NEAT: A Neutral-Atom Transpiler for Joint Mapping and Scheduling of Syndrome Extraction Circuits

Anonymous Author(s)

Abstract

Quantum Error Correction (QEC) demands efficient implementation of syndrome extraction circuits. However, existing compilers for neutral-atom processors largely miss the opportunity to co-optimize these circuits by exploiting both the structural properties of Quantum Error-Correcting Codes (QECCs) and the physical constraints of neutral-atom architectures. In this work, we introduce **NEAT**, an SMT-based compiler that jointly optimizes qubit mapping and syndrome extraction scheduling for a broad class of stabilizer-based QECCs, achieving depth-optimal execution with minimal shuttling overhead on neutral-atom platforms. Across a wide range of QECCs, NEAT consistently achieves near-optimal circuit depth and reduces atom movement by **3**×–**30**× compared the baseline compiler Enola. Logical-level simulations further demonstrate **2**×–**20**× lower logical error rates under realistic hardware noise. A hierarchical symmetry-breaking formulation and relaxed parallel-motion constraints substantially improve solver scalability, yielding up to **100**× speedup in compilation time. Together, these results show that NEAT produces depth-optimal, movement-efficient, and logically robust syndrome extraction schedules, while scaling effectively to large QECCs on neutral-atom hardware.

CCS Concepts

• **Hardware** → **Quantum error correction and fault tolerance**.

Keywords

Quantum Error Correction, Neutral-Atom Quantum Computing, SMT-based Compilation

ACM Reference Format:
Anonymous Author(s). 2026. NEAT: A **Ne**utral-**A**tom **T**ranspiler for Joint Mapping and Scheduling of Syndrome Extraction Circuits. In *Proceedings of The Chips to Systems Conference (DAC) (DAC '26).* ACM, New York, NY, USA, 7 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Quantum computing promises to revolutionize information processing by exploiting superposition and entanglement to achieve exponential or polynomial speedups for specific tasks [14, 28, 34]. However, the inherently fragile qubits make quantum computers susceptible to noise and decoherence [19]. To go beyond **Noisy Intermediate-Scale Quantum (NISQ)** and achieve reliable computation, **Quantum Error Correction (QEC)** encodes logical qubits into redundant physical qubits using **Quantum Error-Correcting Codes (QECCs)** and repeatedly performs syndrome extraction to detect and correct errors [13, 33], allowing logical operations with high fidelity. As illustrated in Fig. 1, a QEC workflow involves multiple rounds of syndrome extraction, decoding, and recovery (or updating the Pauli frame [6, 20, 29]). During this process, **syndrome extraction circuits** are particularly critical, as their fidelity and efficiency directly determine the reliability and overall throughput of fault-tolerant quantum computation, especially for neutral-atom platforms where each QEC cycle operates at the millisecond timescale [1, 2].

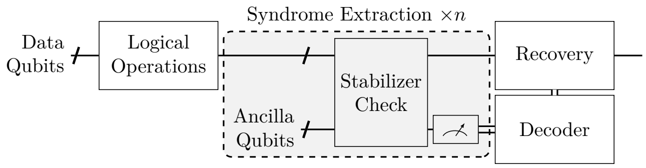


Figure 1: Workflow of quantum error correction. Ancilla qubits are entangled with data qubits via a syndrome extraction circuit to measure the eigenvalues of stabilizer generators. The resulting syndromes are post-processed by the classical decoders to guide recovery operations, thereby preserving the logical information.

Implementing QEC inevitably incurs substantial resource overhead, as each logical qubit requires many physical qubits [11, 25]. Among competing platforms, such as superconducting [8] and ion-trap [15], **neutral-atom quantum computing (NAQC)** offers a uniquely scalable path forward due to its high qubit density and reconfigurable geometry. These features in NAQC—large system size, reconfigurable connectivity, and long-range interactions—are particularly advantageous for implementing QECCs [11, 25], many of which are difficult or impossible to realize on fixed-connectivity superconducting or ion-trap devices.

However, existing quantum compilers, originally developed for NISQ-era superconducting or ion-trap circuits [18, 35], neglect key NAQC constraints—such as shuttling-induced errors, Rydberg blockade radius, and laser scheduling conflicts—limiting their applicability to large-scale atomic processors. Meanwhile, existing NAQC compilers targeting NISQ-level circuits [17, 23, 37, 40] assume a fixed gate schedule and focus on locality or transport distance. As a result, they overlook the unique structure of *syndrome extraction circuits* and cannot fully exploit their commutativity and scheduling flexibility.

