7.36183164e-07) q[0];
8 ry(-7.579507071484729) q[0];
9 sdg q[0];
10 ...//lots of gates
11 measure q[0] -> c[0];
12 barrier q;
13 reset q[0];
14 barrier q;
15 //======= filtering step-2 =======
16 ry(1.26927712e-06) q0[0];
17 ry(-7.593605206153606) q0[0];
18 sdg q0[0];
19 ...//lots of gates
20 measure q[0] -> c[1];
21 ...//filtering step-n
22 //======= Final measurement =======
23 measure q -> r; //sample the prepared state for m shots
```

Listing 1: Block structure with repeated ancilla measurement and reset for the low-energy nuclear projection algoirthm in QASM

m measurement results of r, the algorithm picks those (labeled k) with respect to c being all-zero, which are the right samples of the ground state.

Looking at Listing 1, although the projection algorithm effectively reduces the ancilla usage to 1 qubit, which can significantly reduce simulation cost, it also introduces mid-circuit measurement and the potential dependency on the ancilla. For state-vector simulation of general circuits that only incurs measurement at the end, a well-known and key optimization is that one can simulate the circuit upon the measurement, and sample the final state-vector for the m shots together at once. Here, because of the mid-circuit measurement and dependency, we can only run one shot a time, drastically degrading the simulation efficiency. If the probability of c being all zero is relatively low (see Table 6, we either cannot obtain the state of interest ($k = 0$), or the sampling efficiency is low ($k \ll m$).

Therefore, in addition to gate fusion, we further modify SV-Sim to perform an algorithm-specific implementation. Recalling that for each ancilla measurement over $q[0]$, only $q[0]$ being $|0\rangle$ is of interest. Therefore, during simulation, after obtaining the probability of measuring $q[0]$, we can verify whether $q[0]$ still has any chances to be $|0\rangle$, i.e., $P(q[0] = |0\rangle) > 0$. If so, we simply assert the measurement result of sampling $q[0]$ is state $|0\rangle$, and track its probability $P(q[0] = |0\rangle)$. Otherwise, the simulation terminates and returns a failure. In this way, we can ensure c is always all-zero at the end. More importantly, despite we still need mid-circuit measurement, the dependency over $q[0]$ is resolved. We can simply run the simulation to the end, and sample m times at

Table 4: Evaluation platforms: NERSC Perlmutter, OLCF Summit and OLCF Crusher HPC systems. IC refers to intra-node interconnect. Network refers to inter-node network. Runtime refers to GPU runtime.

Platform	Nodes	CPU	Cores	DRAM	Compiler
Perlmutter	1500	AMD EPYC-7763	64	256 GB	nvc++ 21.11
Summit	4608	IBM Power-9	44	512 GB	gcc-9.3.0
Crusher	192	AMD EPYC-7A53	64	512 GB	gcc-12.2.0

Platform	GPUs	GPU Memory	GPU IC	Network	Runtime
Perlmutter	4 NVIDIA A100	40 GB HBM2	NVLink-V3	Slingshot	CUDA-11.5
Summit	6 NVIDIA V100	16 GB HBM2	NVLink-V2	InfiniBand	CUDA-11.1
Crusher	4 AMD MI250X	64 GB HBM2	InfinityFabric	Slingshot	ROCM-5.3.0

once, of which all of them are of interest (i.e., $m = k$), given c is ensured to be zero. Through this design, we can significantly improve simulation efficiency as well as sampling efficiency for the projection filtering algorithm.

5 Evaluation

5.1 Environment and Settings

We use the U.S. NERSC Perlmutter HPC cluster and the OLCF Summit and Crusher HPC clusters for the evaluation, as listed in Table 4.

Perlmutter: We use the Perlmutter system as the primary platform for the evaluation. Perlmutter is an HPE Cray EX pre-exascale HPC system based on the HPE Cray Shasta platform. This heterogeneous cluster comprises 1536 GPU nodes (each contains 4 NVIDIA A100 40GB GPUs), 256 advanced GPU nodes (4 A100 80GB GPUs), and 3072 CPU nodes (AMD EPYC 7763) linked by HPE Slingshot 11 high-speed interconnect. Each node is equipped with an AMD Milan EPYC 7763 64-core CPU.

Summit: We also use the OLCF Summit cluster. This IBM AC922 system contains more than 27,000 NVIDIA Volta GPUs with more than 9,000 IBM POWER9 CPUs. It has 4,608 nodes. Each node features two IBM POWER9 CPUs with 512 GB DDR4 main memory, and 6 Volta V100 GPUs with 16GB HBM2 memory per GPU. The intra-node interconnect is NVLink and the inter-node network is EDR 100GB InfiniBand.

Crusher: Crusher is an early-access testbed system at ORNL for exascale computing. It shares identical hardware and similar software as the Frontier HPC. Crusher has 2 cabinets, with 128 compute nodes and 64 compute nodes, respectively. Each node consists of a 64-core AMD EPYC 7A53 CPU, 512 GB of DDR4 main memory, and 4 AMD MI250X GPUs. The CPUs and GPUs are connected through Infinity Fabric inside a node. Different nodes are connected through 4 HPE Slingshot 25 GB/s NICs.

