Qubit Movement-Optimized Program Generation on Zoned Neutral Atom Processors

Enhyeok Jang
Yonsei University
Seoul, Republic of Korea
enhyeok.jang@yonsei.ac.kr

Youngmin Kim
Yonsei University
Seoul, Republic of Korea
youngmin.kim@yonsei.ac.kr

Hyungseok Kim
Yonsei University
Seoul, Republic of Korea
kimnumber@yonsei.ac.kr

Seungwoo Choi
Yonsei University
Seoul, Republic of Korea
seungwoo.choi@yonsei.ac.kr

Yipeng Huang
Rutgers University
New Brunswick, USA
yipeng.huang@rutgers.edu

Won Woo Ro
Yonsei University
Seoul, Republic of Korea
wro@yonsei.ac.kr

Abstract

A zoned neutral atom architecture achieves exceptional fidelity by segregating the execution spaces of 1- and 2-qubit gates, being a promising candidate for high-accuracy quantum systems. Unfortunately, naïvely applying programs designed for static qubit topologies to zoned architectures may result in most execution time being consumed by intra-zone travels of atoms. To address this, we introduce *Mantra* (Minimizing trAp movemeNts for aTom aRray Architectures), which rewrites quantum programs to reduce the interleaving of single- and two-qubit gates. *Mantra* incorporates three strategies: (i) a fountain-shaped controlled-Z (CZ) chain, (ii) ZZ-interaction protocol without a 1-qubit gate, and (iii) preemptive gate scheduling. *Mantra* reduces inter-zone movements by 68%, physical gate counts by 35%, and improves circuit fidelities by 17% compared to the standard executions.

CCS Concepts: • **Hardware** → **Emerging architectures**.

Keywords: Rydberg Atom-Based Quantum Computer, Zoned Neutral Atom Architectures, Quantum Program Rewriting

ACM Reference Format:
Enhyeok Jang, Youngmin Kim, Hyungseok Kim, Seungwoo Choi, Yipeng Huang, and Won Woo Ro. 2025. Qubit Movement-Optimized Program Generation on Zoned Neutral Atom Processors. In *Proceedings of the 23rd ACM/IEEE International Symposium on Code Generation and Optimization (CGO '25), March 01–05, 2025, Las Vegas, NV, USA.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3696443.3708937

1 Introduction

Quantum computers utilizing neutral atoms manipulated by optical tweezer arrays can offer a scalable architecture thanks to their flexible reconfigurability, long coherence time of qubits, and parallel gate execution capabilities [6, 22, 27, 69]. One of the recent advancements in this domain is the zoned neutral atom-based qubit architecture, which significantly enhances the fidelity of quantum gates by separating the execution spaces between 1-qubit gates and 2-qubit gates, as shown in Fig. 1 [4, 8, 14, 46, 73, 74, 77]. The zoned architecture has reported 1-qubit and 2-qubit gate fidelities exceeding 99.9% and 99.5%, respectively, surpassing the performance of previously known neutral atom qubit architectures [9, 20].

Despite the superior fidelity, the zoned architecture may introduce an unprecedented challenge related to zone-to-zone movements of atoms. While the pulse application time for gate execution is less than a single microsecond [20, 83], the atom's traveling between zones can require hundreds of microseconds [9, 73]. Note that the trap travel speed is limited to about 0.55 μm/μs in order not to miss atoms trapped [20]. This difference in time scales implies that movements due to the requirement to relocate qubits to another zone for different kinds of gate execution may be a major execution bottleneck of quantum programs. Our experiments with various quantum program benchmark [44, 64, 80] reveal that a naïve program execution on zoned architectures may result in an average of 78.2% (and up to 89.9%) of the total execution time being consumed by zone-to-zone movements of qubits.

We note that gate arrangements in quantum program structures commonly observed may not be optimal in zoned architectures. Applying them to zoned architectures may introduce frequent switching between single-qubit and two-qubit gate operations. For example, translating the UCCSD (Unitary coupled-cluster single and double [45]) circuits for the molecular chemical simulations [37] into representations that zoned architectures can execute requires frequently switching of the execution of 2-qubit CZ gates and single-qubit Hadamard gates. Programs such as quantum neural networks (QNNs [7, 53, 54]) or quantum approximation optimization algorithms (QAOA [21, 34]), where locally entangled gate structures based on ZZ-interactions are prevalent, also require frequent switching between 2-qubit CZ and

CGO '25, March 01–05, 2025, Las Vegas, NV, USA

ACM ISBN 979-8-4007-1275-3/25/03
https://doi.org/10.1145/3696443.3708937

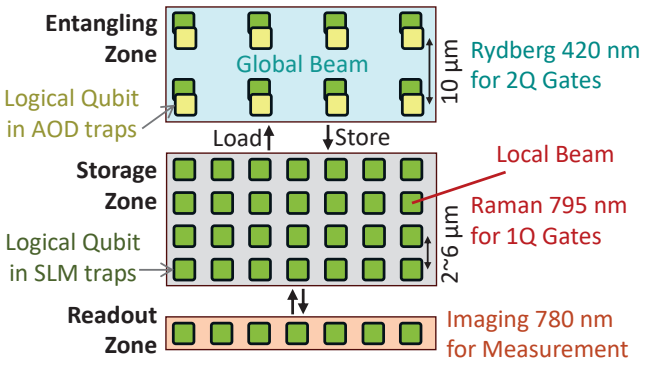

Figure 1. The zoned architecture consisting of logical qubits is divided into three zones: entangling (for 2-qubit gate executions), storage (for 1-qubit gate executions), and readout (for measurement) zones [9]. Some qubits are stored in the static optical tweezer array generated by the spatial light modulator (SLM). Some other qubits are in the movable tweezer array generated by the two-dimensional acousto-optic deflectors (AODs). `Load` means that the logical qubit trapped by the AOD moves from the storage to the entangling zone, and `Store` means from the entangling to the storage zone.

single-qubit RX or Hadamard gates [56] in zoned architecture execution. These interleavings of gates could increase inter-zone movements of atoms, resulting in longer runtimes.

Unfortunately, most previous quantum compiler studies on neutral atom quantum processors are developed for non-zoned architectures [3, 11, 47, 60, 68, 78, 79, 83, 84] or mainly minimizes intra-zone movements for the zoned architecture [73]. Thus, only leveraging existing compilation techniques may not sufficiently reduce the qubits' zone-to-zone movement overhead. By rewriting [23, 38, 42] quantum programs to mitigate frequent transitions between single-qubit and two-qubit gate execution, the inter-zone movement overhead in zoned architectures would be reduced. We introduce *Mantra* (Minimizing trAp movemeNts for aTom aRray Architectures), a quantum program generation technique designed to reduce zone-to-zone movements in zoned architectures.

Mantra reduces the number of necessary single-qubit gate operations in the storage zone; and, when the movement of qubits between zones is unavoidable, *Mantra* greedily runs gates in advance of inter-zone travel. First, *Mantra* adopts a fountain-shaped CZ-tree structure, which allows the cancellation of the single-qubit gates between CZs when translating the molecular simulation circuits into a CZ gate-based circuit. This CZ-tree structure eliminates the requirement for zoned quantum processors to send some logical qubits to the storage zone while performing CZ-tree operations. This tree structure is optimal for running the transversal gate from a trap transfer overhead perspective, as only one logical qubit is moved by AOD, and the other qubits remain in the SLM trap. In addition, *Mantra* introduces a ZZ-interaction control protocol consisting of only Rydberg-mediated gates without a single-qubit gate. This protocol eliminates the need for some logical qubits to go to the storage zone while processing serial ZZ rotational operations, such as the cost Hamiltonian in the QAOA circuits. Finally, *Mantra* reduces the number of inter-zone movements through the preemptive execution of other gates with no computational dependencies but should be run in the same zone. Fetching and aligning these gates in advance, *Mantra* reduces inter-zone travels of logical qubits.

According to the evaluations using various benchmarks [44, 64, 80], *Mantra* achieves an 88% reduction in inter-zone movements, a 35% decrease of physical gate, and a 17% improvement of fidelity compared to standard executions. Applying *Mantra* to the zoned architecture yields an average execution time that is 57% and 41% shorter than that of state-of-the-art compilers for the general-purpose neutral atom architectures [83] and zoned architectures [73], respectively.

We also confirm the benefits of the program rewriting strategies by *Mantra* in scenarios where applying Raman laser for 1-qubit gate operation is permitted directly within the entangling zone. In such cases, the atom transfer overhead between AOD and SLM traps emerges as a new execution bottleneck, as atoms do not need to move to the storage zone for each gate interleaving. Our evaluation across 7 quantum benchmarks shows that *Mantra* effectively mitigates trap transfer and AOD shuttling overhead, reducing overall execution times by up to 87% and an average of 57%.

By considering the challenges of designing compilers for zoned neutral atom processors, *Mantra* can provide guidance for higher-level quantum developers writing programs for such architectures. In doing so, *Mantra* can bridge the gap in the computing stack between high-level quantum algorithm design and the physical program execution in zoned systems. For instance, while algorithm designers often default to using CX gates, zoned architectures can leverage a variety of Rydberg-mediated gates, such as CZ, CPhase, or LP gates, which offer additional flexibility. Moreover, high-level developers might overlook the costs of trap transfers and zone-to-zone movements caused by alternating between single- and two-qubit gates. *Mantra* can mitigate these inefficiencies by exploiting the flexibility of CZ-tree chain synthesis, various Rydberg-mediated gates, and proactive scheduling of gates.

The applying coverage of methodologies by *Mantra* includes both near-term quantum applications [63] without quantum error correction (QEC [12, 75]) and fault-tolerant quantum applications considering QEC. Our proposed program rewriting methodologies could be applied to physical qubits without QEC, and they could also be easily extended to the identical principle for logical qubits consisting of several physical qubits that perform the same operations in parallel, such as Shor or Steane state correction codes [12].

The five main contributions of this paper are as follows.

- We observe that the inter-zone movements of qubits in zoned neutral atom architectures could be a major run-time bottleneck for quantum program executions.
- We also note that frequently interleaved executions between the single-qubit gate and two-qubit gate observed in various quantum program structures could increase the zone-to-zone movements of logical qubits.
- The proposed method, *Mantra*, exploits three main techniques, including an efficient CZ chain structure, a new arbitrary ZZ rotation protocol, and a preemptive gate alignment to reduce the inter-zone movements.
- *Mantra* reduces the number of cross-zone movements by 68% compared to standard program execution. In addition, *Mantra* can generate quantum programs that provide 57% and 41% less run-time compared to most recent compilers for general-purpose atom array-based architectures and the zoned architecture, respectively.
- *Mantra* mitigates the AOD-SLM trap transfer and shuttling overhead, thereby reducing execution times by up to 87% and an average of 57% even in scenarios where Raman laser is allowed in the entangling zone.

2 Background and Motivation

This section introduces a zoned neutral atom architecture where 1-qubit and 2-qubit gate execution spaces are isolated. In this architecture, programming is achieved through pulse-level controls and the movement of atoms using optical traps.

Next, we examine the implementation of SWAP operations in zoned architectures. In atom-based quantum computers, SWAP can be realized either through gate-based operations or by directly repositioning atoms. We highlight the inefficiency of gate-based SWAPs in zoned architectures due to substantial inter-zone movement overheads, underscoring the importance of prioritizing qubit moving-based SWAPs.

Finally, we analyze the execution time of various quantum benchmarks, demonstrating that inter-zone movements may dominate run-time in zoned architectures. This observation motivates the development of new compilation techniques specifically designed to optimize zone-to-zone movements, as opposed to those tailored for the non-zoned architecture.

2.1 Zoned Neutral Atom Array Architecture

Zoned architecture is designed to enhance fidelity by segregating spaces between 1-qubit and 2-qubit gate executions.