To bridge these gaps, we introduce **NEAT (Neutral-Atom Transpiler)**, an SMT-based compiler that jointly optimizes mapping and syndrome-extraction scheduling. Compared to prior SMT-based

DAC '26, Long Beach, California, USA

ACM ISBN 978-1-4503-XXXX-X/2026/07
https://doi.org/XXXXXXX.XXXXXXX

QEC schedulers [26, 41], which assume fixed qubit placements and ignore transport, NEAT explicitly models atomic motion and motion-safety predicates within the same SMT formulation as logical commutation. In contrast to existing NAQC compilers [17, 23, 37], which treat the schedule as a fixed input, NEAT co-designs the qubit movement and gate order. NEAT is, to our knowledge, the first framework that jointly optimizes neutral-atom mapping and gate scheduling in a single symbolic model tailored to neutral-atom architectures. In summary, our contributions are as follows:

- **Joint mapping and scheduling optimization.** We formulate the syndrome extraction compilation problem on neutral-atom platforms as a joint spatiotemporal optimization problem and encode it as a compact SMT model that achieves depth-optimal scheduling while naturally enforcing Rydberg-interaction and motion-safety constraints.
- **End-to-end logical evaluation.** We develop an end-to-end logical-level evaluation framework for benchmarking logical-level fidelity in QEC scenarios, in contrast to the circuit-level fidelity metric used in previous NAQC compilers.
- **Scalable SMT formulation.** We combine a multi-level symmetry-breaking formulation with relaxed parallel-motion constraints to eliminate redundant solutions and substantially improve the compiler's scalability.

Extensive evaluations show that NEAT consistently achieves near-optimal circuit depths and reduced atomic movement overhead, leading to improved logical fidelity and demonstrating its potential as a hardware-aware QEC compiler for neutral-atom quantum processors. All code and data are available in an anonymized repository and will be fully released upon publication.[1]

2 Background

In this section, we briefly review the fundamentals of QECC and neutral-atom architectures that underlie our proposed framework.

2.1 Quantum Error-Correcting Code

A QECC that uses n physical qubits to encode k logical qubits with a code distance d is denoted as an $[[n, k, d]]$ code. For a code on n qubits with m stabilizer checks, the parity-check matrix $H \in \{0, 1\}^{m \times 2n}$, represented in *binary symplectic form (BSF)*, is a binary matrix where each row specifies the X and Z components of a stabilizer. In this representation, each row of H corresponds to a Pauli operator described by two binary vectors—one for its X components and one for its Z components. Two such Pauli operators commute if and only if their symplectic inner product equals zero over $\mathbb{F}_2$. This leads directly to the **symplectic self-orthogonality** of the parity-check matrix:

$$H^X (H^Z)^\top = 0 \pmod 2. \qquad (1)$$

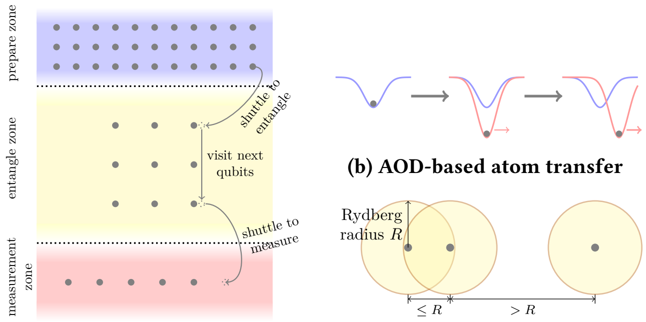
The stabilizer check is implemented by the syndrome extraction circuit. In practice, this involves entangling data qubits with ancilla qubits via CX or CZ gates. The circuit depth directly affects reliability, as deeper circuits accumulate more noise, leading to higher logical error rates. Optimizing these circuits is therefore essential for implementing QECCs on practical hardware such as neutral-atom processors.

2.2 Neutral-Atom Quantum Computing

By trapping individual atoms in arrays of optical tweezers or optical lattices, current neutral-atom processors realize thousands of movable qubits with long coherence times and high-fidelity Rydberg gates [1, 2, 7, 10, 24]. At the same time, emerging zone-based architectures support continuous qubit reloading and mid-circuit correction, further enhancing scalability and robustness [1, 2, 7].

Example 1 (Syndrome extraction in a zoned architecture). *A particularly promising layout for fault-tolerant quantum computing is the* zoned architecture*, which spatially separates different functional regions of the processor [1, 2], as illustrated in Fig. 2a. The syndrome extraction process on such a neutral-atom platform, as shown in Fig. 1, proceeds as follows:*

- ***Data qubits:*** *Before syndrome extraction, data qubits are held in the* entanglement zone*, where they await interaction with ancilla qubits.*
- ***Ancilla qubits:*** *Ancilla qubits are cooled and initialized in the* preparation zone*, then shuttled to the entanglement zone to interact with data qubits. After the* stabilizer checks*, they are transported to the* measurement zone *for readout.*



(a) Schematic of zone architecture

(b) AOD-based atom transfer

(c) Rydberg two-qubit gate

Figure 2: Neutral-atom zone architecture and basic atom operations.

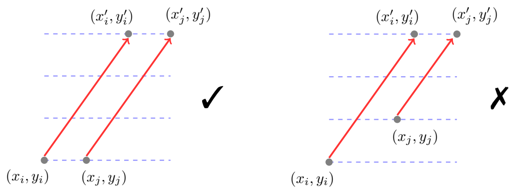
During *syndrome extraction* in NAQC architectures, two primary sources of error must be considered: **shuttling errors** and **two-qubit gate errors**. Shuttling errors arise from atom transport and can be mitigated through *improved qubit mapping*, while two-qubit gate errors are exacerbated by repeated global Rydberg excitation and therefore decrease with *shallower schedules*.