Regarding the problem settings, for testing purposes in this investigation, we used the phenomenological interacting shell model. In this many-body framework one assumes that only a small number of valence nucleons are interacting in a restricted model space via a phenomenological interaction. The rest of the

Table 5: Problem circuits for low-energy nuclear ground state preparation.

Problem	Nucleus	Qubits	Trotter Steps	Filtering Steps	Gates	2-Qubit Gates
P1	^{6}He	7	8	4	11,939	7,088
P2	^{6}He	7	14	4	20,885	12,404
P3	^{6}He	7	26	4	38,777	23,036
P4	^{6}He	7	47	4	70,088	41,642
P5	^{20}O	13	14	8	210,457	126,980
P6	^{20}O	13	24	8	360,767	217,680
P7	^{20}O	13	44	8	661,387	399,080
P8	^{20}O	13	83	8	1,247,596	752,810
P9	^{44}Ca	21	18	8	1,970,931	1,208,052
P10	^{44}Ca	21	30	8	3,284,871	2,013,420
P11	^{44}Ca	21	58	8	6,350,731	3,892,612
P12	^{44}Ca	21	108	8	11,825,481	7,248,312
P13	^{44}Ca	21	214	8	23,431,951	14,362,396
P14	^{44}Ca	21	528	8	57,813,381	35,436,192
P15	^{44}Ca	21	1051	8	115,079,266	70,536,814

nucleons are assumed to constitute an inert core. To simplify even more the problem, we present in this paper test cases where only neutrons are active. Thus, we have considered here two neutrons in the $0p$ shell (six active single-particle states) using the Cohen-Kurath interaction [22], four neutrons in the $1s0d$ model space (12 active single-particle states) using the "universal sd" Wildental interaction [37, 38], and four neutrons in the $1p0f$ shell (20 active states) using the modified KB3 interaction [65, 66]. Taking into account the ^{4}He core for the $0p$ shell, the ^{16}O core for the $1s0d$ shell, and the ^{40}Ca core for the $1p0f$ shell, ^{6}He, ^{20}O, and ^{44}Ca have been considered. For the $0p$ shell case, we also used a deformed Hartree-Fock solution computed with the code SHERPA [67].

With that, we formalize 15 problem instances for evaluating the simulation performance, covering a wide range of problem size from 7-qubit (including 1 ancilla), 11,939 gates to 21-qubit (1 ancilla), 115,079,266 gates. Their features are listed in Table 5. Their QASM file-size ranges from 272KB to 2.2GB, confirming that the circuit depth for these low-energy nuclear state preparation applications are much deeper than general circuits.

Our evaluation covers the following aspects: (i) *Gate Fusion*: We first evaluate the effectiveness of gate fusion, focusing on gate count reduction and simulation time reduction. (ii) *Mid-Circuit Measurement Assertion*: We show the savings of measurement times, and the improvement on sampling efficiency and simulation performance. (iii) *Performance across Platforms*: we show the simulation time on the CPUs and GPUs of the three HPC clusters. For the CPU performance, we show single-core single-thread, and multi-core with 4 threads using OpenMP. (iv) *Ground Energy and Sampling Efficiency*: we show the obtained ground state energy of the 15 problem instances with respect to different trotter steps.

5.2 Evaluation Results

(i) Gate Fusion: Using P1 to P8 in Table 5, Fig. 5 illustrates the reduction of gate count through gate fusion presented in Section 4.2. We progressively apply each fusion operation of Fig. 3, following the strategy in Fig. 4. After that, Fig. 6 shows the corresponding simulation time reduction with gate fusion, tested using an A100 GPU of Perlmutter.
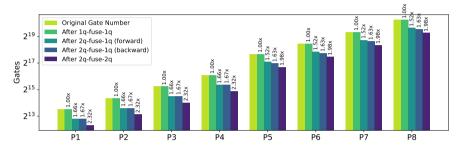

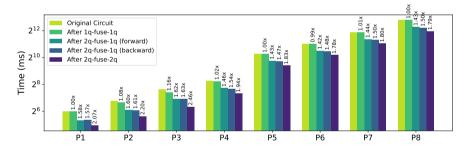**Fig. 5**: Reduction of gate count via the four gate fusion steps in Section 4.2.

Fig. 6: Performance improvement through the four gate fusion steps.

As can be seen, gate fusion can bring on average $2.15\times$ gate count reduction (range: $1.98\text{-}2.32\times$) across the 8 problem instances, corresponding to an on-average $1.98\times$ speedup (range $1.78\text{-}2.46\times$). Regarding these results, we have the following observations: (i) Circuits for different nuclei have specific patterns which may largely impact the benefit from gate reduction. Trotter steps, on the other hand, do not seem to impact the benefit from gate reduction. (ii) For this projection filtering algorithm, the percentage of two-qubit gates is quite high (on average 60.53%, see Table 5). Comparatively, there are much more opportunities for *2-qubit gates fuse 1-qubit gates* and *2-qubit gates fuse 2-qubit gates* than *1-qubit gates fuse 1-qubit gates*. The latter (2-qubit gates fuse 2-qubit gates) is particularly unique from general quantum circuits.