Hardware Configuration: Fig. 1 illustrates the zoned architecture, consisting of 3 zones: entanglement, storage, and readout. The entangling zone is dedicated to executing 2-qubit gates (e.g., CZ), while the storage zone is for single-qubit gate (e.g., single-qubit rotation gates) executions. The readout zone is used for measuring qubits. We consider a scenario for preparing fault-tolerant qubits using the Steane code, which is detailed in Appendix A. Qubits are stored in the static optical tweezer array generated by the spatial light modulator (SLM) [13]. The movable tweezer array generated by the 2-dimensional accousto-optic deflectors (AODs) can carry qubits between zones [20]. For the convenience of explanation, we leverages the terms of `Load` and `Store` to refer to the movement of atoms by the AOD trap between the entanglement and storage zones [73]. `Load` instruction moves the logical qubit trapped by the AOD trap from storage to entangling zone, while `Store` instruction does the opposite.

Programming by Pulse Controls and Trap Movements: The implementation of 2-qubit gates (e.g., transversal CZ) is realized by the Van der Waals' (dipole-dipole) interaction force between two neutral atoms close to each other (about 2 μm) [10]. This implementation is performing by a global pulse in the entangling zone that excites atoms to Rydberg states [9]. The implementation of single-qubit gates (e.g., single-qubit rotations) is implemented using Raman excitation through a AOD trap in the storage zone [9]. The readout process is enabled by moving qubits to the readout zone and illuminating with the focused imaging beam [20].

2.2 SWAP Implementation on Zoned Architectures

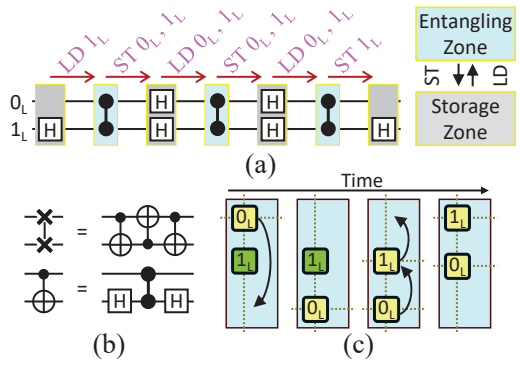

Figure 2. (a) An implementation of gate-based SWAP on the zoned architecture, where LD stands for `Load` and ST stands for `Store`. *L*s within 0_L and 1_L mean that they are logical qubits encoded into seven physical qubits by the Steane code [75], respectively. (b) The equivalent gate decompositions for explaining CZ gate-based SWAP operation in (a): a single SWAP gate could be rewritten with 3 CXs and a single CX could be rewritten with two Hadamard gates and a CZ [56]. (c) An implementation of the movement-based SWAP that complies with the movement constraints of atoms [9, 78].

SWAP operation is essential for quantum programming to exchange information between 2 qubits with each other [56]. On fixed qubit architectures, SWAP is required to perform a two-qubit gate operation between two specific qubits unconnected topologically [31]. Since neutral atom-based qubits can be movable directly, SWAP can be implemented in two ways: (i) by applying pulses that realize SWAP operation, as is done in fixed qubit architectures (gate-based),

and (ii) by exchanging positions of qubits within traps (trap-movement based) [67]. Two SWAP implementation methods have distinct advantages, and the latest quantum compilers for non-zoned architectures often employ a hybrid approach, selecting the appropriate SWAP type based on the context [78, 83]. In contrast, we observe that trap movement-based SWAPs might be superior to gate-based SWAPs for zoned architectures in all scenarios. Gate-based SWAPs in zoned architectures require six inter-zone movements (illustrated in Fig. 2 (a)), leading to unnecessary execution overhead. The detailed gate decomposition process is shown in Fig. 2 (b).

When a two-qubit entangling operation should be processed by moving qubits from the storage zone to the entangling zone, the requirements of SWAP are addressed naturally, as it is a remapping of logical qubits on the physical qubits in the storage zone. Even when a SWAP is required for qubits within the entangling zone, sequential movement adhering to the constraints of AOD trap movement [73], as shown in Fig. 2 (c), is more efficient than a gate-based SWAP.

Note that implementing a gate-based SWAP exclusively in the entangling zone is not allowed due to the requirement of 1-qubit Hadamard gates. This implies that the gate-based SWAP may offer no advantage over trap movement-based ones as long as single- and two-qubit gate executions remain segregated. Therefore, we would adopt the trap movement-based SWAP method with priority for zoned architectures.

2.3 Breakdown Analysis on Zoned Architecture

We perform the execution time breakdown analysis on the zoned architecture using various scalable quantum program benchmarks [44, 64, 80] as shown in Fig. 3. The major execution bottleneck of quantum programs on the zoned architecture is the inter-zone movements by Loads and Stores, which consumes an average of 78.2% and up to 89.9% of the total execution time. Zoned architectures pose a new bottleneck of the inter-zone movements, unlike non-zoned architectures.

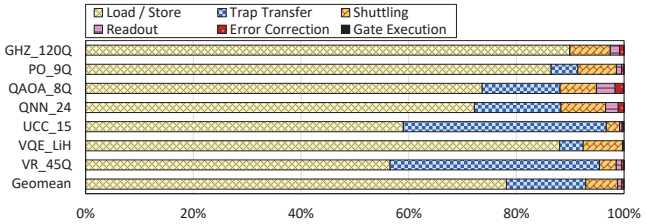

Figure 3. Execution time breakdown on zoned architectures. `Load/Store` is the time for qubits to move to another zone. `Trap Transfer` is the time to handover a qubits between a movable AOD and a static SLM trap. `Shuttling` is the time qubits spend moving in the entangling zone. `Readout` is the time required for measuring. `Error Correction` is the time spent preparing logical qubits and detecting errors. `Gate Execution` is the pulse application time for executing gates.

Note that the trap transfer and shuttling is often the main execution time overhead in non-zoned neutral atom architectures [17, 78, 79]. Therefore, various compiler techniques have been proposed for the initial mapping and routing of logical qubits to mitigate the trap transfer and shuttling overhead [3, 11, 47, 60, 68, 83, 84]. However, according to our evaluation, time for the trap transfer and shuttling account for less than 20% of the total runtime in zoned architectures. This suggests that previous compilation techniques developed for non-zoned architectures may not contribute significantly to reducing execution time for zoned architectures, which will be covered in more detail in the evaluation section. In other words, it is motivated by the need to develop compilation techniques different from the existing ones to optimize the zone-to-zone movements of qubits for zoned architectures.

3 Mantra Compilation Methodology

This section describes the compile methodology of *Mantra* to reduce the inter-zone movement in the zoned architectures. We note that the frequent switching of single-qubit and two-qubit gate executions can occur when quantum programs run naïvely on zoned neutral atom architectures. One may wonder why quantum program structures are written to require these frequent interleavings of gate execution. This may result from the high-level quantum programs (or even from the quantum algorithm) being written by adopting the CX gate as a representative two-qubit gate. Note that CX gates in quantum programs to execute on neutral atom-based processors should be translated into the representation based on the CZ and H (i.e., their native gates). Due to this translation process, 1-qubit and 2-qubit gate executions could be frequently interleaved, resulting in excessive zone-to-zone movements of atoms. The execution efficiency of the zoned architectures could be improved by rewriting programs so that they require fewer zone-to-zone movements of qubits.

Mantra minimizes 1-qubit gate operations required for the storage zone execution. *Mantra* adjusts program scheduling to preemptively run some gates that could be executed in the same zone when the inter-zone movement is inevitable.

3.1 Fountain-Shaped CZ-Tree Chain

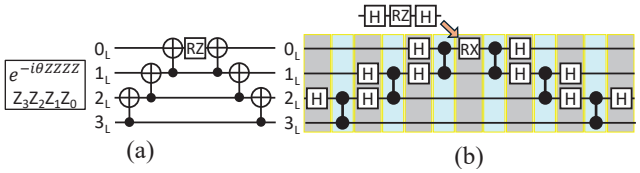

Figure 4. (a) An example Hamiltonian simulation kernel [45] and its circuit implementation with standard chain structure, and (b) CZ Gate-Based Implementation of the circuit of (a).

This section discusses the efficiently executed CZ-tree chain structure on the zoned neutral atom architectures, where

the chain structure is required for Hamiltonian (or quantum) simulation kernels [45] such as UCCSD (Unitary Coupled-Cluster Single and Double [5, 72]) circuits. Fig. 4 (a) shows an example Pauli string and its corresponding quantum circuit, where the CX gates are generally constructed in a path-shaped standard chain structure, where the control qubit of the next CX becomes the target qubit of the previous CX. Unfortunately, as shown in Fig. 4 (b), the CZ gate-based circuit translated from Fig. 4 (a) may raise frequent switchings between the execution of single-qubit and 2-qubit gates. Since the circuit in Fig. 4 (b) requires single-qubit Hadamard (H [56]) or rotation-X (RX [56]) gate(s) between each CZ gate execution, inter-zone movement is required for each gate execution step. For example, a n-qubit single Pauli-string circuit could require at least $2(n-1)$ times of interleavings for the different zone execution in the standard chain structure.

The (path-shaped) standard chain structure of executed in the zoned architecture may result in frequent inter-zone movement. To address this, *Mantra* adopts a fountain-shaped chain structure, as shown in Fig. 5 (a). As shown in Fig. 5 (b), the target qubit of all CX gates in the proposed fountain-shaped chain structure is 0_L in common. Note that two consecutive Hadamard gates become the identity operation since the Hadamard operator is a Hermitian matrix [58]. As described in Fig. 5 (c), the CZ trees are only processed within the entangling zone without having to travel to the storage zone since Hadamard gates between CZs are canceled. Therefore, the fountain-shaped chain structure could reduce inter-zone movements required for the CZ-tree executions.

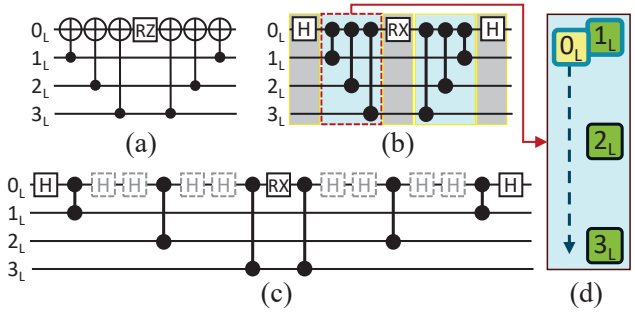

Figure 5. (a) Implementation of a circuit with fountain-shaped chain structure for the simulation kernel in Fig. 4, (b) CZ-based Implementation of the circuit of (a), (c) Describing the gate cancellation process of H gates in the fountain CZ chain, and (d) A possible AOD trap movement for CZ chain.

This fountain-shaped CZ chain structure not only reduces the number of zone-to-zone movements for logical qubits but is also efficient from the perspective of atom shuttling within the entangling zone. 0_L qubit participates in a computation for all CZ in the tree operations. Therefore, transversal CZ operations could be achieved by only moving the logical qubit of 0_L by the AOD trap, as described in Fig. 5 (d). The standard chain structure described in Fig. 4 requires frequent trap transfers between AOD and SLMs since they should be moved to the storage zone immediately after CZ operation. In contrast, the fountain-shaped chain structure does not require any AOD-SLMs trap transfer during a CZ-tree operation process. Therefore, the fountain-shaped CZ chain structures are expected to be relatively more efficient than standard chain structures, not only in zoned but also in non-zoned architectures. The cancellation of Hadamard gates in the CZ-tree reduces laser pulse application time slightly and thereby reduces fidelity degradation due to the single-qubit gate executions, and the trap transfers could also be reduced.

