Zihan Chen* zihan.chen.cs@rutgers.edu Rutgers University Piscataway, New Jersey, USA

Henry Chen hc867@scarletmail.rutgers.edu Rutgers University Piscataway, New Jersey, USA

Yipeng Huang yipeng.huang@rutgers.edu Rutgers University Piscataway, New Jersey, USA Jiakang Li* jiakang.li@rutgers.edu Rutgers University Piscataway, New Jersey, USA

Zirui Li zl606@scarletmail.rutgers.edu Rutgers University Piscataway, New Jersey, USA

Huiyang Zhou hzhou@ncsu.edu North Carolina State University Raleigh, North Carolina, USA

Eddy Z. Zhang eddy.zhengzhang@gmail.com Rutgers University Piscataway, New Jersey, USA Minghao Guo* minghao.guo@rutgers.edu Rutgers University Piscataway, New Jersey, USA

Joel Bierman jhbierma@ncsu.edu North Carolina State University Raleigh, North Carolina, USA

Yuan Liu q_yuanliu@ncsu.edu North Carolina State University Raleigh, North Carolina, USA

Abstract

This paper introduces Genesis, the first compiler designed to support Hamiltonian Simulation on hybrid continuous-variable (CV) and discrete-variable (DV) quantum computing systems. Genesis is a two-level compilation system. At the first level, it decomposes an input Hamiltonian into basis gates using the native instruction set of the target hybrid CV-DV quantum computer. At the second level, it tackles the mapping and routing of qumodes/qubits to implement long-range interactions for the gates decomposed from the first level. Rather than a typical implementation that relies on SWAP primitives similar to qubit-based (or DV-only) systems, we propose an integrated design of connectivity-aware gate synthesis and beamsplitter SWAP insertion tailored for hybrid CV-DV systems. We also introduce an OpenQASM-like domain-specific language (DSL) named CVDV-QASM to represent Hamiltonian in terms of Pauli-exponentials and basic gate sequences from the hybrid CV-DV gate set. Genesis has successfully compiled several important Hamiltonians, including the Bose-Hubbard model, \mathbb{Z}_2 -Higgs model, Hubbard-Holstein model, Heisenberg model and Electron-vibration coupling Hamiltonians, which are critical in domains like quantum field theory, condensed matter physics, and quantum chemistry. Our implementation is available at Genesis-CVDV-Compiler https://github.com/ruadapt/Genesis-CVDV-Compiler.

*These authors have equal contributions.

\odot \odot

This work is licensed under a Creative Commons Attribution 4.0 International License. *ISCA '25, Tokyo, Japan* © 2025 Copyright held by the owner/author(s). ACM ISBN 979-8-4007-1261-6/2025/06 https://doi.org/10.1145/3695053.3731065

CCS Concepts

• Computer systems organization \rightarrow Quantum computing; • Software and its engineering \rightarrow Compilers.

Keywords

Quantum Computing, Hamiltonian Simulation, Hybrid CV-DV, Quantum Architecture, Compiler

ACM Reference Format:

Zihan Chen, Jiakang Li, Minghao Guo, Henry Chen, Zirui Li, Joel Bierman, Yipeng Huang, Huiyang Zhou, Yuan Liu, and Eddy Z. Zhang. 2025. Genesis: A Compiler Framework for Hamiltonian Simulation on Hybrid CV-DV Quantum Computers. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture (ISCA '25), June 21–25, 2025, Tokyo, Japan.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3695053.3731065

1 Introduction

To date, most quantum computing architectures are homogeneous, featuring two-state, discrete-variable (DV) realizations as qubits. The hybrid continuous-variable discrete-variable (CV-DV) quantum architecture is an emerging platform incorporating both qubits and qumodes. A qumode has a countable infinity of states in principle, thereby providing a larger Hilbert space for computation, and often has a longer lifetime than that of a qubit. As a result, qumodes have been an attractive target for quantum error correction. For instance, superconducting cavity architecture was the first to achieve memory quantum error correction above the break-even point [4, 8, 13, 16, 31, 34, 35, 39] with bosonic codes.

Most notably, a hybrid CV-DV system can simulate mixtures of fermionic and bosonic matter. Simulating physical systems has long been considered a key killer application of quantum computers, as ISCA '25, June 21–25, 2025, Tokyo, JapanZihan Chen, Jiakang Li, Minghao Guo, Henry Chen, Zirui Li, Joel Bierman, Yipeng Huang, Huiyang Zhou, Yuan Liu, and Eddy Z. Zhang

originally proposed by Feynman [12]. While DV qubits can potentially address Fermion simulation challenges, they are poorly suited for bosonic fields due to the difficulty of mapping bosonic systems to qubits and implementing bosonic field operators on infinitedimensional Hilbert spaces. In contrast, bosonic field operators are natively available in hardware with bosonic modes or qumodes. Recent work has demonstrated the great potential of simulating Fermion-Boson mixtures on hybrid CV-DV quantum processors for applications such as material discovery [23], molecular simulation [47, 48], topological models [37], and lattice gauge theory [6].



Figure 1: A typical hybrid CV-DV architecture using the superconducting technology. The qumodes are connected in a sparse manner. Each qubit connects to a qumode, and there is no direct connection between qubits [28].