Table 6: Evaluation results for mid-circuit measurement assertion on P9.

	M1	M2	M3	M4	M5	M6	M7	M8	Success	Time(s)
Case-1	723	186	31	28	14	9	3	3	27	22553
Case-2	701	166	54	21	30	11	0	0	41	24147
Case-3	726	149	35	22	22	21	5	4	40	23891
MMA	0.29602	0.48617	0.69349	0.74823	0.73060	0.77238	0.93470	0.95811	0.037738	52.621

Overall, these results demonstrate that our gate fusion technique can effectively reduce the gate count and improve simulation performance for the low-energy nuclear ground state projection algorithm.

(ii) Mid-Circuit Measurement Assertion: We use P9 as a non-trivial exemplar case for this evaluation. Shown in Listing 1, the projection circuit includes a series of mid-circuit ancilla measurements to amplify the probability of the ground state, given the measurement result is 0. However, it is also possible that the measurement gives 1, which implies that the present shot fails to project to the correct state, seen as a rejection. As shown in Table 5, the filtering steps for P9 is 8, implying 8 mid-circuit measurements, labeled as M1 to M8. In Table 6, we repeat three simulations (Case-1 to 3) of the P5 circuit on a Perlmutter A100 GPU using 1024 shots without applying mid-circuit measurement assertion. The values under M1 to M8 show the number of rejected shots during each mid-circuit measurement.

Within the three trails, for P9, on average 36 of the 1024 shots can project to the final state of interest, implying a sampling efficiency of 3.52% for the projection algorithm. Additionally, because of repeated end-to-end runs due to mid-circuit measurement, the simulation time even on an A100 GPU is still quite significant, on-average 392 minutes. By applying mid-circuit measurement assertion, despite the probability of measuring all-zero remains low, we can always assert it measures 0 and tracking the probability.

The last row of Table 6 shows the ground truth probability of measuring 0 during each mid-circuit measurement. Overall, the probability of obtaining all 0s for M1 to M8 is 3.77%. With mid-circuit measurement assertion, we can assert the probability, and ensure the ground state is well-prepared. We can then sample the state by 1024 shots at once in parallel, providing 1024 effective samples. Comparatively, to obtain the same number of effective samples, the original method would need $1024/0.0377 = 27,162$ shots on average and have to be executed sequentially due to mid-circuit measurement.

Overall, the simulation time with MMA is 52.621s, 447× faster than the baseline. When demanding the same number of effective samples, e.g., 1024, the speedup is about 12,719×. Although the benefit is circuit dependent, it confirms the usefulness and effectiveness of mid-circuit measurement assertion. For larger problems or problems with even lower success rate, the gains can be even more extraordinary.

(iii) Performance across Platforms: We perform the simulations over the three CPUs and three GPUs of the Perlmutter, Summit and Crusher systems listed in Table 4. Fig. 7 shows the simulation time of the P1, P5 and P9 circuits
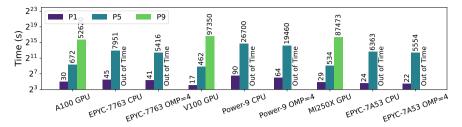
Fig. 7: Simulation time for P1, P5 and P9 across the CPUs and GPUs of Perlmutter, Summit and Crusher.

Table 7: Results of ground state energy in MeV.

Problem	Nucleus	Trotter	Success Rate	Reference E(MeV)	Result E(MeV)	Kernel Time
P1	^{6}He	8	26.90%	-3.90983149	-2.64541903	30.8ms
P2	^{6}He	14	40.87%	-3.90983149	-3.36176356	39.4ms
P3	^{6}He	26	40.35%	-3.90983149	-3.80798262	80.4ms
P4	^{6}He	47	37.33%	-3.90983149	-3.90193879	158.4ms
P5	^{20}O	14	27.25%	-35.26663264	-34.89780987	672.8ms
P6	^{20}O	24	38.17%	-35.26663264	-35.14916931	1.16s
P7	^{20}O	44	41.69%	-35.26663264	-35.22824245	2.10s
P8	^{20}O	83	42.53%	-35.26663264	-35.25485687	3.95s
P9	^{44}Ca	14	3.77%	-5.00586901	-4.33397239	52.62s
P10	^{44}Ca	30	10.01%	-5.00586901	-4.53559332	87.68s
P11	^{44}Ca	58	12.92%	-5.00586901	-4.85291636	169.50s
P12	^{44}Ca	108	13.60%	-5.00586901	-4.91713907	315.59s
P13	^{44}Ca	214	13.76%	-5.00586901	-4.93915187	625.60s
P14	^{44}Ca	528	13.76%	-5.00586901	-4.94418587	25.7min
P15	^{44}Ca	1051	13.74%	-5.00586901	-4.94454380	56.5min

on the three GPUs (A100, V100 and MI250X) and CPUs (EPYC-7763, Power-9 and EPYC-7A53 with 1 and 4 threads). We put the raw number of simulation latency (in ms) above the bars. Note, P9 runs too long on the CPUs so we omit them. As can be seen, for small cases such as 7-qubit P1, GPUs do not exhibit any performance advantages. For larger cases, such as 13-qubit P5, the performance of GPUs is well above CPUs. For even larger problems such as the 21-qubit P9, A100 is better than MI250X, and then V100, implying that A100 requires sufficient workload to deliver superior simulation performance.