Algorithm 1 Fountain-Shaped CZ-Tree Implementation

```
function CZ_FOUNTAIN(pauli: Pauli)
    chain ← QC(pauli.num_qubits)
    control, target ← None, None
    chain.h(target) //Applying Hadmamrd Gates
    for i ← 0 to pauli.num_qubits do
        pauli_i ← pauli.to_label[i]
        if pauli_i ≠ 'I' then
            if target = None then
                target ← i
            else
                control ← i
            end if
        end if
        if control ≠ None and target ≠ None then
            chain.cz(control, target) //Applying Controlled-Z Gates
            control ← None
        end if
    end for
    chain.h(target)
    return chain
end function
```

The fountain-shaped CZ chain structure shown in Fig. 5 (b) can be implemented as described in Algorithm 1. Note that *Mantra* is not the first to propose a fountain-shaped chain structure. The flexibility of CX (or CZ) chain synthesis in Hamiltonian simulation kernels is well known [37, 45], and many commercial quantum compilers (such as IBM's Qiskit SDK [31]) can provide various chain structures. There are many recent studies for efficient placement of 2-qubit gate trees in sparsity topologies such as superconducting-based qubits [36, 50–52]. Furthermore, a method for synthesizing efficient fermionic system simulation circuits on non-zoned neutral atomic array processors has recently been proposed [24]. Unfortunately, we note that these recent studies cannot adequately reduce the inter-zone movement overhead, as they are not developed for zoned architectures. In the implementation presented in Algorithm 1, the order of application of CZ is in ascending order of the qubit numbering due to the movement efficiency of AOD traps, as opposed to the descending-order provided by `Lie-Trotter` in Qiskit SDK.

We estimate the number of the LD and ST instructions according to the standard (path-shaped) and fountain-shaped

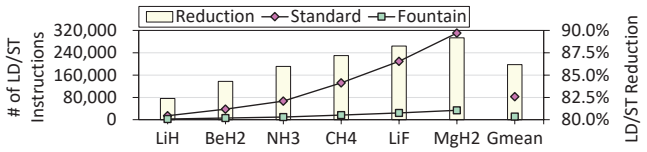

Figure 6. Comparative analysis of the number of Load/Store instructions by two chain structures (Standard VS. Fountain) in VQE (Variational quantum eigensolver) circuits of the UCCSD (Unitary coupled-cluster single and double) Ansatz. In this experiment, all molecular simulation circuits were subjected to the gate cancellation by Qiskit Transpiler [31].

chain structures for chemical simulation circuits encoded by the Jordan-Wigner transformation [39], as described in Fig. 6. In the standard chain structure, the number of LD and ST instructions is proportional to four times the number of CZs in the simulation circuit, corresponding to complexity equivalent to $O(n^4)$ when the number of qubits is n. On the other hand, the fountain-shaped chain structure requires two LDs and two STs for each Pauli-string (regardless of the number of qubits), resulting in lower complexity. As a result, the LD and ST instruction counts in the fountain chain structure are more advantageously scaled than those in the standard chain. As shown in Fig. 6, the fountain chain structure requires 83% fewer LD or ST instructions, averaging in six molecular simulation circuits over the standard chain.

3.2 1Q-Gateless Arbitrary ZZ-Interaction Protocol

A ZZ-interaction is another prime operation for occurring interleaved executions of single- and two-qubit gates in the CZ gate-based decomposition template for zoned neutral atom architecture. In this section, we analyze the execution efficiency of the zoned architecture in scenarios where locally entangled structures with arbitrary-angular ZZ interactions are prevalent across all logical qubit pairs, as shown in the Fig. 7 (a). These locally entangled structures are often observed on the QAOA cost Hamiltonians for dense graphs [21] or ansätze for variational approach-based QNNs [2, 64].

The circuit in Fig. 7 (a) is translated as Fig. 7 (b) based on the CZ gate to run on the zoned architecture. Unfortunately, quantum program structures such as Fig. 7 (b) may cause excessive inter-zone movements of logical qubits due to frequent switching of 1-qubit and 2-qubit gate executions. Moreover, in this program structure, it is challenging to apply the flexibility of the CZ synthesis [45] discussed in Section 3.1. To address this challenge, *Mantra* proposes a ZZ rotation gate protocol that does not require a single-qubit gate by leveraging two Rydberg-mediated gates capable of performing by neutral atoms. The unitary operator for the arbitrary rotating ZZ gate to be implemented is as follows:

$$RZZ(\gamma) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{i\gamma} & 0 & 0 \\ 0 & 0 & e^{i\gamma} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

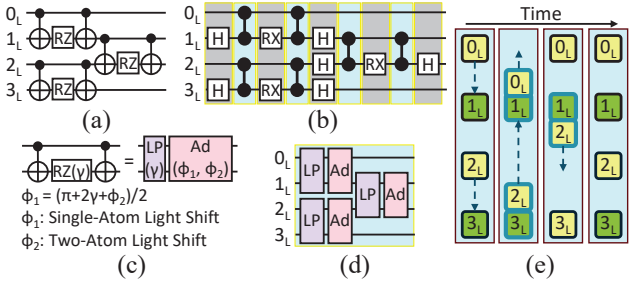

Figure 7. (a) An example of the simulation circuit consisting of local entanglements with ZZ rotations, (b) CZ-based Implementation of the circuit of (a), (c) The proposed arbitrary ZZ rotation protocol consisting of a single adiabatic (Ad) [40] and a single Levine-Pichler (LP) gate [43], (d) Implement the circuit in (a) by applying the proposed protocol, and (e) One possible qubit movement for circuit (d), where the entanglements could be implemented without both the AOD-SLM trap transfer and trip of logical qubits (to the storage zone).

where γ is a commonly used for cost Hamiltonians in QAOA.

The essential principle of the proposed gate protocol is to achieve the arbitrary angle ZZ-rotation by combining Levine-Pichler gates and CPhase (controlled phase) gates for offsetting phase shift for the $|11\rangle$ state. Hamiltonian for the optimal control of Rydberg atom-based qubits is described in detail in Appendix B. There are various versions of entangling gates for realizing controlled phase shift [20], but in this section, we would describe them by adopting the adiabatic gate [40]. Another version that implements the proposed gate protocol with the time-optimal `CPHASE` gate [20] instead of the adiabatic gate is detailed in Appendix C.

The unitary operators implementing the adiabatic and the Levine-Pichler gate can be written as follows, respectively:

$$Ad(\phi_1, \phi_2) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{\phi_2 - 2\phi_1} \end{bmatrix} \text{ and } LP(\gamma) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{i\gamma} & 0 & 0 \\ 0 & 0 & e^{i\gamma} & 0 \\ 0 & 0 & 0 & e^{2\gamma+\pi} \end{bmatrix}$$

where ϕ_1 could be selected by the single-atom light shift (which is induced by the global Rydberg laser [55]), ϕ_2 could be selected by the two-atom light shift [40], and γ is an angle for ZZ rotation. By applying an adiabatic gate and a Levine-Pichler gate, where a single-atom and two-atom light shifts in the adiabatic gate are chosen so that $\phi_1 = (\pi + 2\gamma + \phi_2)/2$, the ZZ rotation could be achieved without any single-qubit gate, as shown in Fig. 7 (c). It is attributed to the phase shift cancellation to the fourth amplitude ($|11\rangle$) of the Levine-Pichler gate by controlled Z rotation from the adiabatic gate.

The proposed protocol does not require qubits to be sent to the storage zone while executing the ZZ-rotation operations, as it can be performed in the entangling zone only, as shown in Fig. 7 (d). It differs from the program structure of Fig. 7 (b) that requires execution interleavings between different

zones for every gate-layer. One possible shuttling process without trap transfers can be realized as shown in Fig. 7 (e).

3.3 Preemptive Identical-Zoned Gate Alignment

This section discusses a scheme to reduce the inter-zone movement overhead by adjusting the execution timing of some gates. We utilize examples of GHZ-state preparation [19]. The fundamental principle of the proposed gate alignment methodology can be described as follows: First, find initial gate(s) in the program and identify its execution zone. Next, among the subsequent gates in the program, the gates that have no computational dependency but can be processed together in the first zone execution step are identified. These identified gates are preemptively aligned in the first zone execution step. Then, identify gates that need to be processed in the next zone execution step and apply the identical rules. This process is repeated until the final zone execution steps.

Similar to the case of Hamiltonian simulation circuits as discussed in Section 3.1 [15, 37], there is a configuration flexibility of the CX (or CZ)-tree chain when preparing the GHZ state. Connecting 2-qubit gates as the fountain shape may be more efficient rather than connecting CZs as the path shape. The fountain-shaped CZ structure can reduce trap transfers and zone-to-zone travels as discussed in Fig. 5.

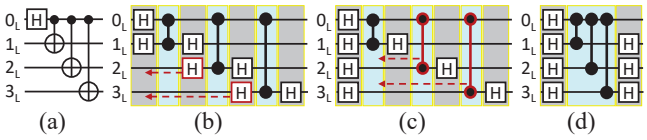

Figure 8. (a) An example of a 4-qubit GHZ-state circuit consisting of the fountain-shaped CX chain, (b) CZ-based Implementation of the circuit of (a), where H gates first applied to the logical qubits 2_L and 3_L can be executed more preemptively before the first CZ. (c) The second and third CZs can be applied preemptively prior to the second storage zone execution, and (d) A circuit with the alignment applied.

Fig. 8 (a) shows an example of a four-qubit GHZ state preparation circuit consisting of a fountain-shaped CX chain. Note that their CXs' control and target qubits are opposite to the simulation circuits discussed in Fig. 5 when preparing GHZ states with fountain shape-based chains; the control qubit of CXs is concentrated in one qubit. Section 3.1 adopts a fountain-shaped CX-tree structure, exploiting the opportunity to cancel single-qubit Hadamard gates by sharing the target qubits of multiple CXs. Unfortunately, in this scenario, the Hadamard gates would be generated from different qubits in the process of translating to CZ-based circuits, and hence, it is challenging to cancel them from each other. Instead, in this scenario, we may consider an approach to adjust the execution schedule of some gates without computational dependence to reduce zone-to-zone movements of the qubit.

Fig. 8 (b) and (c) describe a preemptive gate alignment method to reduce the inter-zone travel overhead using an example of a 4-qubit fountain-shaped GHZ state circuit. As shown in Fig. 8 (b), Hadamard gates initially applied to each of 2_L and 3_L can be preemptively executed in the first storage zone execution step. Then, as shown in Fig. 8 (c), the CZ gates associated with 2_L and 3_L can also be preemptively executed in the first entangling zone execution step. Applying this realignment provides circuits such as Fig. 8 (d), and it enables the preparation of GHZ status with only two inter-zone movements, regardless of the number of qubits. If X-basis initialization and measurement are supported on the device, applying requirements for Hadamard gates are eliminated, so the GHZ state preparation could be performed without having to go to the storage zone during program execution.

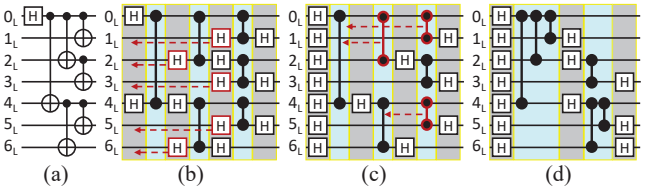

Figure 9. (a) An example of a 7-qubit GHZ-state consisting of parallel CX chain [16], (b) CZ-based Implementation of the circuit of (a), where some H gates could be applied preemptively in the first step of the storage zone execution. (c) Some CZs could be applied in the entangling zone execution steps further ahead of their current location in the circuit, and (d) The circuit applied the preemptive gate alignment.

Furthermore, we illustrate an example of reducing inter-zone travel overhead by using the proposed gate alignment method for the *parallel* CX structure in which the GHZ-state circuit depth expands logarithmically as the number of qubits increases, as shown in Fig. 9 [16]. Fig. 9 (a) shows an example of a 7-qubit GHZ state circuit consisting of the parallel CX chain. Hadamard gates firstly applied to the logical qubits 1_L, 2_L, 3_L, 5_L, and 6_L could be preemptively executed in the first storage zone execution step, as shown in Fig. 9 (b). Then, as shown in Fig. 9 (c), some CZs can also be preemptively executed in the earlier entangling zone execution step. Applying this realignment provides circuits such as Fig. 9 (d). In this case, the number of travels between the entangling zone and storage zone is reduced from 6 to 4.

Comparison with Existing Compilation Techniques: Gate scheduling has been adopted for various purposes by many quantum program compilers [35, 71, 73, 85], although to the best of our knowledge, there is no efficient one to reduce the inter-zone travel overhead in zoned architectures. For example, `ASAPSchedule` and `ALAPSchedule` in Qiskit SDK [35] aims to reduce idling time by scheduling gates at the earliest or late possible. Unfortunately, these schemes

cannot be used to reduce atom movements between zones as they do not separate gates by 1-qubit and 2-qubit categories.