To transfer atoms between different trapping sites, modern neutral-atom processors use **acousto-optic deflectors (AODs)** to shuttle atoms between regions [7]. This enables flexible and reconfigurable qubit arrangements. Such shuttling can be parallelized when the transport trajectories do not intersect, minimizing the overall transfer time, as illustrated in Fig. 2b. Following [17], we say that two qubit moves $e_i : (x_i, y_i) \to (x_i', y_i')$ and $e_j : (x_j, y_j) \to (x_j', y_j')$ are **parallel-legal** if and only if

$$(x_i \star x_j) \iff (x_i' \star x_j') \quad \text{and} \quad (y_i \star y_j) \iff (y_i' \star y_j') \qquad (2)$$

for all $\star \in \{<, =, >\}$. Intuitively, this condition preserves the relative ordering of atoms along both axes and forbids crossing trajectories.

[1] https://anonymous.4open.science/r/dac_2026_neat/



(a) valid non-crossing moves **(b) invalid crossing moves**

Figure 3: Examples of parallel atomic movements.

Example 2 (Parallel movement with parallel-legal). *It is important to note that two movements being parallel does not necessarily imply they can be executed concurrently. Fig. 3 illustrates two such cases with red arrows. In Fig. 3a, two parallel moves are valid since their relative ordering is preserved. In contrast, the configuration in Fig. 3b is invalid because the moves violate the monotonic ordering condition:*

$$y_i < y_j, \text{ but } y_i' = y_j' \tag{3}$$

In neutral-atom processors, two-qubit gates exploit the *Rydberg blockade mechanism* [10, 22], where atoms excited to high-lying Rydberg states interact strongly within a blockade radius R. When two atoms lie within this range, excitation of one inhibits excitation of the other, enabling conditional gate operations, as shown in Fig. 2c. This spatial constraint directly limits which qubits can interact simultaneously and thus influences gate scheduling and circuit layout decisions.

Collectively, these architectural and operational advances position neutral-atom platforms as leading candidates for scalable, continuously operated, fault-tolerant quantum processors.

3 Problem Statement

Efficient compilation of syndrome extraction circuits is essential for realizing fault-tolerant quantum computation, especially on neutral-atom platforms where shuttling and Rydberg-interaction constraints strongly couple logical scheduling with physical motion. Unlike fixed-qubit architectures, neutral-atom execution requires determining *both* the schedule of two-qubit gates and the physical movement of ancilla atoms. These two subproblems are tightly interdependent: the schedule determines ancilla trajectories, while feasible trajectories restrict which gates can be executed in parallel. This coupling makes syndrome extraction a fundamentally *joint* spatiotemporal optimization problem.

3.1 Commutativity in Stabilizer Measurements

Stabilizer measurements admit rich commutation structure. Two-qubit entangling gates acting on different ancilla–data pairs often commute, and even gates across different stabilizers can be reordered when they overlap on an even number of qubits. Exploiting this flexibility is crucial for minimizing circuit depth.

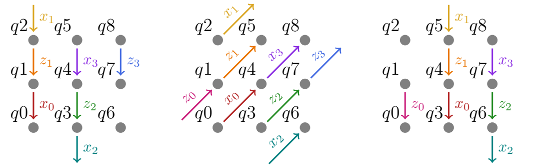
Most existing compilers leverage only *local* commutativity. Frameworks such as **Qiskit** [18] and **TKet** [35] apply peephole rewrites and as-soon-as-possible (ASAP) scheduling. **Tremblay et al.** [38] employ coloration-based grouping for CSS codes to identify parallelizable stabilizers while it is still far from optimal. More recent SMT-based approaches such as **Peham et al.** [26] and **QECC-Synth** [41] begin to capture broader inter-stabilizer commutativity and achieve depth-optimal scheduling *with fixed-qubit placements*.

3.2 Movement-Induced Constraints on NAQC

In neutral-atom architectures, logical operations are constrained not only by circuit topology but also by the physical transport of atoms. Atoms are moved between different locations using AODs and optical traps, as illustrated in Fig. 2b. During transport, idle qubits accumulate decoherence, and the motion process itself introduces additional latency and spatially dependent errors.

Existing NAQC compilers, including **ZAC** [23], **Enola** [37], and **DasAtom** [17], consider locality or minimize transport distance but assume a *fixed* gate order. In particular, Enola also offers coloration-based scheduling, but the internal ancilla visitation order remains fixed by construction. Ignoring commutativity during movement optimization forfeits major opportunities to reduce both circuit depth and total shuttling.

Example 3 (Manual Trajectory Design for the Surface Code). *In superconducting architectures, a standard syndrome extraction schedule of the surface code [9] achieves depth 4. We can also manually design ancilla trajectories on a neutral-atom array as in Fig. 4 that use only three movement operations. However, under the same total distance, many equally short mappings fail to reach depth 4 due to transport conflicts.*



(a) Move 1 **(b) Move 2** **(c) Move 3**

Figure 4: Manual ancilla trajectories for a surface-code stabilizer that together realize a depth-4 syndrome extraction using only three movement operations and a total transport distance of $2 + \sqrt{2}$ grid units.