(iv) Ground Energy and Sampling Efficiency: Table 7 lists the ground state energy obtained through our simulation, the reference ground energy, and the theoretical successful rate of the projection filtering algorithm. We also list the GPU kernel execution time for the simulation. All the results here are obtained on an A100 GPU of Perlmutter.

As can be seen, the high success rates indicate that the projection filtering algorithm is quite effective in converging to the ground state energy, even for non-trivial problems. Additionally, with only a few filtering steps, i.e., 4 for ^{6}He and 8 for ^{20}O and ^{44}Ca, we can already achieve an accuracy with less

than 0.01 MeV, 0.01 MeV, and 0.1 MeV, respectively. It also confirms that the simulator can generate the correct energy number with proper trotter steps. We also compare the SV-Sim energy results with Qiskit simulation results, they are aligned with each other to 10^{-14}, further confirming the numerical accuracy of our simulation. Meanwhile, it is interesting to observe that for ^{44}Ca, increasing the trotter steps from 528 to 1051 only brings very incremental gain of 0.0004 MeV, implying marginal impact. We set the investigation of impact from trotter steps as a future work.

6 Conclusion

Numerical simulation represents an important method for direct verification of quantum circuit correctness. Here we have demonstrated verification of deep quantum circuits for nuclear physics applications using a high-performance numerical simulator that incorporates several unique methods. The first is the use of gate fusion to reduce the effective simulation depth and, therefore, reduce the amount of computation required to generate the quantum state. We have described and demonstrated the use of 1-qubit and 2-qubit gate fusion to reduce the simulation depth of circuits used to prepare nuclear state ansatz. The second method manages the simulated state in the presence of mid-circuit measurements. We simulated the role of mid-circuit measurement by following the complete simulated state and using post-selection of the designated ancilla to calculate the projected state as well as the probability of the outcome.

Together these methods have permitted the verification of quantum circuits derived from a state preparation algorithm using an energy-filtering algorithm. We demonstrated simulation of several examples of deep circuits acting on 21 qubits with more than 115,000,000 gates. We have also tested these examples on a variety of high-performance computing systems that emphasize the benefits for GPU-accelerated processing. For future work, we would like to investigate the simulation of larger nucleus problems, such as ^{26}Mg, ^{56}Fe and ^{58}Ni, etc., and the impact of accuracy from the filtering and the trotter steps.

In conclusion, we have extended the capabilities for numerical simulation of deep quantum circuits for nuclear physics using high-performance computing systems. These methods provide powerful tools for verification of quantum circuit design.

Acknowledgement

A Implementation of the Ge *et al.* algorithm

We briefly review here the algorithm used for state preparation that follows the work of Ref. [39] and describe how has been implemented. We consider an Hamiltonian with spectrum in an interval $I \in (0,1)$, spectral gap Δ, an initial trial state $|\Phi\rangle$ with overlap χ with the ground state $|\Psi_0\rangle$, and we assume we know the ground state energy E. After defining the shifted Hamiltonian $H' = H - E\,\mathbf{1}$ the state defined below

$$|\tilde{\Phi}\rangle = \frac{\cos^M(H')}{||\cos^M(H')||}|\Phi\rangle \tag{13}$$

becomes close to the exact ground state, if the power of M is chosen as in Ref. [39]. It is possible to approximate the operator in Eq. (13) as a linear combination of unitaries in the following way:

$$\cos^{2m} H' = \sum_{k=-m_0}^{m_0} \alpha_k e^{-2iHk} + R, \tag{14}$$

with

$$\alpha_k = \frac{1}{4^m}\binom{2m}{m+k}. \tag{15}$$