3.4 Chain Execution Efficiency by Qubit Array Sizes

This section analyzes the efficient CZ chain structure according to various sizes and topologies of logical qubit arrays in the zoned architecture, using GHZ state preparation as an example. Section 3.3 described two CZ chain examples for GHZ state preparation in zoned architectures: fountain-shaped and parallel structures. The parallel chains may seem more favorable for execution, as their circuit depth scales logarithmically with increasing qubits, while the fountain-shaped chain's one scales linearly. However, regarding the number of inter-zone movements, the parallel chain requires logarithmically expanding LD or ST instructions as the number of qubits grows. In contrast, fountain-shaped chains require only a single LD and ST each regardless of the qubit counts.

Although the distance between the entangling and storage zone is only 20 μm [9], logical qubits (needed to go another zone) travel much further than that for the following reasons: (i) Each atom that makes up the qubit is separated from one another with sufficient distances on a two-dimensional lattice to avoid unintended interactions. Thus, as the qubit scale in a quantum program increases, the average travel distance of logical qubits for moving to different zones also grows. (ii) Note that atoms in AOD traps cannot cross each other [73, 78, 79, 83]. This constraint on their movement can require some logical qubits to detour from other qubits or transfer others to SLM traps when they move to other zones.

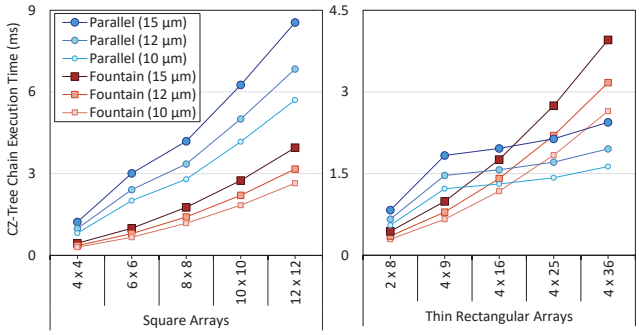

Figure 10. A comparative analysis of the execution time of two CZ-tree chain structures (Parallel VS. Fountain) for preparing GHZ states by various 2-dimentional atom array configurations (square and thin rectangular-shaped) and inter-atom distances (10, 12, and 15 μm) in entangling zone.

Fig. 10 compares the execution times of parallel-shaped and fountain-shaped chains for preparing GHZ states across various array sizes, shapes, and inter-atom distances. For square-shaped arrays, the fountain-shaped chain achieves shorter execution times on average compared to the parallel chain for the same number of qubits over the tested range of inter-atom distances and array sizes. Although parallel GHZ chains [16] are theoretically expected to be more efficient over fountain chains thanks to their logarithmic depth scalability when hardware topology is not considered, the results in Fig. 10 suggest this is not always true in practical hardware execution. The difference in running time of the two chain structures is due to two reasons as follows: (i) The application relationships and sequences of CZ gates in parallel chains are more complex than those in fountain-shaped chains, leading to linear scalability of atom travel time within the entangling zone as the number of qubits increases. (ii) while the number of LD/ST operations in parallel chains scales logarithmically to the number of qubits, the fountain-shaped chain requires a constant number of LD/ST operations regardless of the number of qubits. As a result, the fountain-shaped chain shows better run-time scalability than the parallel chain in the 2-dimensional square arrays.

For thin rectangular-shaped arrays, which keep the number of the row of logical qubits constant, the average traveling time between zones would be kept almost constant. In this case, the parallel CZ chain structure provides better time scalability with an increasing number of qubits compared to the fountain-shaped one, and at some point (approximately 80 physical qubits according to our calculation), the parallel chain would provide less execution time than the fountain-shaped chain. Though the parallel CZ chain structure is more advantageous over the fountain-shaped chain in terms of execution time complexity, the practical CZ chain structure depends on qubit topologies (i.e., the arrangement of atoms).

Discussion: Fig. 10 suggest that minimizing inter-zone travel distances of qubits through constraining the vertical size of zones (a thin rectangular array) could enhance execution efficiency, particularly for programs with high gate parallelism. Nevertheless, the thin rectangular array may limit the reconfigurability within zones. Instead, by expanding the contacted area between the entangling zone and storage zone, hardware developers can consider a design that, on average, allows large amounts of atom movement with short distances while maintaining reconfigurability within the zone. For instance, a 3-dimensional architecture that stacks entangling and storage zones while preserving their size could be considered. This may ensure reconfigurability on 2-dimensional intra-zone movements and simultaneously allows efficient inter-zone movement in a direction perpendicular to them. However, potential thermal issues due to increased contact area between zones should be considered.

4 Evaluation Methodology

We build the in-house quantum program execution modeling for our evaluations, as there are currently no commercially available zoned neutral atom processor and no simulators for zoned architectures. Hardware parameters for modeling the zoned neutral atom architecture are described in Table 1.

Table 1. Hardware Parameters for Neutral Atom Array

Pulse Time of 1Q Gate	625 ns (BB1 pulse, except RZ) [10]
Pulse Time of 2Q Gate	380 ns (global Rydberg pulse) [10]
Readout Imaging Time	500 μs [9]
Fidelity of 1Q Gate	0.999 (Raman laser pulse [20])
Fidelity of 2Q Gate	0.995 (time-optimal CZ [20])
Fidelity of Readout	0.998 (pushout and imaging [20])
Trap Transfer Time	150 μs [9]
Fidelity of Transfer	0.999 [9]
Atom Array Size	41 x 41 (for each zone)
# of Physical Qubits	1,681 qubits
# of Logical Qubits	~120 qubits (using EDFT [9])
AOD Trap Speed	0.55 μm/μs [10]
Shuttling (Unit Dist.)	21.8 μs
Coherence Time	100 s (in Storage zone) [9], 4 s (out of Storage zone) [9]
Logical Basis	X-basis and Z-basis [9]
Atom Unit Distance	12 μm (E. Zone), 6 μm (S. Zone) [73]
LD/ST Time	36.4 μs (Min.), 527.3 μs (Median)
Fidelity Drop by X-talk	0.005 (1Q Gate) [43], 0.007 (CZ) [43]

4.1 Modeling Zoned Rydberg Array Architecture

We assume a square-shaped array of atoms of 41×41 (1,681 physical qubits). Although this array is larger than the size performed in the zoned architecture research [9], it is still less than the recently performed tweezer array of 2,088 physical qubits [62]. The basic layout of the array of zoned architecture follows the original study: each zone is 20 micrometers apart from each other [9]. Each physical qubit is arranged in a two-dimensional lattice shape, 12 micrometers in the entangling zone and 6 micrometers in the storage zone. The preparation of fault-tolerant qubits adopts the EDFT (post-selecting on flags and error detection) scheme [9], whereby 14 physical qubits constitute a single logical qubit. Therefore, we could prepare 120 logical qubits under the assumption that 1,680 or more physical qubits are successfully trapped.

4.2 Gate Operation Policies

Three gate execution policies are considered for evaluations.

Type 1 (*default*): The execution of single-qubit and two-qubit gates is completely isolated according to the zone. Qubits in the storage zone should be moved to the entangling zone when 2-qubit gate execution is required, and vice versa.

Type 2: We consider a scenario in which each gate interleaving does not require inter-zone movement. In this type, all qubits should start from the storage zone, but applying a local Raman laser (for the single-qubit gate execution) to the entangling zone is partially permitted. Atoms subject to a single-qubit gate pulse should be sufficiently (more than 12 μm) apart from all the other atoms, which requires to the shuttling overhead within the entangling zone. It is assumed that there is no fidelity drop due to the cross (X)-talk error when applying other kinds of pulses, thanks for the detuning techniques for protecting quantum states of qubits [48, 59].

Type 3 (non-zoned architecture): It is assumed that both single-qubit and two-qubit gates can be executed in place on the single zone [43]. In this case, the atoms do not have to go out of the zone before the measurement. Due to the in-place execution, we assume that atoms experience a fidelity drop by other kinds of pulse application, as shown in the Table 1.

4.3 Execution Time Estimation

The following events are collected for run-time estimation.

Load/Store: It refers to the time for logical qubits to travel between entangling and storage zones. Load and Store time includes for logical qubits to depart from their initial location and move to a specific location in another zone. The AOD trap speed is limited to 0.55 μm/μs [10]. Therefore, the minimum Load or Store time is 36.4 μs, assuming that the distance between the entangling and storage zone is 20 μm.

Trap Transfer: This means the time required for the handover of atoms between AOD traps and SLM traps. Some logical qubits require a transfer between AOD and SLM traps due to the implementation of transversal entanglement gates or the movement constraints of AOD traps [78, 79]. The trap transfer time of approximately 150 to 300 μs is required [9].

AOD Shuttling and SWAP: Note that AOD traps cannot cross each other and have several movement constraints [9, 73, 78]. The AOD trap movement policy for the baseline follows OLSQ-DPQA [78]. Assuming that the distance between each atom within the entangling zone is 12 μm and placed in a 2-dimensional square-shaped lattice structure, a qubit trapped in AOD requires about 21.8 μs to travel one edge of the unit lattice. For zoned architectures, we modify the SWAP operation to always adopt a trap movement-based rather than a gate-based one. This is because trap-movement-based SWAP is almost always more efficient than gate-based SWAP in zoned architectures, as discussed in the Section 2.2.

Readout (Measurement): An imaging time of approximately 0.5 ms is required to measure the state of qubits [9].

Error Correction: This refers to the amount time it takes to perform the EDFT method [9], where it includes times for preparing logical qubits in $|0\rangle$ or $|+\rangle$ state through 7-qubit Steane code [75], applying parallel transversal entangling gates, and traveling ancilla logical qubits to the storage zone.

Gate Exeution Time: The single-qubit pulse application time is assumed to be 625 nanoseconds [83]. This time is applied to single-qubit gates rotating arbitrary axes except for Z rotations. The Z-rotation could be virtually implemented by the control software without consuming time [9]. The 2-qubit pulse time is assumed to be 380 nanoseconds [83].

4.4 High-Throughput Trap Movement Strategies

Two strategies are applied to achieve high-throughput execution in zoned architectures. They are implemented in our emulated modeling and are reflected in all runtime scenarios.

Transferring Traps in Advance: The trap transfer time takes far longer than the gate pulse time. We can hide the

Table 2. Quantitative evaluations of Mantra over standard program execution on various scalable benchmarks.

Workloads	Logical Qubits	# of LD/STs (X-basis calculation is allowed.)			# of Physical Gates			Total Circuit Fidelity		
		Standard	Mantra	Reduced	Standard	Mantra	Reduced	Standard	Mantra	Improved
GHZ	40Q	78	10 (8)	87.2% (89.7%)	826	826	0.0%	0.75	0.76	1.5%
	80Q	158	2 (0)	98.7% (100.0%)	1,666	1,666	0.0%	0.56	0.57	2.6%
	120Q	238	2 (0)	99.2% (100.0%)	2,506	2,506	0.0%	0.42	0.43	3.1%
PO	3Q	36	6	83.3%	322	259	19.6%	0.89	0.89	0.9%
	6Q	180	6	96.7%	1,596	1,281	19.7%	0.55	0.58	4.6%
	9Q	432	6	98.6%	3,808	3,052	19.8%	0.24	0.27	11.4%
QNN	8Q	46	16	65.2%	959	567	40.9%	0.68	0.71	5.8%
	16Q	94	32	66.0%	3,731	2,051	45.0%	0.21	0.27	28.6%
	24Q	142	48	66.2%	8,295	4,431	46.6%	0.03	0.11	73.7%
UCC	5Q	160	40	75.0%	1,680	707	57.9%	0.57	0.65	14.9%
	10Q	360	40	88.9%	3,780	1,407	62.8%	0.28	0.40	40.0%
	15Q	560	40	92.9%	5,880	2,107	64.2%	0.14	0.24	71.5%
VR	15Q	84	84	0.0%	910	588	35.4%	0.74	0.78	4.7%
	30Q	174	174	0.0%	1,855	1,218	34.3%	0.54	0.59	9.7%
	45Q	264	264	0.0%	2,800	1,848	34.0%	0.39	0.45	14.6%
Geometric Mean		153.6	20.6 (18.5)	86.6% (87.9%)	2,010.1	1,297.8	35.4%	0.37	0.43	17.1%

latency by performing trap transfers while logical qubits travel between zones. Suppose one of the two logical qubits scheduled for a 2-qubit transversal operation is being loaded from storage zone, or it needs to go to the storage zone and come back. In that case, we transfer the remaining qubits waiting in the entangling zone to the SLM trap in advance.