Therefore, neither scheduling nor mapping can be optimized in isolation. Only a *joint* optimization—reasoning over gate ordering, atom placement, and multi-stage movement simultaneously—can produce hardware-feasible and fidelity-maximizing syndrome extraction circuits.

In this work, we formalize these challenges and present a systematic approach that jointly optimizes both mapping and scheduling. Our objective can be formally stated as follows:

Problem: Given a parity-check matrix H, find a schedule of Rydberg excitations, trap transfers, and shuttling operations that realizes the stabilizer checks on a neutral-atom architecture with high fidelity, while satisfying NAQC hardware constraints (e.g., the parallel-legal-move constraint).

4 Proposed Solution

We formulate the joint mapping and scheduling problem as a **satisfiability modulo theory (SMT)** model that unifies logical commutation and physical motion within a single symbolic reasoning framework. This enables simultaneous optimization of the syndrome extraction order and atom transport while ensuring Rydberg-safe parallelism.

4.1 Symbolic Formulation

Depth-optimal syndrome extraction requires each data qubit to participate in every entangling layer, leaving no idle stages. On neutral-atom hardware, transporting data qubits would introduce excessive motional heating and decoherence. We therefore assume that data qubits remain fixed after initialization, while ancilla qubits are shuttled to their target data qubits via AOD-based transport. In this setting, our goal is to jointly determine the static placement of data qubits on physical traps and the dynamic shuttling trajectories of ancilla qubits throughout syndrome extraction.

Each ancilla movement incurs heating before re-cooling, as illustrated in Fig. 3, making the *number* of movements the dominant error source rather than the geometric travel distance. Accordingly, our formulation is designed so that, among depth-optimal schedules, ancilla trajectories tend to have few movements in practice.

The process proceeds in T discrete *scheduling stages*, each including both ancilla transport and entangling operations. Given the parity-check matrices H^X, H^Z of the code and a bounded 2-D grid representing the atom array, the SMT solver determines:

(1) data-qubit placement,
(2) trajectories of ancilla qubits across stages, and
(3) the stage index of each two-qubit gate.

Index sets. We define $t \in \{0, \dots, T-1\}$ (stages), $q \in \mathcal{Q}$ (data qubits), $a \in \mathcal{A}_X$, $b \in \mathcal{A}_Z$, $c \in \mathcal{A} = \mathcal{A}_X \cup \mathcal{A}_Z$, and grid coordinates $(x, y) \in [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \cap \mathbb{Z}^2$.

Entries $H^X_{a,q} = 1$ and $H^Z_{b,q} = 1$ indicate ancilla–data interactions. Shared data qubits between ancilla a, b are

$$\Omega_{a,b} = \{q \mid H^X_{a,q} = H^Z_{b,q} = 1\}. \tag{4}$$

Interaction times are encoded by $T^{(X)}_{a,q}, T^{(Z)}_{b,q} \in \{-1, 0, \dots, T-1\}$, where -1 means inactive. Each data qubit q has a fixed coordinate (x_q, y_q), and each ancilla follows a trajectory $(x^{(\star)}_{c,t}, y^{(\star)}_{c,t})$. A binary variable $\ell_t \in \{0, 1\}$ marks activation of the global Rydberg laser at stage t; when $\ell_t = 1$, ancilla–data pairs within the blockade radius R execute two-qubit gates (see Fig. 2c).

4.2 Constraints

The model constraints ensure logical correctness, temporal exclusivity, and geometric feasibility.

(C1) Check-matrix activation. Visitation variables are active only when required by the parity-check matrices:

$$H^{(\star)}_{c,q} = 0 \iff T^{(\star)}_{c,q} = -1, \tag{5}$$

$$H^{(\star)}_{c,q} = 1 \iff 0 \le T^{(\star)}_{c,q} \le T - 1. \tag{6}$$

(C2) Temporal exclusivity. Each qubit and ancilla can participate in at most one two-qubit gate per layer:

$$\forall q \neq q' \ : \ H^{(\star)}_{c,q} = H^{(\star)}_{c,q'} = 1 \Rightarrow T^{(\star)}_{c,q} \neq T^{(\star)}_{c,q'}, \tag{7}$$

$$\forall c \neq c' \ : \ H^{(\star)}_{c,q} = H^{(\star)}_{c',q} = 1 \Rightarrow T^{(\star)}_{c,q} \neq T^{(\star)}_{c',q}. \tag{8}$$

(C3) Commutation parity. For overlapping X- and Z-stabilizers a and b,

$$\sum_{q \in \Omega_{a,b}} \mathbf{1}[T^{(X)}_{a,q} < T^{(Z)}_{b,q}] \equiv 0 \pmod 2, \tag{9}$$

which preserves the global commutativity discussed in Section 3.1.