and m_0 properly chosen in order to vanishing of quantity R, a prescription is provided in Ref. [39]. The implementation of the above operator can be done using Linear Combination of Unitaries using the Prepare and Select oracles of Ref. [68] and briefly reviewed below. Given the $2m_0 + 1$ unitaries of Eq. 14 we define an ancillary register of dimension $n_A = \lceil \log_2(2m_0 + 1) \rceil$. We can implement a block encoding of the linear combination of unitaries using the algorithm originally developed in Ref. [69, 70]and using the approach reported in Ref. [71]. In particular we first need a prepare operator P acting only on the ancillary register defined by the following equation:

$$P|0\rangle = \frac{1}{\sqrt{\alpha}}\sum_{k=0}^{2m_0}(\alpha_{-2m_0+k})^{1/2}|k\rangle, \tag{16}$$

and $\alpha = \sum_{k=-m_0}^{m_0} |\alpha_k|$. Following Ref. [71] for the general case the gate decomposition of the unitary P can be done using generic circuit synthesis as originally reported in Ref. [72] whose exponential scaling should not be a limitation for the problems at hand (i. e. for 10^3 unitaries only 10 ancillary qubits will be needed). After we define the select oracle, acting both on the ancillary register and the target register, in the following way

$$S = \sum_{k=0}^{2m_0+1} |k\rangle\langle k| \otimes U^k \, , \tag{17}$$

where $U = e^{-2iH}$. Although the implementetion in principle requires applying several multi-controlled unitaries, in Ref. [71], Lemma 3.5, has been shown that only a number of single controlled unitaries equal to the number of ancillary qubits are necessary and sufficient. Therefore the implementation of the LCU method for the problem at hand can be expressed as in Fig. 2 of Ref. [71]. We are now in the position to discuss the success probability of the procedure similarly to Ref. [68] we can define the following quantity

$$\eta^2 = \langle\Phi|O^2|\Phi\rangle \, , \tag{18}$$

where we have defined the operator $O = \sum_{k=-m_0}^{m_0} \alpha_k U^k$. The success probability of postselecting all $0s$ on the ancillary qubit is

$$P_s = \frac{\eta^2}{\alpha^2} \, . \tag{19}$$

References

[1] Cloët, I.C., Dietrich, M.R., Arrington, J., Bazavov, A., Bishof, M., Freese, A., Gorshkov, A.V., Grassellino, A., Hafidi, K., Jacob, Z., et al.: Opportunities for nuclear physics & quantum information science. arXiv preprint arXiv:1903.05453 (2019)

[2] Beck, D., Carlson, J., Davoudi, Z., Formaggio, J., Quaglioni, S., Savage, M., Barata, J., Bhattacharya, T., Bishof, M., Cloet, I., et al.: Quantum information science and technology for nuclear physics. input into us long-range planning, 2023. arXiv preprint arXiv:2303.00113 (2023)

[3] Zhang, D.-B., Xing, H., Yan, H., Wang, E., Zhu, S.-L.: Selected topics of quantum computing for nuclear physics. Chinese Physics B **30**(2), 020306 (2021). https://doi.org/10.1088/1674-1056/abd761

[4] Cervia, M.J., Balantekin, A.B., Coppersmith, S.N., Johnson, C.W., Love, P.J., Poole, C., Robbins, K., Saffman, M.: Lipkin model on a quantum computer. Phys. Rev. C **104**, 024305 (2021). https://doi.org/10.1103/PhysRevC.104.024305

 [5] Qian, W., Basili, R., Pal, S., Luecke, G., Vary, J.P.: Solving hadron structures using the basis light-front quantization approach on quantum computers. Phys. Rev. Res. **4**, 043193 (2022). https://doi.org/10.1103/PhysRevResearch.4.043193

 [6] Stetcu, I., Baroni, A., Carlson, J.: Variational approaches to constructing the many-body nuclear ground state for quantum computing. Phys. Rev. C **105**, 064308 (2022). https://doi.org/10.1103/PhysRevC.105.064308

 [7] Romero, A.M., Engel, J., Tang, H.L., Economou, S.E.: Solving nuclear structure problems with the adaptive variational quantum algorithm. Phys. Rev. C **105**, 064317 (2022). https://doi.org/10.1103/PhysRevC.105.064317

 [8] Dumitrescu, E.F., McCaskey, A.J., Hagen, G., Jansen, G.R., Morris, T.D., Papenbrock, T., Pooser, R.C., Dean, D.J., Lougovski, P.: Cloud quantum computing of an atomic nucleus. Phys. Rev. Lett. **120**, 210501 (2018). https://doi.org/10.1103/PhysRevLett.120.210501

 [9] Siwach, P., Arumugam, P.: Quantum computation of nuclear observables involving linear combinations of unitary operators. Phys. Rev. C **105**, 064318 (2022). https://doi.org/10.1103/PhysRevC.105.064318

[10] de Jong, W.A., Lee, K., Mulligan, J., Płosko ń, M., Ringer, F., Yao, X.: Quantum simulation of nonequilibrium dynamics and thermalization in the schwinger model. Phys. Rev. D **106**, 054508 (2022). https://doi.org/10.1103/PhysRevD.106.054508