Placing Qubits near Zone Borders: To minimize inter-zone travel time (LD/ST), physical qubits are arranged along the borders nearest to adjacent zones. Since the number of physical qubits is known before program execution, all qubits could be pre-placed as close to the zone borders as possible.

4.5 Fidelity Estimation

Each qubit's fidelity (F) can be estimated as follows: $F = F_{1Q}^{N_{1Q}} \times F_{2Q}^{N_{2Q}} \times F_{\text{Transfer}}^{N_{\text{Transfer}}} \times F_{\text{Readout}} \times F_{\text{Decoherence}}$, where F_{1Q} is the fidelity of a 1-qubit gate, N_{1Q} is the number of applied 1-qubit gates, F_{2Q} is the fidelity of a 2-qubit gate, N_{2Q} is the number of applied 2-qubit gates, F_{Transfer} is the fidelity of a trap transfer, N_{Transfer} is the number of applied trap transfers, F_{Readout} is the readout imaging fidelity, and $F_{\text{Decoherence}}$ is the fidelity by decoherence. The gate fidelity itself is equivalent to each other in both non-zoned and zoned architectures since we consider homogeneous Rydberg atoms. However, crosstalk errors by other types of gate pulses are considered as a fidelity drop term in the case of non-zoned architectures. Decoherence follows a model that relaxes exponentially, where the time constant depends on whether each individual qubit exists in or out of storage zone: $F_{\text{Decoherence}} = e^{-(\frac{T_{\text{In}}}{100} + \frac{T_{\text{Out}}}{4})}$. T_{In} is the amount of time that the atom exists in the storage zone during the program execution, and T_{Out} is the amount of time it exists out of the storage zone. Total fidelity can be obtained by multiplying all fidelities from individual qubits.

4.6 Quantum Program Benchmarks

We utilize five scalable quantum circuits obtained from the MQT Bench [64], SupermarQ [80] and QASMBench [44]: `GHZ` (Greenberger–Horne–Zeilinger State Preparation), `PO` (Portfolio Optimization with QAOA), `QNN` (Quantum Neural Network), `UCC` (Unitary Coupled-Cluster Singles and Doubles ansatz randomly sampled), and `VR` (Vehicle Routing). For n-qubit `UCC` circuits, it consists of 10 Pauli strings, where the n matrices of each Pauli string are randomly sampled with the same probability among Pauli-I, X, Y, and Z. Furthermore, we utilize six molecular simulation circuits for 12-qubit `LiH`, 14-qubit `BeH2`, 16-qubit `NH3`, 18-qubit `CH4`, 20-qubit `LiF`, and 22-qubit `MgH2`, where the Pauli strings of each molecule can be obtained from PySCF [76] and the circuit synthesis of the Jordan-Wigner transformation-based VQE circuits utilizes the Qiskit SDK [35]. We also utilize Power Law [30] and Sherrington-Kirkpatrick (SK) [70] model graph-based QAOA for Max-Cut circuits with varying qubit and layer counts. The gate cancellation is applied to all quantum benchmark programs by Qiskit Transpiler (optimization level = 3) [35].

We consider execution per Pauli-string since we are interested in the scale of workloads that can be processed with fidelity that is meaningful to current zoned architectures, although estimating execution times for complete Pauli-strings is not challenging. For example, the VQE circuit simulating `LiH` requires 36.0 minutes by NALAC and 11.6 minutes by *Mantra*, which exceed the Rydberg atom's coherence time.

5 Results and Analysis

5.1 Evaluations on Scalable Benchmarks

As shown in Table 2, *Mantra* reduces the number of LDs or STs, decreases physical gates, and improves circuit fidelity on average, compared to the standard execution. `GHZ` 40Q differs in LD or ST counts from 80Q and 120Q since their CZ

chain structures are different from each other, as discussed in Section 3.4. GHZ40Q has relatively low qubit requirements, allowing *Mantra* to place logical qubits as thin rectangular-shaped arrays along the boundaries of other zones. The parallel chain structure is more advantageous in this case. For GHZ 80Q and 120Q, the fountain-shaped chain is more advantageous than the parallel ones due to further travel distance between zones. Due to the proposed gate scheduling, all CZs in the GHZ 80Q and 120Q are processed in a single entangling zone execution step, requiring only two zone-to-zone movements. If X-basis initialization and measurement are permitted [9], H gates in GHZ circuits are unnecessary. In this case, GHZ 80Q and 120Q can be performed without LD/ST.

PO workloads require the number of LDs and STs of complexity proportional to the square of the number of qubits for standard execution (due to gate counts in the cost Hamiltonian), but *Mantra* only requires 1 LD and 1 ST per QAOA layer, regardless of the number of qubits. For workloads with deep circuit depth, such as QNNs and UCCs, the fidelity improvement by *Mantra* is greater than for other workloads. This is attributed to a reduction in the overall run-time by reducing LD and STs, resulting in less fidelity drop by the decoherence. *Mantra* does not always reduce LD/STs or gates. For GHZ workloads where only the gate scheduling is applicable, the number of LD/STs is reduced, but that of the gates is not. Since VR consists of fixed path-shaped CX chains with no synthetic flexibility, the number of LD/STs does not decrease.

5.2 Evaluations on Molecular Simulation Workloads

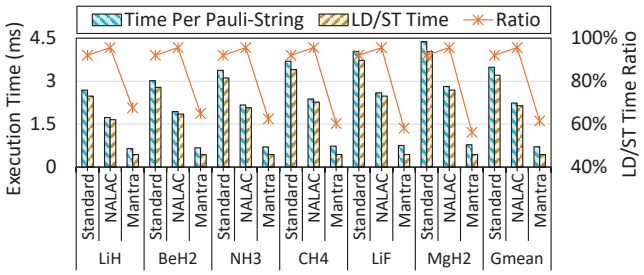

Figure 11. Comparative analysis of average execution time per single Pauli-string of six VQE circuits using Jordan-Wigner encoder according to quantum program compilers.

Fig. 11 illustrates the total execution and LD or ST time per a single Pauli-string for VQE circuits according to the 3 execution scenarios: Standard approach, NALAC, and *Mantra*.

As shown in Fig. 11, both NALAC and *Mantra* reduce execution time over the standard execution across all VQE circuits. NALAC reduces execution time per Pauli string by 36% and the LD/ST time ratio by 33% on average, over the standard execution. *Mantra* reduces execution time per Pauli-string by 79% and the LD/ST time ratio by 86% on average, over the standard execution. The performance improvement by *Mantra* over NALAC can be attributed to its reduction of LD and ST operations, which are major bottlenecks in zoned architecture execution. This is also revealed by the ratio of LD/ST times to the entire program run time, which averages 96% for NALAC, slightly higher than the standard execution ratio of 92%, while *Mantra* achieves a lower average of 61%. As the number of qubits for the molecular simulations increases, the execution reduction ratio of *Mantra* against the standard program execution gradually increases because *Mantra* requires a constant number of LD or ST instructions per single Pauli-string, regardless of the molecule workload.

5.3 Evaluations on QAOA Workloads

Fig. 12 illustrates the total run-time of QAOA circuits for PL and SK models. In both compilers and target graphs, the execution time increases linearly with the number of QAOA layers. For PL graphs, the QAOA circuit depth scales linearly with the number of qubits. For SK models, the number of RZZs grows quadratically with the number of qubits. Although the depth complexity for SK models is linear due to the fermionic SWAP network [28], the run-time increases nearly quadratically with the number of qubits due to the shuttling and trap transfer overheads in the entangling zone.

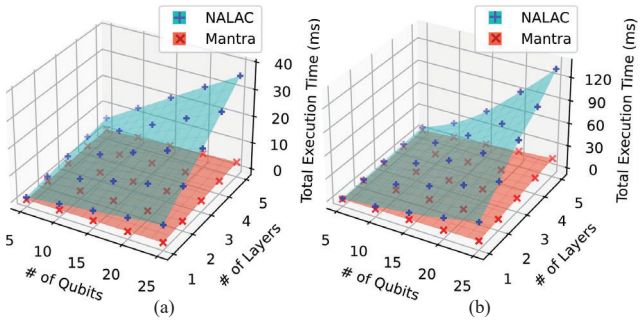

Figure 12. A comparative analysis of the total execution time of QAOA (Quantum approximate optimization algorithm) circuits for the (a) Power Law models [30] and (b) Sherrington-Kirkpatrick model [70] as a target graph according to the number of qubits and layers: NALAC VS. *Mantra*

In NALAC, LDs or STs are required for the cost Hamiltonian, which scales with the number of RZZ operations. However, *Mantra* employs the new gate protocol, which eliminates the need to access the storage zone during processing of the cost Hamiltonian. For PL models, the total execution time of *Mantra* relative to NALAC decreases from 62% to 89% (average 79%) in a given qubit range. For SK models, the total execution time of *Mantra* relative to NALAC decreases from 75% to 91% (average 86%) in a given qubit range. As the number of qubits and layers increases, *Mantra* could further reduce the QAOA execution time over NALAC. These results indicate that a major bottleneck in running QAOA circuits in zoned architectures may arise from inter-zoned movement

overhead arising from processing the cost Hamiltonian, and *Mantra* could efficiently reduce the inter-zoned movements.

5.4 1Q Gate Execution Allowed in Entangling Zone

One possible implementation of zoned architectures can permit 1-qubit gate execution within the entangling zones. Even in this case, atoms in the entangling zone would still need to be re-positioned to apply a local Raman beam realizing 1-qubit gates. In this scenario, while atoms requiring single-qubit gates no longer need to travel to the storage zone for each operation, the interleaving of single-qubit and two-qubit gate operations would still hinder program execution due to intra-zone atom movements. *Mantra* can improve execution efficiency by effectively mitigating trap transfer and shuttling overheads, even under this operation policy.

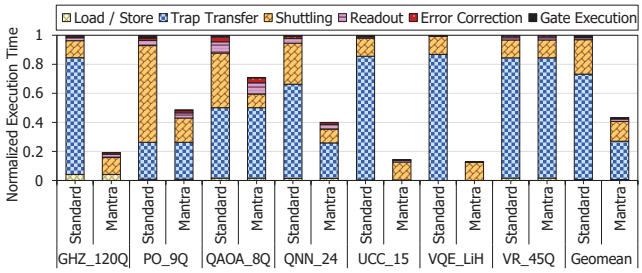

Figure 13. Normalized execution time comparison for various workloads by Standard and *Mantra* when 1Q gates are allowed in the entangling zone (i.e., Type 2 operation policy).

This section evaluates the performance achieved by *Mantra* under the Type 2 policy, where the Raman laser is partially allowed in the entangling zone. Fig. 13 illustrates the normalized run-times for various workloads under this policy, comparing the Standard and *Mantra* compilation strategies. Allowing 1-qubit gates in the entangling zone suppresses the `Load/Store` overhead remarkably, as atoms no longer need to move between the storage and entangling zones for each gate interleaving. While `Load/Store` are the primary bottlenecks under the Type 1 policy, they account for only 0.7% of the total run-time in this case. Instead, `Trap Transfer` and `Shuttling` emerge as dominant execution time bottlenecks.