(C4) Spatial–temporal linking. Whenever $T^{(\star)}_{c,q} = t$, the ancilla and data qubit must be co-located and the global laser active:

$$T^{(\star)}_{c,q} = t \iff (x^{(\star)}_{c,t}, y^{(\star)}_{c,t}) = (x_q, y_q) \ \wedge \ \ell_t = 1. \tag{10}$$

For alternative architectural assumptions, the co-location constraint can be generalized as:

$$T^{(\star)}_{c,q} = t \iff \left\|(x^{(\star)}_{c,t}, y^{(\star)}_{c,t}) - (x_q, y_q)\right\|_2 \le R \ \wedge \ \ell_t = 1. \tag{11}$$

where R denotes the effective *Rydberg interaction radius* illustrated in Fig. 2c.

(C5) Spatial exclusivity. All data and ancilla qubits must occupy unique grid sites:

$$\forall q \neq q' \ : \ (x_q, y_q) \neq (x_{q'}, y_{q'}), \tag{12}$$

$$\forall t, \ c \neq c' \ : \ (x^{(\star)}_{c,t}, y^{(\star)}_{c,t}) \neq (x^{(\star)}_{c',t}, y^{(\star)}_{c',t}). \tag{13}$$

(C6) Motion safety: Let $e_{c,t} = ((x^{(\star)}_{c,t}, y^{(\star)}_{c,t}), (x^{(\star)}_{c,t+1}, y^{(\star)}_{c,t+1}))$. Parallel moves are allowed only if non-conflicting:

$$\Pi(e_{c,t}, e_{c',t}) = 1 \ \text{ or at least one of the moves is static.} \tag{14}$$

This enforces the hardware-aware safety condition: parallel atomic movements are permitted only when geometrically non-conflicting according to the parallel-legal move rule in Eq. 2. In this work, we consider two choices of the predicate Π: (1) a strict "no-crossing" rule derived directly from Eq. 2, and (2) a relaxed "parallel-same" rule that permits parallel motion only when the two movement vectors are identical (Fig. 3a). These two motion-safety predicates primarily affect solver scalability, and we compare their impact in Section 5.3.

4.3 Objective Function

The optimization goal is to minimize the total number of scheduling stages:

$$\min \max_{c,q} T^{(\star)}_{c,q}. \tag{15}$$

Minimizing the maximal interaction time reduces both the total number of movement stages and the sequential two-qubit layers required for full syndrome extraction, directly improving overall execution fidelity. While we focus on 2-D neutral-atom arrays with global Rydberg pulses, the constraints (C4)–(C6) can be adapted to alternative architectures by modifying the interaction-radius and motion-safety predicates.

4.4 Symmetry Breaking

The symbolic formulation admits multiple equivalent solutions due to spatial and temporal symmetries. In particular, if $\mathcal{T} = \{T_{c,q}^{(\star)}\}$ is a valid schedule, then the time-reversed schedule $(T-1) - T_{c,q}^{(\star)}$ is also valid; similarly, any spatial placement can be rotated or reflected within the dihedral group D_4 without violating the constraints in (C1)–(C6). Exploring all such symmetric solutions severely degrades solver performance.

We therefore introduce two practical levels of symmetry-breaking constraints [31]:

- **Level 1 (sum-vector inequalities).** Lightweight linear inequalities fix the temporal direction and select a representative spatial sector of the D_4 orbit by constraining the average visitation time and the signed sums of (x_q, y_q) around a reference center.
- **Level 2 (lexicographic order).** A stronger form that enforces lexicographic minimality of the decision-variable vector under all temporal and spatial group operations, removing almost all symmetry-related ties at the cost of additional constraints.

In practice, Level 1 already provides substantial speedups with modest overhead, while Level 2 offers further improvements on the largest instances. We compare both choices empirically in Section 5.3.

Overall, our solution encodes qubit placement, ancilla motion, and stabilizer-interaction timing into a single SMT formulation that unifies stabilizer commutation rules with hardware constraints on NAQC. By minimizing the maximal interaction time and applying hierarchical symmetry breaking to remove redundant spatial and temporal equivalents, the solver produces depth-optimal schedules that, in all our benchmarks, also exhibit substantially reduced ancilla movement compared to existing NAQC compilers. This symbolic framework forms the basis of **NEAT**, a neutral-atom transpiler that jointly handles mapping and syndrome extraction scheduling for general QECCs.

5 Evaluation

We evaluate **NEAT** using two complementary experiments. The first measures end-to-end physical performance on representative stabilizer codes. We assess the generality and scalability of NEAT, using **Enola** [37] as a baseline because its problem formulation—movement-aware scheduling on 2-D neutral-atom arrays—is the closest to ours, enabling a fair and meaningful comparison. Table 1 lists the physical parameters used for movement-cost estimation and Stim-based simulations. The second analyzes the scalability of our compiler through an ablation study.

Table 1: Key parameters, where *l* represents the default spacing (i.e., unit distance) in the SLM array, and *a* denotes the acceleration of qubit movement. [37]

Parameter	f_{2q}	f_{trans}	T_2	T_{2q}	T_{trans}	l	a
Value	99.5%	99.9%	1.5 s	360 ns	1.5 μs	15 μm	2750 m/s^2

All optimization instances are solved using the CP-SAT backend of OR-Tools [27] on a workstation equipped with eight Intel Xeon Platinum 8253 CPUs (2.20 GHz, 256 threads) under 64-bit Linux. All experiments are executed under a 2-hour timeout (7200 s).