[11] Kiss, O., Grossi, M., Lougovski, P., Sanchez, F., Vallecorsa, S., Papenbrock, T.: Quantum computing of the ^{6}Li nucleus via ordered unitary coupled clusters. Physical Review C **106**(3) (2022). https://doi.org/10.1103/PhysRevC.106.034325

[12] Tacchino, F., Chiesa, A., Carretta, S., Gerace, D.: Quantum computers as universal quantum simulators: State-of-the-art and perspectives. Advanced Quantum Technologies **3**(3), 1900052 (2020). https://doi.org/10.1002/qute.201900052

[13] Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S.C., Endo, S., Fujii, K., McClean, J.R., Mitarai, K., Yuan, X., Cincio, L., *et al.*: Variational quantum algorithms. Nature Reviews Physics **3**(9), 625–644 (2021)

[14] Raeisi, S., Wiebe, N., Sanders, B.C.: Quantum-circuit design for efficient simulations of many-body quantum dynamics. New Journal of Physics **14**(10), 103017 (2012). https://doi.org/10.1088/1367-2630/14/10/103017

[15] Pérez-Obiol, A., Romero, A.M., Menéndez, J., Rios, A., García-Sáez, A., Juliá-Díaz, B.: Nuclear shell-model simulation in digital quantum computers. Scientific Reports **13**, 12291 (2023). https://doi.org/10.1038/s41598-023-39263-7

[16] Stetcu, I., Baroni, A., Carlson, J.: Projection algorithm for state preparation on quantum computers. arXiv e-prints, 2211–10545 (2022) arXiv:2211.10545 [quant-ph]. https://doi.org/10.48550/arXiv.2211.10545

[17] Kiss, O., Grossi, M., Roggero, A.: Importance sampling for stochastic quantum simulations. Quantum **7**, 977 (2023). https://doi.org/10.22331/q-2023-04-13-977

[18] Wright, J., Gowrishankar, M., Claudino, D., Lotshaw, P.C., Nguyen, T., McCaskey, A.J., Humble, T.S.: Numerical simulations of noisy quantum circuits for computational chemistry. Materials Theory **6**(1), 18 (2022)

[19] Piarulli, M., Baroni, A., Girlanda, L., Kievsky, A., Lovato, A., Lusk, E., Marcucci, L.E., Pieper, S.C., Schiavilla, R., Viviani, M., Wiringa, R.B.: Light-nuclei spectra from chiral dynamics. Phys. Rev. Lett. **120**, 052503 (2018). https://doi.org/10.1103/PhysRevLett.120.052503

[20] Lonardoni, D., Carlson, J., Gandolfi, S., Lynn, J.E., Schmidt, K.E., Schwenk, A., Wang, X.B.: Properties of nuclei up to $a = 16$ using local chiral interactions. Phys. Rev. Lett. **120**, 122502 (2018). https://doi.org/10.1103/PhysRevLett.120.122502

[21] Maris, P., Roth, R., Epelbaum, E., Furnstahl, R.J., Golak, J., Hebeler, K., Hüther, T., Kamada, H., Krebs, H., Le, H., Meißner, U.-G., Melendez, J.A., Nogga, A., Reinert, P., Skibi ński, R., Vary, J.P., Witała, H., Wolfgruber, T.: Nuclear properties with semilocal momentum-space regularized chiral interactions beyond n^{2}LO. Phys. Rev. C **106**, 064002 (2022). https://doi.org/10.1103/PhysRevC.106.064002

[22] Cohen, S., Kurath, D.: Effective interactions for the 1p shell. Nuclear Physics **73**(1), 1–24 (1965). https://doi.org/10.1016/0029-5582(65)90148-3

[23] Jordan, P., Wigner, E.: Über das paulische äquivalenzverbot. Zeitschrift für Physik **47**(9), 631–651 (1928). https://doi.org/10.1007/BF01331938

[24] Abramowitz, M., Stegun, I.A.: Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, ninth Dover printing, tenth GPO printing edn. Dover, New York (1964)

[25] Bravyi, S.B., Kitaev, A.Y.: Fermionic quantum computation. Annals of Physics **298**(1), 210–226 (2002). https://doi.org/10.1006/aphy.2002.6254

[26] Tranter, A., Sofia, S., Seeley, J., Kaicher, M., McClean, J., Babbush, R., Coveney, P.V., Mintert, F., Wilhelm, F., Love, P.J.: The bravyi–kitaev transformation: Properties and applications. International Journal of Quantum Chemistry **115**(19), 1431–1441 (2015) https://onlinelibrary.wiley.com/doi/pdf/10.1002/qua.24969. https://doi.org/10.1002/qua.24969

[27] Tranter, A., Love, P.J., Mintert, F., Coveney, P.V.: A comparison of the bravyi–kitaev and jordan–wigner transformations for the quantum simulation of quantum chemistry. Journal of chemical theory and computation **14**(11), 5617–5630 (2018)

[28] Seeley, J.T., Richard, M.J., Love, P.J.: The Bravyi-Kitaev transformation for quantum computation of electronic structure. The Journal of Chemical Physics **137**(22), 224109 (2012) https://doi.org/10.1063/1.4768229. https://doi.org/10.1063/1.4768229

[29] McArdle, S., Endo, S., Aspuru-Guzik, A., Benjamin, S.C., Yuan, X.: Quantum computational chemistry. Rev. Mod. Phys. **92**, 015003 (2020). https://doi.org/10.1103/RevModPhys.92.015003