The program rewriting strategies by *Mantra* can reduce Trap Transfer and Shuttling overheads. For instance, in workloads such as `QAOA_8Q`, `UCC_15`, and `VQE_LiH`, *Mantra* eliminates the need for Trap Transfer, unlike the Standard approach. Across seven workloads, the geometric mean of Trap Transfer execution overhead is reduced by approximately 64%. Similarly, in workloads such as `PO_9Q`, `QAOA_8Q`, and `QNN_24`, *Mantra* efficiently reduces Shuttling time compared to the Standard. On the geometric mean across all workloads, Shuttling is reduced by approximately 43%. When these improvements are combined, the overall execution is reduced by approximately 57% on the geometric mean across all workloads. These results demonstrate that *Mantra* not only reduces zone-to-zone movement but also effectively suppresses execution overhead in scenarios where one-qubit and two-qubit gate operations occur within the same zone.

5.5 Evaluations on Zoned/Non-Zoned Architectures

We compare the performance of different compilers on zoned (Type 1) and non-zoned (Type 3) gate operational policies.

On the architectural adoption perspective: Zoned architectures can achieve better fidelity than non-zoned architectures by isolating gate execution zones. Unfortunately, they may require longer execution times due to the overhead associated with inter-zone movement. As shown in Fig. 14, results indicate that zoned architectures can take an average of 7.3× longer in standard execution compared to non-zoned architectures. This increased execution time may raise concerns for those considering the adoption of zoned architectures. Nevertheless, we argue that achieving high-accuracy computational results is often more critical, even if it comes at the cost of longer execution times. *Mantra* supports this argument by reducing the execution time of zoned architectures by 65% compared to standard execution. Although the execution time with *Mantra* on zoned architectures still remains longer than that of non-zoned architectures, it achieves a reasonable runtime of 2.6× and far improves program fidelity, increasing it from 9% to 52% on average over standard execution in non-zoned architectures.

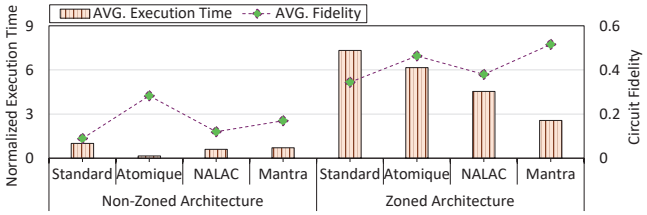

Figure 14. Comparison of normalized average execution time and fidelity of scalable benchmarks under different compilers in non-zoned and zoned neutral atom architectures (lower execution time is better and higher fidelity is better).

On the domain-specific compiler perspective: Unlike non-zoned architectures, zoned architectures pose a new execution bottleneck (i.e., inter-zone movement), which may require a different feature for the compiler to mitigate them. *Mantra* focuses on reducing this inter-zone travel overhead, which can further reduce the overall run-time in the zoned architecture compared to state-of-the-art compilers for neutral atom-based quantum computers. Atomique [83], which is a general-purpose atom array compiler, provides significant performance benefits in various neutral atom array architectures. In non-zoned architectures, Atomique [83] achieves the shortest average execution time, about 7× faster than the standard execution, and provides the highest circuit fidelity among compilers. Atomique [83] leverages the

high-parallelism mapper and router to eliminate shuttling and trap transfer significantly, but this overhead may not be a bottleneck in zoned architectures as described in Fig. 3. As described in Fig. 14, it is observed that the run-time reduction ratio by Atomique [83] is less for zoned architectures than for non-zoned architectures. NALAC [73] is the first and latest compiler for zoned architectures, and it efficiently performs gate execution by leveraging DSatur algorithm-based parallel AOD shuttling. The underlying idea of NALAC [73] mainly considers gate scheduling within entangling zones, not reducing the number of inter-zone travels (note that the LD/ST time itself is reduced by 40% due to the AOD shuttling parallelism.). Thus, NALAC may provide fewer execution time reductions in the zoned architecture than the *Mantra*.

5.6 Execution Breakdown According to Compiler

This section analyzes the execution time breakdown according to different compilers on zoned architectures. As shown in Fig. 15, *Mantra* reduces the average execution time by 64% compared to standard execution, 57% compared to Atomique, and 41% compared to NALAC. *Mantra* achieves the lowest average execution time by reducing LD/ST more than previous compilers, but it also increases trap transfer and shuttling time. This increase can be attributed to revealing the overhead of trap transfer and shuttling time, which was previously hidden within LD/ST operations, as *Mantra* reduces inter-zone movements. In standard execution, while some logical qubits are moved to the storage zone for single-qubit gate execution, other qubits remaining in the entangling zone can perform AOD to SLM transfers in advance. These preparations for later transversal entangling operations are hidden within the inter-zone travel time. By eliminating the need for certain qubits to move to the storage zone, *Mantra* unveils these shuttling and trap transfer times. While *Mantra* primarily focuses on reducing (inter-zone) LD/ST overhead, Atomique targets (intra-zone) trap transfer and shuttling optimization. Combining both methods could further enhance the performance of zoned architectures, as shown in Fig. 15.

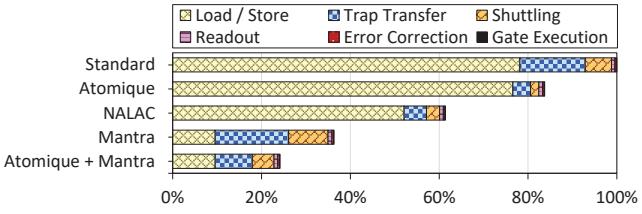

Figure 15. Execution time breakdown analysis with different quantum compilers in the zoned neutral atom architecture.

6 Related Work

This section discusses the latest related studies. The topics below relate to the techniques or schemes utilized by *Mantra*.

Recent studies discuss compilation methodologies to efficiently upload Hamiltonian simulation circuits to superconductor-based quantum processors. *Paulihedral* [45] introduces a compiler framework that optimizes quantum simulation kernels by utilizing a Pauli intermediate representation (Pauli-IR). *Paulihedral* performs a compilation process in Pauli string-level representations instead of gate-based circuit level, resulting in lower transformation overhead. *Tetris* [37] proposes a framework for VQA using an intermediate representation (Tetris-IR) based on Pauli strings. *Tetris* efficiently cancels CXs and minimizes SWAP insertion overhead.

Alternative gate templates [33] could be applied depending on qubit architecture or input quantum states to reduce program execution costs. *Orchestrated Trios* [18] introduces an alternative strategy for the Toffoli gate decomposition. *Orchestrated Trios* [18] adaptively leverages two templates for the Toffoli gate: one that uses 6 CXs, requiring connections between all pairs of 3 qubits, and an alternative that uses 8 CXs, connecting only two pairs among the three qubits. *Relaxed Peephole Optimization* (RPO) [49] replaces some gates with an alternative one when its inputs are in certain states.

The order of gate execution could be rearranged to improve the accuracy or speed of quantum hardware. To reduce program depth, the *Intelligent Approach* [1] modifies the order of ZZ-rotation operations that configure the QAOA cost Hamiltonian. *VAQEM* [65] adjusts the gate alignment to execute some gates at a timing that minimizes the impact of idling errors. While each approach aims to reduce circuit depth and minimize error from idling, the gate alignment of *Mantra* focuses on reducing inter-zone movements of qubits.

7 Conclusion

Inter-zone travels of qubits may be an execution bottleneck on zoned architecture. This may be intensified by existing program structures configured to frequently interleaved gate executions. Naïvely applying the existing program structure to the zoned architecture could require quite inter-zone travels. Using several case studies, we explain program rewriting strategies (*Mantra*) to reduce inter-zone atoms' movements.

Acknowledgments

The authors extend appreciation to the reviewers for the feedback. The authors are grateful to Hengyun Zhou for discussions on the real zoned neutral atom processor, Joonhee Choi for discussions on the tweezer array, and Young Jung Kang for discussions on the quantum error correction. This research was funded by the National Research Foundation of Korea (NRF), supported by the Korean government under the project "Creation of the Quantum Information Science R&D Ecosystem Based on Human Resources" (No. RS-2023-00303229). Won Woo Ro is the corresponding author and the correspondences of this research should be directed to him.

A Preparations for Fault-Tolerant Qubits

The fault-tolerant quantum computing aims to protect quantum information in qubits from errors caused by noise and decoherence [61]. Quantum error correction (QEC) is required to achieve such fault-tolerant quantum computing, and various error correction codes such as Steane code [26, 75], surface code [10, 25, 81], toric code [41], and 3D block code [9] could be applied to neutral atom-based quantum computers.

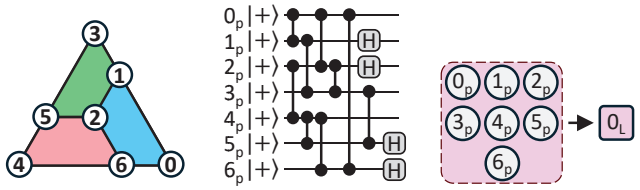

Figure 16. (a) A graph-based representation of 7-qubit Steane code for error correction, (b) A quantum circuit to initialize fault-tolerant logical qubits using Steane code, and (c) A single logical qubit (0_L) consisting of 7 physical qubits.

Among these QEC codes, we adopt the 7-qubit Steane code [75] as shown in Fig. 16, which is relatively straightforward to prepare fault-tolerant qubits and allows to achieve transversal operations in full Clifford gate groups [10]. Fig. 16 (a) represents a graph-based representation of the 7-qubit Steane code, where the colors in the graph represent stabilizer plaquettes that detect errors by the measurement results from seven physical qubits [57, 66]. Fig. 16 (b) shows a quantum circuit to prepare a logical state of $|+\rangle (= (|0\rangle + |1\rangle)/\sqrt{2})$ using the Steane code [10]. This work describes seven physical qubits (circular shaped) encoded by the Steane code as a single logical qubit (square shaped), as shown in Fig. 16 (c).

B Hamiltonian for Time-Optimal Gates

The implementation of entangling gates, such as Controlled-Phase (CPHASE [20]) and Levine-Pichler (LP [32]) gates, leverages the Rydberg blockade effect and van der Waals interactions [10] in neutral atom quantum processors. The Hamiltonian governing these gates could be expressed as follows.

$$H(t) = \sum_i \frac{\Omega_i(t)}{2}\left(\sigma_i^+ + \sigma_i^-\right) + \sum_i \Delta_i n_i + \sum_{i<j} V_{ij} n_i n_j,$$

The parameters in the Hamiltonian are defined as follows [82]. $\Omega_i(t)$ represents the Rabi frequency controlling the transition between the ground state $|g\rangle$ and the Rydberg-excited state $|r\rangle$. The detuning Δ_i is the laser offset from resonance. The operator $n_i = |r\rangle\langle r|$ is the Rydberg population operator, indicating whether the atom is in the Rydberg state. The interaction strength between two atoms i and j is given by $V_{ij} = C_6/r_{ij}^6$, which describes van der Waals interaction as a function of the distance r_{ij} [29]. σ_i^+ and σ_i^- are the raising and lowering operators for the transitions of atom i.

The van der Waals interaction term $V_{ij} n_i n_j$ is critical for inducing phase shifts necessary for entangling operations. By modulating the Rabi frequency $\Omega(t)$ and detuning $\Delta(t)$, the dynamics of the system can be controlled to implement gates with high fidelity and minimal duration [48, 59]. For the CPHASE gate, the Rabi frequency and detuning govern the phase accumulation on the $|11\rangle$ state. Similarly, LP gate applies controlled phase shifts, assigning the same specific phase to $|01\rangle$ and $|10\rangle$ while applying a different one to $|11\rangle$.