5.1 Performance Across QECCs

As discussed in Section 3.2, Enola cannot produce depth-optimal syndrome-extraction circuits. We therefore use NEAT-generated schedules as inputs to Enola so that both compilers share the same two-qubit interaction skeleton, yielding identical circuit depths and enabling a fair comparison of movement overhead.

As shown in Table 2, NEAT reduces the number of movement operations for Surface7 (58 → 3) and for Surface13 (177 → 3), while shortening the total travel distance by factors of 3.0× and 6.9×, respectively. For Planar13, NEAT uses only 3 moves versus 292 for Enola, and the corresponding total distance drops by over *one order of magnitude* (1580 vs. 11115 grid units). Even on other small codes, NEAT consistently matches or improves upon Enola in both movement count and total distance. Overall, these results demonstrate that our joint spatial–temporal formulation consistently produces motion-efficient schedules with near-optimal circuit depths that scale robustly across heterogeneous QECCs.

5.2 Logical-Level Fidelity Validation

Prior NAQC compilers rely on analytic fidelity models that multiply per-gate and per-qubit noise factors. However, such fidelity scores are *ill-suited* for QEC settings: these models penalize circuits with more qubits and more gates, even though larger circuits usually correspond to higher-distance codes with substantially better fault-tolerant behavior.

For instance, a NEAT-compiled $d = 7$ surface code receives an estimated fidelity of $\sim$ 0.4 under such models, whereas a larger $d = 13$ surface code is assigned a fidelity below 0.05—yet the d=13 code achieves orders-of-magnitude lower logical error rate in practice. This discrepancy motivates using end-to-end logical-level simulation rather than analytic fidelity limited to the circuit level.

To this end, we implement an end-to-end *logical-level simulation framework* based on **Stim** [12] to evaluate the logical error rates. Our simulations are based on the standard memory experiments [4, 9], including d rounds of syndrome extraction with circuit-level noise parameters defined as in Table 1. The *logical error rates* (LER) are illustrated in Fig. 5. Both NEAT and Enola circuits are evaluated under the same circuit-level noise model and decoded with the same decoder configuration (**BP+OSD** [30] or **Pymatching** [16]).

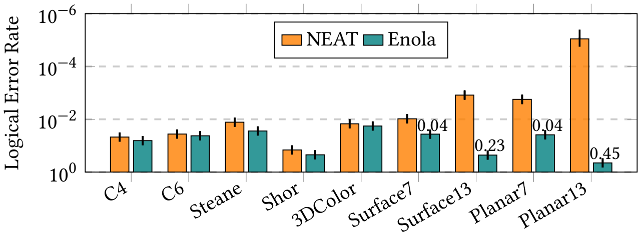


Figure 5: Logical error rates of NEAT and Enola [37] compiled circuits under Stim-based noise simulation.

Table 2: Optimized circuit depth and motion metrics for representative QECCs. Distances are given in grid units.

Code	Parameters	|Q|	#gate	#depth		NEAT		Enola [37]	
				device-agnostic	NEAT	#move	total distance	#move	total distance
*C*4 [39]	[[4, 2, 2]]	6	8	4	5	4	8.0	11	17.899
*C*6 [20]	[[6, 2, 2]]	10	16	4	4	4	17.656	8	29.656
Steane [36]	[[7, 1, 3]]	13	24	6	6	6	51.901	23	51.549
Shor [32]	[[9, 1, 3]]	17	24	6	6	5	37.142	19	48.892
3D Color [21]	[[8, 3, 2]]	13	24	8	8	7	54.422	27	56.507
Surface7 [3]	[[49, 1, 7]]	97	168	4	4	3	161.882	58	493.827
Surface13 [3]	[[289, 1, 13]]	337	624	4	4	3	568.587	177	3907.205
Planar7 [5]	[[85, 1, 7]]	169	312	4	4	3	355.829	100	1291.641
Planar13 [5]	[[313, 1, 13]]	625	1200	4	4	3	1580.122	292	11115.277

Across all tested code examples, NEAT achieves significantly lower LER than Enola, with improvements of 2×–20× for surface codes where movement-induced errors dominate error accumulation and similarly large gains on planar codes. Importantly, NEAT preserves the expected **fault-tolerance trend**: both Surface7 → 13 and Planar7 → 13 exhibit clear LER reductions as distance increases. Enola shows no such behavior—its d=13 surface and planar circuits even incur *higher* LER than their d=7 counterparts. These results confirm that NEAT's joint optimization of mapping and scheduling reduces noise accumulation while maintaining the correct logical-level scaling on neutral-atom hardware.

5.3 Scalability Ablation

To quantify how each component of our formulation contributes to solver scalability, we conduct an ablation on the surface code, varying:

- **Symmetry-breaking level:** Level 0 (none), Level 1 (sum-vector), and Level 2 (lexicographic).
- **Motion-safety predicate:** either "no-crossing" (derived from Eq. 2) or "parallel-same" (movement vectors are strictly equal, as in Fig. 3a).