[30] Berry, D.W., Kieferová, M., Scherer, A., Sanders, Y.R., Low, G.H., Wiebe, N., Gidney, C., Babbush, R.: Improved techniques for preparing eigenstates of fermionic hamiltonians. npj Quantum Information **4**(1), 22 (2018). https://doi.org/10.1038/s41534-018-0071-5

[31] Shee, Y., Tsai, P.-K., Hong, C.-L., Cheng, H.-C., Goan, H.-S.: Qubit-efficient encoding scheme for quantum simulations of electronic structure. Phys. Rev. Res. **4**, 023154 (2022). https://doi.org/10.1103/PhysRevResearch.4.023154

[32] Vary, J.P.: The Many-Fermion-Dynamics Shell-Model Code. The Many-Fermion-Dynamics Shell-Model Code, Iowa State University (unpublished) (1992)

[33] Brown, B.A., Rae, W.D.M.: The shell-model code nushellx@msu. Nuclear Data Sheets **120**, 115–118 (2014). https://doi.org/10.1016/j.nds.2014.07.022

[34] Dytrych, T., Maris, P., Launey, K.D., Draayer, J.P., Vary, J.P., Langr, D., Saule, E., Caprio, M.A., Catalyurek, U., Sosonkina, M.: Efficacy of the su(3) scheme for ab initio large-scale calculations beyond the lightest nuclei. Computer Physics Communications **207**, 202–210 (2016). https://doi.org/10.1016/j.cpc.2016.06.006

[35] Johnson, C.W., Ormand, W.E., McElvain, K.S., Shan, H.: BIGSTICK: A

flexible configuration-interaction shell-model code. arXiv e-prints, 1801–08432 (2018) arXiv:1801.08432 [physics.comp-ph]

[36] Di Matteo, O., McCoy, A., Gysbers, P., Miyagi, T., Woloshyn, R.M., Navrátil, P.: Improving hamiltonian encodings with the gray code. Phys. Rev. A **103**, 042405 (2021). https://doi.org/10.1103/PhysRevA.103.042405

[37] Wildenthal, B.H.: Empirical strengths of spin operators in nuclei. Progress in Particle and Nuclear Physics **11**, 5–51 (1984). https://doi.org/10.1016/0146-6410(84)90011-5

[38] Brown, B.A., Richter, W.A.: New "USD" Hamiltonians for the *sd* shell. Phys. Rev. C **74**, 034315 (2006). https://doi.org/10.1103/PhysRevC.74.034315

[39] Ge, Y., Tura, J., Cirac, J.I.: Faster ground state preparation and high-precision ground energy estimation with fewer qubits. J. Math. Phys. **60**, 022202 (2019). https://doi.org/10.1063/1.5027484

[40] Motta, M., Sun, C., Tan, A.T.K., O'Rourke, M.J., Ye, E., Minnich, A.J., Brandão, F.G.S.L., Chan, G.K.-L.: Determining eigenstates and thermal states on a quantum computer using quantum imaginary time evolution. Nature Physics **16**(2), 205–210 (2020) arXiv:1901.07653 [quant-ph]. https://doi.org/10.1038/s41567-019-0704-4

[41] Choi, K., Lee, D., Bonitati, J., Qian, Z., Watkins, J.: Rodeo algorithm for quantum computing. Phys. Rev. Lett. **127**, 040505 (2021). https://doi.org/10.1103/PhysRevLett.127.040505

[42] Jouzdani, P., Johnson, C.W., Mucciolo, E.R., Stetcu, I.: Alternative approach to quantum imaginary time evolution. Phys. Rev. A **106**, 062435 (2022). https://doi.org/10.1103/PhysRevA.106.062435

[43] Lacroix, D.: Symmetry-assisted preparation of entangled many-body states on a quantum computer. Phys. Rev. Lett. **125**, 230502 (2020). https://doi.org/10.1103/PhysRevLett.125.230502

[44] Ruiz Guzman, E.A., Lacroix, D.: Accessing ground-state and excited-state energies in a many-body system after symmetry restoration using quantum computers. Phys. Rev. C **105**, 024324 (2022). https://doi.org/10.1103/PhysRevC.105.024324

[45] Somma, R., Ortiz, G., Gubernatis, J.E., Knill, E., Laflamme, R.: Simulating physical phenomena by quantum networks. Phys. Rev. A **65**, 042323 (2002). https://doi.org/10.1103/PhysRevA.65.042323

[46] Li, A., Fang, B., Granade, C., Prawiroatmodjo, G., Heim, B., Roetteler, M., Krishnamoorthy, S.: Sv-sim: scalable pgas-based state vector simulation of quantum circuits. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–14 (2021)

[47] Li, A., Subasi, O., Yang, X., Krishnamoorthy, S.: Density matrix quantum circuit simulation via the bsp machine on modern gpu clusters. In: SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–15 (2020). IEEE

[48] Grurl, T., Kueng, R., Fuß, J., Wille, R.: Stochastic quantum circuit simulation using decision diagrams. In: 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 194–199 (2021). IEEE

[49] Nguyen, T., Lyakh, D., Dumitrescu, E., Clark, D., Larkin, J., McCaskey, A.: Tensor network quantum virtual machine for simulating quantum circuits at exascale. arXiv preprint arXiv:2104.10523 (2021)