C Proposed Arbitrary ZZ-Rotation Protocol

This section discusses a unitary that combines time-optimal CPHASE with LP gate to realize ZZ-interaction operations with arbitrary angles. The target unitary for the $RZZ(\gamma)$ is:

$$RZZ(\gamma) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{i\gamma} & 0 & 0 \\ 0 & 0 & e^{i\gamma} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

It is realized by combining 2 gates. A CPHASE is shown as:

$$U_{\text{CPHASE}}(\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{bmatrix},$$

which applies a phase ϕ to the $|11\rangle$. The CPHASE(ϕ) gate is implemented using a time-optimal protocol, where the Rabi frequency $\Omega(t)$ and detuning $\Delta(t)$ are modulated to achieve the desired phase ϕ on the $|11\rangle$ state [20]. The phase accumulation is given by $\phi(t) = \phi_0 + \frac{\Omega^2}{\Delta} t$, where ϕ_0 is the initial phase offset. To shape the pulse, the Rabi frequency is adjusted as $\Omega(t) = \Omega_{\max} \exp\left(-\frac{t^2}{2\sigma^2}\right)$, where σ determines the width of the pulse, and $\Omega_{\max}$ is the maximum amplitude [20]. This smooth Gaussian profile minimizes off-resonant scattering and enhances gate fidelity. The gate duration is determined by $\tau_{\text{CPHASE}} = \frac{2\pi}{\sqrt{\Delta^2 + \Omega_{\max}^2}}$, where Δ is the detuning, and $\Omega_{\max}$ is optimized to ensure sufficient phase accumulation while minimizing τ_{CPHASE} [20]. An LP gate can be represented as:

$$U_{\text{LP}}(\gamma) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{i\gamma} & 0 & 0 \\ 0 & 0 & e^{i\gamma} & 0 \\ 0 & 0 & 0 & e^{i(2\gamma+\pi)} \end{bmatrix},$$

which applies a phase γ to the $|01\rangle$ and $|10\rangle$ states, and $2\gamma + \pi$ to the $|11\rangle$. When combined, the resulting unitary becomes:

$$U = U_{\text{LP}}(\gamma) \cdot U_{\text{CPHASE}}(\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{i\gamma} & 0 & 0 \\ 0 & 0 & e^{i\gamma} & 0 \\ 0 & 0 & 0 & e^{i(2\gamma+\pi+\phi)} \end{bmatrix}.$$

To cancel the additional phase term introduced by the LP gate and achieve the desired $RZZ(\gamma)$ operation, the CPHASE gate is configured to apply $\phi_{\text{CPHASE}} = -2\gamma - \pi$, thereby ensuring the total phase shift on $|11\rangle$ satisfies $2\gamma + \pi + \phi = 0$.

This proposed gate protocol is expected to improve program execution efficiency not only in the zoned neutral atom architecture but also in the non-zoned architecture. By adopting the time-optimal LP gate [32] and the CPHASE(ϕ) gate [9, 20], *Mantra* can realize arbitrary rotating ZZ-interaction gates with shorter pulse duration than existing CX-based decomposition in both types of neutral atom architectures.

References

[1] Mahabubul Alam, Abdullah Ash-Saki, and Swaroop Ghosh. 2020. Circuit compilation methodologies for quantum approximate optimization algorithm. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 215–228.

[2] Saman Amarasinghe, Riyadh Baghdadi, Zohreh Davoudi, William Detmold, Marc Illa, Assumpta Parreño, Andrew V Pochinsky, Phiala E Shanahan, and Michael L Wagman. 2023. Variational study of two-nucleon systems with lattice QCD. *Physical Review D* 107, 9 (2023), 094508.

[3] Jonathan M Baker, Andrew Litteken, Casey Duckering, Henry Hoffmann, Hannes Bernien, and Frederic T Chong. 2021. Exploiting long-distance interactions and tolerating atom loss in neutral atom quantum architectures. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 818–831.

[4] Philip Ball. 2024. The Best Qubits for Quantum Computing Might Just Be Atoms. *Quantamagazine* (2024).

[5] Panagiotis Kl Barkoutsos, Jerome F Gonthier, Igor Sokolov, Nikolaj Moll, Gian Salis, Andreas Fuhrer, Marc Ganzhorn, Daniel J Egger, Matthias Troyer, Antonio Mezzacapo, et al. 2018. Quantum algorithms for electronic structure calculations: Particle-hole Hamiltonian and optimized wave-function expansions. *Physical Review A* 98, 2 (2018), 022322.

[6] Nora Bauer, Kübra Yeter-Aydeniz, Elias Kokkas, and George Siopsis. 2024. Solving Power Grid Optimization Problems with Rydberg Atoms. *arXiv preprint arXiv:2404.11440* (2024).

[7] Pablo Bermejo, Paolo Braccia, Manuel S. Rudolph, Zoë Holmes, Lukasz Cincio, and M. Cerezo. 2024. Quantum Convolutional Neural Networks are (Effectively) Classically Simulable. arXiv:2408.12739 [quant-ph] https://arxiv.org/abs/2408.12739

[8] Quemix Blog. 2024. Recent realizations of logical quantum processors based on reconfigurable atomic arrays. *https://www.quemix.com/en/notes009/* (2024).

[9] Dolev Bluvstein, Simon J Evered, Alexandra A Geim, Sophie H Li, Hengyun Zhou, Tom Manovitz, Sepehr Ebadi, Madelyn Cain, Marcin Kalinowski, Dominik Hangleiter, et al. 2024. Logical quantum processor based on reconfigurable atom arrays. *Nature* 626, 7997 (2024), 58–65.

[10] Dolev Bluvstein, Harry Levine, Giulia Semeghini, Tout T Wang, Sepehr Ebadi, Marcin Kalinowski, Alexander Keesling, Nishad Maskara, Hannes Pichler, Markus Greiner, et al. 2022. A quantum processor based on coherent transport of entangled atom arrays. *Nature* 604, 7906 (2022), 451–456.

[11] Sebastian Brandhofer, Ilia Polian, and Hans Peter Büchler. 2021. Optimal mapping for near-term quantum architectures based on Rydberg atoms. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 1–7.

[12] A Robert Calderbank and Peter W Shor. 1996. Good quantum error-correcting codes exist. *Physical Review A* 54, 2 (1996), 1098.

[13] Joonhee Choi, Adam L Shaw, Ivaylo S Madjarov, Xin Xie, Ran Finkelstein, Jacob P Covey, Jordan S Cotler, Daniel K Mark, Hsin-Yuan Huang, Anant Kale, et al. 2023. Preparing random states and benchmarking with many-body quantum chaos. *Nature* 613, 7944 (2023), 468–473.

[14] Jahan Claes. 2022. High-threshold fault-tolerant measurement-based quantum computing with biased noise qubits. *Bulletin of the American Physical Society* (2022).

[15] Alexander Cowtan, Will Simmons, and Ross Duncan. 2020. A generic compilation strategy for the unitary coupled cluster ansatz. *arXiv preprint arXiv:2007.10515* (2020).

[16] Diogo Cruz, Romain Fournier, Fabien Gremion, Alix Jeannerot, Kenichi Komagata, Tara Tosic, Jarla Thiesbrummel, Chun Lam Chan, Nicolas Macris, Marc-André Dupertuis, et al. 2019. Efficient quantum algorithms for GHZ and W states, and implementation on the IBM quantum computer. *Advanced Quantum Technologies* 2, 5-6 (2019), 1900015.

[17] Ethan Decker. 2024. Arctic: A Field Programmable Quantum Array Scheduling Technique. *arXiv preprint arXiv:2405.06183* (2024).

[18] Casey Duckering, Jonathan M Baker, Andrew Litteken, and Frederic T Chong. 2021. Orchestrated trios: compiling for efficient communication in quantum programs with 3-qubit gates. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. 375–385.

[19] Albert Einstein, Boris Podolsky, and Nathan Rosen. 1935. Can quantum-mechanical description of physical reality be considered complete? *Physical review* 47, 10 (1935), 777.

[20] Simon J Evered, Dolev Bluvstein, Marcin Kalinowski, Sepehr Ebadi, Tom Manovitz, Hengyun Zhou, Sophie H Li, Alexandra A Geim, Tout T Wang, Nishad Maskara, et al. 2023. High-fidelity parallel entangling gates on a neutral-atom quantum computer. *Nature* 622, 7982 (2023), 268–272.

[21] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028* (2014).

[22] Jernej Rudi Finžgar, Aron Kerschbaumer, Martin JA Schuetz, Christian B Mendl, and Helmut G Katzgraber. 2024. Quantum-informed recursive optimization algorithms. *PRX Quantum* 5, 2 (2024), 020327.

[23] Robert W Floyd. 1964. Bounded context syntactic analysis. *Commun. ACM* 7, 2 (1964), 62–67.

[24] Daniel González-Cuadra, Dolev Bluvstein, Marcin Kalinowski, Raphael Kaubruegger, Nishad Maskara, Piero Naldesi, Torsten V Zache, Adam M Kaufman, Mikhail D Lukin, Hannes Pichler, et al. 2023. Fermionic quantum processing with programmable neutral atom arrays. *Proceedings of the National Academy of Sciences* 120, 35 (2023), e2304294120.

[25] Daniel Gottesman. 1997. *Stabilizer codes and quantum error correction*. California Institute of Technology.

[26] Hadi Goudarzi, Mohammad Javad Dousti, Alireza Shafaei, and Massoud Pedram. 2014. Design of a universal logic block for fault-tolerant realization of any logic operation in trapped-ion quantum circuits. *Quantum information processing* 13 (2014), 1267–1299.

[27] TM Graham, Y Song, J Scott, C Poole, L Phuttitarn, K Jooya, P Eichler, X Jiang, A Marra, B Grinkemeyer, et al. 2022. Multi-qubit entanglement and algorithms on a neutral-atom quantum computer. *Nature* 604, 7906 (2022), 457–462.

[28] Akel Hashim, Rich Rines, Victory Omole, Ravi K Naik, John Mark Kreikebaum, David I Santiago, Frederic T Chong, Irfan Siddiqi, and Pranav Gokhale. 2022. Optimized SWAP networks with equivalent circuit averaging for QAOA. *Physical Review Research* 4, 3 (2022), 033028.

[29] John L Heilbron and Thomas S Kuhn. 1969. The genesis of the Bohr atom. *Historical studies in the physical sciences* 1 (1969), vi–290.

[30] Fei Hua, Yuwei Jin, Yanhao Chen, Suhas Vittal, Kevin Krsulich, Lev S Bishop, John Lapeyre, Ali Javadi-Abhari, and Eddy Z Zhang. 2023. Caqr: A compiler-assisted approach for qubit reuse through dynamic circuit. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*. 59–71.

[31] IBM. 2024. IBM Quantum. *https://quantum-computing.ibm.com/* (2024).

[32] Sven Jandura and Guido Pupillo. 2022. Time-optimal two-and three-qubit gates for Rydberg atoms. *Quantum* 6 (2022), 712.

[33] Enhyeok Jang, Seungwoo Choi, and Won Woo Ro. 2023. Quixote: Improving Fidelity of Quantum Program by Independent Execution of Controlled Gates. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 1–6.

[34] Enhyeok Jang, Dongho Ha, Seungwoo Choi, Youngmin Kim, Jaewon Kwon, Yongju Lee, Sungwoo Ahn, Hyungseok Kim, and Won Woo Ro. 2024. Recompiling QAOA Circuits on Various Rotational Directions. In *Proceedings of the 2024 International Conference on Parallel Architectures and Compilation Techniques*. 309–324.

[35] Ali Javadi-Abhari, Matthew Treinish, Kevin Krsulich, Christopher J. Wood, Jake Lishman, Julien Gacon, Simon Martiel, Paul D. Nation, Lev S. Bishop, Andrew W. Cross, Blake R. Johnson, and Jay M. Gambetta. 2024. Quantum computing with Qiskit. https://doi.org/10.48550/arXiv.2405.08810 arXiv:2405.08810 [quant-ph]

[36] Zhang Jiang, Amir Kalev, Wojciech Mruczkiewicz, and Hartmut Neven. 2020. Optimal fermion-to-qubit mapping via ternary trees with applications to reduced quantum states learning. *Quantum* 4 (2020), 276.

[37] Yuwei Jin, Zirui Li, Fei Hua, Tianyi Hao, Huiyang Zhou, Yipeng Huang, and Eddy Z Zhang. 2024. Tetris: A Compilation Framework for VQA Applications in Quantum Computing. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 277–292.

[38] Alister Johnson, Camille Coti, Allen D Malony, and Johannes Doerfert. 2022. MARTINI: The little match and replace tool for automatic application rewriting with code examples. In *European Conference on Parallel Processing*. Springer, 19–34.