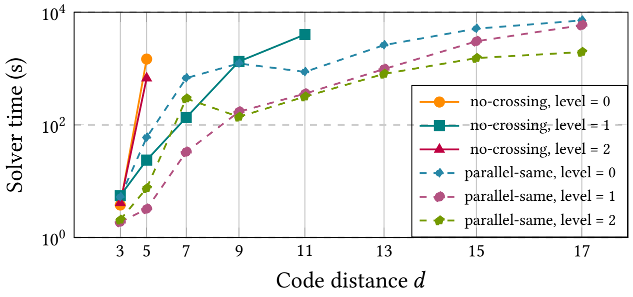


Figure 6: Solver runtime for surface codes under different configuration settings. Missing points indicate failure to reach depth 4 within the 7200 s timeout.

All configurations target the known minimum depth 4 for surface-code syndrome extraction, as in Fig. 4.

Fig. 6 shows the solver runtime scaling with code distances. Across all settings, Level 0 is always the slowest, while Level 1 reduces runtime by up to 62× (no-crossing) or about one order of magnitude (parallel-same). Level 2 adds only minor benefit and mainly helps on the largest parallel-same instances. For any symmetry level, parallel-same is typically 3–11× faster than no-crossing. Combining Level 1 with parallel-same yields the best overall performance and, at d=5, gives **nearly two orders of magnitude** improvement over the no-crossing, Level 0 baseline. Meanwhile, Level 2 with parallel-same is the only setting that reaches d=17.

Overall, our evaluation shows that NEAT effectively unifies symbolic reasoning with neutral-atom hardware constraints. The SMT formulation, combined with hierarchical symmetry breaking, scales to realistic QEC codes and reduces solver runtime, while preserving depth optimality and motion safety. At the physical level, NEAT's movement-aware schedules substantially reduce atomic transport and yield lower logical error rates under Stim-based simulations compared to Enola.

Although solving the SMT model remains time-consuming for the largest instances, syndrome-extraction circuits are compiled *offline* and reused across many QEC cycles and algorithmic runs. In typical fault-tolerant settings where a fixed code and syndrome extraction pattern are executed for millions of cycles, this one-time compilation cost is amortized over long computations and becomes negligible compared to the overall runtime.

6 Conclusions

In this study, we present NEAT, a hardware-aware compiler that formulates syndrome-extraction for a broad class of stabilizer-based QECCs on 2-D neutral-atom architectures as a unified scheduling-and-placement problem solved via SMT. By combining stabilizer commutativity, Rydberg-safe parallelism, and motion-safety constraints within a search space with broken symmetries, NEAT jointly optimizes mapping and scheduling, reducing both circuit depth and ancilla motion. Across all tested codes, NEAT consistently finds schedules that require substantially less atomic transport than state-of-the-art baselines.

Future work includes extending the approach to larger-distance codes through hierarchical or heuristic SMT formulations, exploring the use of additional ancilla qubits to simplify extraction circuits, and integrating NEAT with continuous-operation and mid-circuit-recycling architectures to support scalable fault-tolerant neutral-atom computation.

References

[1] 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.

[2] Dolev Bluvstein, Alexandra A Geim, Sophie H Li, Simon J Evered, J Pablo Bonilla Ataides, Gefen Baranes, Andi Gu, Tom Manovitz, Muqing Xu, Marcin Kalinowski, et al. 2025. A fault-tolerant neutral-atom architecture for universal quantum computation. *Nature* (2025), 1–3.

[3] H. Bombin and M. A. Martin-Delgado. 2007. Optimal resources for topological two-dimensional stabilizer codes: Comparative study. *Phys. Rev. A* 76 (Jul 2007), 012305. Issue 1. doi:10.1103/PhysRevA.76.012305

[4] Sergey Bravyi, Andrew W Cross, Jay M Gambetta, Dmitri Maslov, Patrick Rall, and Theodore J Yoder. 2024. High-threshold and low-overhead fault-tolerant quantum memory. *Nature* 627, 8005 (2024), 778–782.

[5] Sergey B Bravyi and A Yu Kitaev. 1998. Quantum codes on a lattice with boundary. *arXiv preprint quant-ph/9811052* (1998).

[6] Christopher Chamberland, Pavithran Iyer, and David Poulin. 2018. Fault-tolerant quantum computing in the Pauli or Clifford frame with slow error diagnostics. *Quantum* 2 (2018), 43. doi:10.22331/q-2018-01-04-43

[7] Neng-Chun Chiu, Elias C Trapp, Jinen Guo, Mohamed H Abobeih, Luke M Stewart, Simon Hollerith, Pavel L Stroganov, Marcin Kalinowski, Alexandra A Geim, Simon J Evered, et al. 2025. Continuous operation of a coherent 3,000-qubit system. *Nature* (2025), 1–3.

[8] John Clarke and Frank K Wilhelm. 2008. Superconducting quantum bits. *Nature* 453, 7198 (2008), 1031–1042.