[50] Aaronson, S., Gottesman, D.: Improved simulation of stabilizer circuits. Physical Review A **70**(5), 052328 (2004)

[51] Fu, X., Rol, M.A., Bultink, C.C., Van Someren, J., Khammassi, N., Ashraf, I., Vermeulen, R., De Sterke, J., Vlothuizen, W., Schouten, R., *et al.*: An experimental microarchitecture for a superconducting quantum processor. In: Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture, pp. 813–825 (2017)

[52] Jones, T., Brown, A., Bush, I., Benjamin, S.C.: Quest and high performance simulation of quantum computers. Scientific reports **9**(1), 1–11 (2019)

[53] Chen, Y.-T., Farquhar, C., Parrish, R.M.: Low-rank density-matrix evolution for noisy quantum circuits. npj Quantum Information **7**(1), 1–12 (2021)

[54] Markov, I.L., Shi, Y.: Simulating quantum computation by contracting tensor networks. SIAM Journal on Computing **38**(3), 963–981 (2008)

[55] Miller, D.M., Thornton, M.A., Goodman, D.: A decision diagram package for reversible and quantum circuit simulation. In: 2006 IEEE International Conference on Evolutionary Computation, pp. 2428–2435 (2006). IEEE

[56] Grurl, T., Fuß, J., Wille, R.: Considering decoherence errors in the simulation of quantum circuits using decision diagrams. In: Proceedings of the 39th International Conference on Computer-Aided Design, pp. 1–7 (2020)

[57] Bravyi, S., Browne, D., Calpin, P., Campbell, E., Gosset, D., Howard, M.: Simulation of quantum circuits by low-rank stabilizer decompositions. Quantum **3**, 181 (2019)

[58] McKay, D.C., Alexander, T., Bello, L., Biercuk, M.J., Bishop, L., Chen, J., Chow, J.M., Córcoles, A.D., Egger, D., Filipp, S., et al.: Qiskit backend specifications for openqasm and openpulse experiments. arXiv preprint arXiv:1809.03452 (2018)

[59] Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D.P., Margolus, N., Shor, P., Sleator, T., Smolin, J.A., Weinfurter, H.: Elementary gates for quantum computation. Physical review A **52**(5), 3457 (1995)

[60] Cross, A.: The ibm q experience and qiskit open-source quantum computing software. In: APS March Meeting Abstracts, vol. 2018, pp. 58–003 (2018)

[61] Microsoft: What are Q# and the Quantum Development Kit? https://learn.microsoft.com/en-us/azure/quantum/overview-what-is-qsharp-and-qdk. Accessed: 2023-08-16

[62] Mintz, T.M., Mccaskey, A.J., Dumitrescu, E.F., Moore, S.V., Powers, S., Lougovski, P.: Qcor: A language extension specification for the heterogeneous quantum-classical model of computation. ACM Journal on Emerging Technologies in Computing Systems (JETC) **16**(2), 1–17 (2020)

[63] Cross, A.W., Bishop, L.S., Smolin, J.A., Gambetta, J.M.: Open quantum assembly language. arXiv preprint arXiv:1707.03429 (2017)

[64] Microsoft: Quantum intermediate representation (2023). https://learn.microsoft.com/en-us/azure/quantum/concepts-qir

[65] Kuo, T.T.S., Brown, G.E.: Reaction matrix elements for the 0f-1p shell nuclei. Nuclear Physics A **114**(2), 241–279 (1968). https://doi.org/10.1016/0375-9474(68)90353-9

[66] Poves, A., Zuker, A.: Theoretical spectroscopy and the fp shell. Physics Reports **70**(4), 235–314 (1981). https://doi.org/10.1016/0370-1573(81)90153-8

[67] Stetcu, I., Johnson, C.W.: Random phase approximation vs exact shell-model correlation energies. Phys. Rev. C **66**, 034301 (2002). https://doi.org/10.1103/PhysRevC.66.034301

[68] Roggero, A., Gu, C., Baroni, A., Papenbrock, T.: Preparation of excited states for nuclear dynamics on a quantum computer. Phys. Rev. C **102**,

064624 (2020). https://doi.org/10.1103/PhysRevC.102.064624

[69] Childs, A.M., Wiebe, N.: Hamiltonian simulation using linear combinations of unitary operations **12**(11–12), 901–924 (2012)

[70] Childs, A.M., Kothari, R., Somma, R.D.: Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. SIAM Journal on Computing **46**(6), 1920–1950 (2017) https://doi.org/10.1137/16M1087072. https://doi.org/10.1137/16M1087072

[71] Holmes, Z., Muraleedharan, G., Somma, R.D., Subasi, Y., Şahinoğlu, B.: Quantum algorithms from fluctuation theorems: Thermal-state preparation. arXiv e-prints, 2203–08882 (2022) arXiv:2203.08882 [quant-ph]

[72] Shende, V.V., Bullock, S.S., Markov, I.L.: Synthesis of quantum-logic circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **25**(6), 1000–1010 (2006). https://doi.org/10.1109/TCAD.2005.855930