[39] P Jordan and E Wigner. 1928. Über das Paulische Äquivalenzverbot. *Zeitschrift für Physik* 47, 9 (1928), 631–651.

[40] Tyler Keating, Robert L Cook, Aaron M Hankin, Yuan-Yu Jau, Grant W Biedermann, and Ivan H Deutsch. 2015. Robust quantum logic in neutral atoms via adiabatic Rydberg dressing. *Physical Review A* 91, 1 (2015), 012337.

[41] A Yu Kitaev. 1997. Quantum Communication, Computing, and Measurement. In *Proceedings of the 3rd International Conference of Quantum Communication and Measurement, New York: Plenum*.

[42] Steve Kommrusch, Martin Monperrus, and Louis-Noël Pouchet. 2023. Self-Supervised Learning to Prove Equivalence Between Straight-Line Programs via Rewrite Rules. *IEEE Transactions on Software Engineering* 49, 7 (2023), 3771–3792.

[43] Harry Levine, Alexander Keesling, Giulia Semeghini, Ahmed Omran, Tout T Wang, Sepehr Ebadi, Hannes Bernien, Markus Greiner, Vladan Vuletić, Hannes Pichler, et al. 2019. Parallel implementation of high-fidelity multiqubit gates with neutral atoms. *Physical review letters* 123, 17 (2019), 170503.

[44] Ang Li, Samuel Stein, Sriram Krishnamoorthy, and James Ang. 2023. Qasmbench: A low-level quantum benchmark suite for nisq evaluation and simulation. *ACM Transactions on Quantum Computing* 4, 2 (2023), 1–26.

[45] Gushu Li, Anbang Wu, Yunong Shi, Ali Javadi-Abhari, Yufei Ding, and Yuan Xie. 2022. Paulihedral: a generalized block-wise compiler optimization framework for quantum simulation kernels. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. 554–569.

[46] Yiyi Li and Jeff D Thompson. 2024. High-rate and high-fidelity modular interconnects between neutral atom quantum processors. *PRX Quantum* 5, 2 (2024), 020363.

[47] Yongshang Li, Yu Zhang, Mingyu Chen, Xiangyang Li, and Peng Xu. 2023. Timing-aware qubit mapping and gate scheduling adapted to neutral atom quantum computing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 42, 11 (2023), 3768–3780.

[48] Joanna W Lis, Aruku Senoo, William F McGrew, Felix Rönchen, Alec Jenkins, and Adam M Kaufman. 2023. Midcircuit operations using the omg architecture in neutral atom arrays. *Physical Review X* 13, 4 (2023), 041035.

[49] Ji Liu, Luciano Bello, and Huiyang Zhou. 2021. Relaxed peephole optimization: A novel compiler optimization for quantum circuits. In *2021 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*. IEEE, 301–314.

[50] Ji Liu, Alvin Gonzales, Benchen Huang, Zain Hamid Saleem, and Paul Hovland. 2024. QuCLEAR: Clifford Extraction and Absorption for Significant Reduction in Quantum Circuit Size. *arXiv preprint arXiv:2408.13316* (2024).

[51] Yuhao Liu, Shize Che, Junyu Zhou, Yunong Shi, and Gushu Li. 2024. Fermihedral: On the Optimal Compilation for Fermion-to-Qubit Encoding. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*. 382–397.

[52] Yuhao Liu, Kevin Yao, Jonathan Hong, Julien Froustey, Yunong Shi, Ermal Rrapaj, Costin Iancu, and Gushu Li. 2024. Ternary Tree Fermion-to-Qubit Mapping with Hamiltonian Aware Optimization. *arXiv preprint arXiv:2409.02010* (2024).

[53] Stefano Markidis. 2023. Programming quantum neural networks on NISQ systems: an overview of technologies and methodologies. *Entropy* 25, 4 (2023), 694.

[54] Jarrod R McClean, Sergio Boixo, Vadim N Smelyanskiy, Ryan Babbush, and Hartmut Neven. 2018. Barren plateaus in quantum neural network training landscapes. *Nature communications* 9, 1 (2018), 4812.

[55] Anupam Mitra, Michael J Martin, Grant W Biedermann, Alberto M Marino, Pablo M Poggi, and Ivan H Deutsch. 2020. Robust Mølmer-Sørensen gate for neutral atoms using rapid adiabatic Rydberg dressing. *Physical Review A* 101, 3 (2020), 030301.

[56] Michael A Nielsen and Isaac L Chuang. 2010. *Quantum computation and quantum information*. Cambridge university press.

[57] Daniel Nigg, Markus Mueller, Esteban A Martinez, Philipp Schindler, Markus Hennrich, Thomas Monz, Miguel A Martin-Delgado, and Rainer Blatt. 2014. Quantum computations on a topologically encoded qubit. *Science* 345, 6194 (2014), 302–305.

[58] Ben Noble, James W Daniel, et al. 1977. *Applied linear algebra*. Vol. 477. Prentice-Hall Englewood Cliffs, NJ.

[59] MA Norcia, WB Cairncross, K Barnes, P Battaglino, A Brown, MO Brown, K Cassella, C-A Chen, R Coxe, D Crow, et al. 2023. Midcircuit qubit measurement and rearrangement in a Yb 171 atomic array. *Physical Review X* 13, 4 (2023), 041034.

[60] Tirthak Patel, Daniel Silver, and Devesh Tiwari. 2022. Geyser: a compilation framework for quantum computing with neutral atoms. In *Proceedings of the 49th Annual International Symposium on Computer Architecture*. 383–395.

[61] Asher Peres. 1985. Reversible logic and quantum computers. *Physical review A* 32, 6 (1985), 3266.

[62] Grégoire Pichard, Desiree Lim, Étienne Bloch, Julien Vaneecloo, Lilian Bourachot, Gert-Jan Both, Guillaume Mériaux, Sylvain Dutartre, Richard Hostein, Julien Paris, et al. 2024. Rearrangement of individual atoms in a 2000-site optical-tweezer array at cryogenic temperatures. *Physical Review Applied* 22, 2 (2024), 024073.

[63] John Preskill. 2018. Quantum computing in the NISQ era and beyond. *Quantum* 2 (2018), 79.

[64] Nils Quetschlich, Lukas Burgholzer, and Robert Wille. 2023. MQT Bench: Benchmarking software and design automation tools for quantum computing. *Quantum* 7 (2023), 1062.

[65] Gokul Subramanian Ravi, Kaitlin N Smith, Pranav Gokhale, Andrea Mari, Nathan Earnest, Ali Javadi-Abhari, and Frederic T Chong. 2022. Vaqem: A variational approach to quantum error mitigation. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 288–303.

[66] Ciaran Ryan-Anderson, Justin G Bohnet, Kenny Lee, Daniel Gresh, Aaron Hankin, JP Gaebler, David Francois, Alexander Chernoguzov, Dominic Lucchetti, Natalie C Brown, et al. 2021. Realization of real-time fault-tolerant quantum error correction. *Physical Review X* 11, 4 (2021), 041058.

[67] Ludwig Schmid, David F Locher, Manuel Rispler, Sebastian Blatt, Johannes Zeiher, Markus Müller, and Robert Wille. 2024. Computational capabilities and compiler development for neutral atom quantum processors—connecting tool developers and hardware experts. *Quantum Science and Technology* 9, 3 (2024), 033001.

[68] Ludwig Schmid, Sunghye Park, Seokhyeong Kang, and Robert Wille. 2023. Hybrid circuit mapping: leveraging the full spectrum of computational capabilities of neutral atom quantum computers. *arXiv preprint arXiv:2311.14164* (2023).

[69] Pascal Scholl, Adam L Shaw, Richard Bing-Shiun Tsai, Ran Finkelstein, Joonhee Choi, and Manuel Endres. 2023. Erasure conversion in a high-fidelity Rydberg quantum simulator. *Nature* 622, 7982 (2023), 273–278.

[70] David Sherrington and Scott Kirkpatrick. 1975. Solvable model of a spin-glass. *Physical review letters* 35, 26 (1975), 1792.

[71] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, and Ross Duncan. 2020. t| ket>: a retargetable compiler for NISQ devices. *Quantum Science and Technology* 6, 1 (2020), 014003.

[72] Igor O Sokolov, Panagiotis Kl Barkoutsos, Pauline J Ollitrault, Donny Greenberg, Julia Rice, Marco Pistoia, and Ivano Tavernelli. 2020. Quantum orbital-optimized unitary coupled cluster methods in the strongly correlated regime: Can quantum algorithms outperform their classical equivalents? *The Journal of chemical physics* 152, 12 (2020).

[73] Yannick Stade, Ludwig Schmid, Lukas Burgholzer, and Robert Wille. 2024. An Abstract Model and Efficient Routing for Logical Entangling Gates on Zoned Neutral Atom Architectures. *arXiv preprint arXiv:2405.08068* (2024).

[74] Yannick Stade, Ludwig Schmid, Lukas Burgholzer, and Robert Wille. 2024. Optimal State Preparation for Logical Arrays on Zoned Neutral Atom Quantum Computers. arXiv:2411.09738 [quant-ph] https://arxiv.org/abs/2411.09738

[75] Andrew Steane. 1996. Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 452, 1954 (1996), 2551–2577.

[76] Qiming Sun, Timothy C Berkelbach, Nick S Blunt, George H Booth, Sheng Guo, Zhendong Li, Junzi Liu, James D McClain, Elvira R Sayfutyarova, Sandeep Sharma, et al. 2018. PySCF: the Python-based simulations of chemistry framework. *Wiley Interdisciplinary Reviews: Computational Molecular Science* 8, 1 (2018), e1340.

[77] Shinichi Sunami, Shiro Tamiya, Ryotaro Inoue, Hayata Yamasaki, and Akihisa Goban. 2024. Scalable Networking of Neutral-Atom Qubits: Nanofiber-Based Approach for Multiprocessor Fault-Tolerant Quantum Computer. *arXiv preprint arXiv:2407.11111* (2024).

[78] Daniel Bochen Tan, Dolev Bluvstein, Mikhail D Lukin, and Jason Cong. 2024. Compiling quantum circuits for dynamically field-programmable neutral atoms array processors. *Quantum* 8 (2024), 1281.

[79] Daniel Bochen Tan, Shuohao Ping, and Jason Cong. 2024. Depth-optimal addressing of 2D qubit array with 1D controls based on exact binary matrix factorization. In *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 1–6.

[80] Teague Tomesh, Pranav Gokhale, Victory Omole, Gokul Subramanian Ravi, Kaitlin N Smith, Joshua Viszlai, Xin-Chuan Wu, Nikos Hardavellas, Margaret R Martonosi, and Frederic T Chong. 2022. Supermarq: A scalable quantum benchmark suite. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 587–603.

[81] Joshua Viszlai, Sophia Fuhui Lin, Siddharth Dangwal, Jonathan M Baker, and Frederic T Chong. 2023. An architecture for improved surface code connectivity in neutral atoms. *arXiv preprint arXiv:2309.13507* (2023).

[82] Thad G Walker and Mark Saffman. 2012. Entanglement of two atoms using rydberg blockade. In *Advances in Atomic, Molecular, and Optical Physics*. Vol. 61. Elsevier, 81–115.

[83] Hanrui Wang, Pengyu Liu, Daniel Bochen Tan, Yilian Liu, Jiaqi Gu, David Z Pan, Jason Cong, Umut A Acar, and Song Han. 2024. Atomique: A quantum compiler for reconfigurable neutral atom arrays. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 293–309.

[84] H Wang, DB Tan, P Liu, Y Liu, J Gu, J Cong, and S Han. 2024. Q-pilot: Field programmable qubit array compilation with flying ancillas. In *Proceedings of the 61st ACM/IEEE Design Automation Conference (DAC)*. ACM/IEEE.

[85] Ed Younis, Costin C Iancu, Wim Lavrijsen, Marc Davis, Ethan Smith, and USDOE. 2021. Berkeley Quantum Synthesis Toolkit (BQSKit) v1. https://doi.org/10.11578/dc.20210603.2

Received 2024-09-12; accepted 2024-11-04