[9] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. 2002. Topological quantum memory. *J. Math. Phys.* 43, 9 (Sept. 2002), 4452–4505. doi:10.1063/1.1499754

[10] 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.

[11] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. 2012. Surface codes: Towards practical large-scale quantum computation. *Physical Review A—Atomic, Molecular, and Optical Physics* 86, 3 (2012), 032324.

[12] Craig Gidney. 2021. Stim: a fast stabilizer circuit simulator. *Quantum* 5 (July 2021), 497. doi:10.22331/q-2021-07-06-497

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

[14] Lov K Grover. 1996. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 212–219.

[15] Hartmut Häffner, Christian F Roos, and Rainer Blatt. 2008. Quantum computing with trapped ions. *Physics reports* 469, 4 (2008), 155–203.

[16] Oscar Higgott and Craig Gidney. 2025. Sparse blossom: correcting a million errors per core second with minimum-weight matching. *Quantum* 9 (2025), 1600. doi:10.22331/q-2025-01-20-1600

[17] Yunqi Huang, Dingchao Gao, Shenggang Ying, and Sanjiang Li. 2025. Dasatom: A divide-and-shuttle atom approach to quantum circuit transformation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2025).

[18] 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, et al. 2024. Quantum computing with Qiskit. *arXiv preprint arXiv:2405.08810* (2024).

[19] A Yu Kitaev. 1997. Quantum error correction with imperfect gates. In *Quantum communication, computing, and measurement*. Springer, 181–188.

[20] Emanuel Knill. 2005. Quantum computing with realistically noisy devices. *Nature* 434, 7029 (2005), 39–44. doi:10.1038/nature03350

[21] Aleksander Kubica, Beni Yoshida, and Fernando Pastawski. 2015. Unfolding the color code. *New Journal of Physics* 17, 8 (2015), 083026.

[22] 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.

[23] Wan-Hsuan Lin, Daniel Bochen Tan, and Jason Cong. 2025. Reuse-aware compilation for zoned quantum architectures based on neutral atoms. In *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 127–142.

[24] Hannah J Manetsch, Gyohei Nomura, Elie Bataille, Xudong Lv, Kon H Leung, and Manuel Endres. 2025. A tweezer array with 6100 highly coherent atomic qubits. *Nature* (2025), 1–3.

[25] Pavel Panteleev and Gleb Kalachev. 2021. Degenerate quantum LDPC codes with good finite length performance. *Quantum* 5 (2021), 585.

[26] Tom Peham, Ludwig Schmid, Lucas Berent, Markus Müller, and Robert Wille. 2025. Automated Synthesis of Fault-Tolerant State Preparation Circuits for Quantum Error-Correction Codes. *PRX Quantum* 6, 2 (2025), 020330.

[27] Laurent Perron and Frédéric Didier. 2025. *CP-SAT*. Google. https://developers.google.com/optimization/cp/cp_solver/

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

[29] Leon Riesebos, Xiang Fu, Savvas Varsamopoulos, Carmen G Almudever, and Koen Bertels. 2017. Pauli frames for quantum computer architectures. In *Proceedings of the 54th Annual Design Automation Conference 2017*. 1–6. doi:10.1145/3061639.3062300

[30] Joschka Roffe, David R. White, Simon Burton, and Earl Campbell. 2020. Decoding across the quantum low-density parity-check code landscape. *Physical Review Research* 2, 4 (Dec 2020). doi:10.1103/physrevresearch.2.043423

[31] Francesca Rossi, Peter Van Beek, and Toby Walsh. 2006. *Handbook of constraint programming*. Elsevier.

[32] Peter W Shor. 1995. Scheme for reducing decoherence in quantum computer memory. *Physical review A* 52, 4 (1995), R2493.

[33] Peter W Shor. 1996. Fault-tolerant quantum computation. In *Proceedings of 37th conference on foundations of computer science*. IEEE, 56–65.

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

[35] 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.

[36] 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.

[37] Daniel Bochen Tan, Wan-Hsuan Lin, and Jason Cong. 2025. Compilation for dynamically field-programmable qubit arrays with efficient and provably near-optimal scheduling. In *Proceedings of the 30th Asia and South Pacific Design Automation Conference*. 921–929.

[38] Maxime A Tremblay, Nicolas Delfosse, and Michael E Beverland. 2022. Constant-overhead quantum error correction with thin planar connectivity. *Physical Review Letters* 129, 5 (2022), 050504.

[39] Lev Vaidman, Lior Goldenberg, and Stephen Wiesner. 1996. Error prevention scheme with four particles. *Physical Review A* 54, 3 (1996), R1745.

[40] Hanrui Wang, Daniel Bochen Tan, Pengyu Liu, Yilian Liu, Jiaqi Gu, Jason Cong, and Song Han. 2024. Q-pilot: Field programmable qubit array compilation with flying ancillas. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*. 1–6.

[41] Keyi Yin, Hezi Zhang, Xiang Fang, Yunong Shi, Travis S Humble, Ang Li, and Yufei Ding. 2025. QECC-Synth: A Layout Synthesizer for Quantum Error Correction Codes on Sparse Architectures. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*. 876–890.