In this paper, we provide compiler support for mapping Hamiltonian simulation instances onto hybrid CV-DV processors. Compilation support for hybrid CV-DV architectures is still in its infancy. Existing tools, such as Bosonic Qiskit [40], StrawberryField [21], Perceval [49], and Bosehedral [33], offer preliminary support for programming, simulating, and composing circuits—primarily for domain-specific applications like Gaussian Boson Sampling (GBS) or general bosonic circuits. However, none support hybrid CV-DV Hamiltonian simulation. This compilation process requires synthesizing a circuit from the Hamiltonian operator's mathematical representation and mapping the synthesized logical circuit onto a physical circuit while ensuring compatibility with hardware constraints (an example of a superconducting hybrid CV-DV architecture is shown in Fig. 1).

Our work fills this gap. We introduce **Genesis**, the first comprehensive compilation framework for Hamiltonian simulation on hybrid CV-DV computers. It consists of two levels of compilation: **Level-1** decomposes an n-qubit-m-qumode Hamiltonian into universal basis gates, while **Level-2** compilation deals with hardware constraints, including topology constraints, multi-qubit gates, and ancilla qubit/qumode allocation. Our framework successfully compiles important Hamiltonians, such as the Bose-Hubbard model, \mathbb{Z}_2 -Higgs model, Hubbard-Holstein model, and vibration-electron (vibronic) coupling Hamiltonians, crucial in quantum field theory, condensed matter physics, and quantum chemistry. We make the following contributions.

• We propose the first compiler to synthesize circuits from hybrid CV-DV Hamiltonians using a template-rewriting approach based on product formulas and trotterization. This formulation enables automatic rule search and allows further rule expansion without modifying the core algorithm.

- When synthesizing gates from a Hamiltonian, the universality of bosonic systems often requires hybrid qubit-qumode operations. Therefore, the synthesis often requires the allocation of ancilla qubits or qumodes, as well as the mapping and routing of these ancilla qubits/qumodes in the computation. This feature is implemented in our compiler.
- We also provide the first compilation support for multi-qubit Pauli gate implementation in a hybrid CV-DV system. As popular hybrid CV-DV architectures do not have connectivity among qubits, rather, only connectivity exists among qumodes and between qubits and qumodes, we propose to synthesize multi-qubit Pauli gates by leveraging an effect similar to phase kickback in DV systems.
- Besides the compiler support, we propose a domain-specific language (DSL) design for hybrid CV-DV Hamiltonian simulation, which represents Hamiltonians as Pauli strings and basic gate sequences.
- We conducted extensive experiments, evaluating benchmarks with 600-1900 multi-qubit Pauli-string Hamiltonian terms and six key Fermion-Boson Hamiltonian models. We assessed different mapping, routing, and qumode allocation strategies, including the Floating Qubit approach.

2 Background and Motivation

2.1 Advantage of Hybrid CV-DV Systems over CV-only and DV-only Systems

Compared with CV-only or DV-only systems, hybrid CV-DV systems offer several key advantages as summarized in Table 1.

System Characteristics	Hybrid CV-DV	CV-Only	DV-Only	
Energy Truncation for Sim- ulating Bosonic States & Operators	Not-required	Not-required	Required	
Support for Simulating Na- tive Bosonic Operators (e.g. Square-root Factors under Fock basis)	Easy	Easy	Non-trivial [6]	
Gaussian Resource Gener- ation Difficulty	Easy	Easy	Easy [22]	
Non-Gaussian Resource Generation Difficulty	Easy	Difficult [46]	N/A	
Error Channel Complexity	Medium	Low [28]	High	

Table 1: Hybrid CV-DV, CV-only and DV-only Systems.

(1) Advantage in simulation: Quantum simulation is one of the most promising applications of quantum computers. However, simulating fermion-boson mixtures is challenging for DV systems. While mapping fermion operations to qubit operations is possible, representing a bosonic mode with qubits requires truncating its infinite-dimensional Hilbert space. Moreover, implementing native bosonic operations in DV hardware is further complicated, for example, by the quantum arithmetic needed to realize square-root factors under the Fock basis [6, 8, 13, 14]. In contrast, CV hardware employs oscillators with infinite-dimensional Hilbert space and has intrinsic support for native bosonic operators, but has limited support for modeling fermions. A hybrid CV-DV architecture combines



Figure 2: Compilation workflow of Genesis. It first decomposes the Hamiltonian into CV-DV basis gate sets, as well as the Pauli gate we defined in this paper, in the CVDV-QASM language format. Finally, it considers connectivity constraints, performs the hardware mapping and routing stage, and outputs physical circuits.

the strengths of both: it harnesses the larger Hilbert space of the CV system while leveraging discrete qubits to perform fermion-mapped operations in the simulation of fermion-boson mixtures.

(2) Advantage in providing non-Gaussian resources: Achieving universal CV-based quantum computation requires non-Gaussian operations, such as cubic interactions [28, 46]. However, non-Gaussian gates are challenging to realize on CV-only platforms. Alternatively, universal control of oscillators can also be achieved by the addition of qubit-controlled oscillator gates, which are more straightforward and much less costly and have been demonstrated successfully in the lab for superconducting-cavity circuits [11, 42], trapped-ion, and neutral-atom architectures [3, 7, 15, 17, 38].

(3) Advantage in error modeling and QEC: In many CV implementations (such as superconducting resonators or optical modes), photon loss (plus minor phase noise) is the primary error source, yielding Gaussian noise channels [9, 43, 47]. Notably, error mechanisms in CV systems are simpler for a single oscillator with many levels than for multiple qubits of equivalent total dimensionality. Using multiple qubits to represent one oscillator introduces complex error sources, such as crosstalk and correlated errors across control lines, gates, and measurements. In contrast, CV's core error model is comparatively easier to characterize and correct. This simplicity is why bosonic error-correcting codes were the first to achieve the memory break-even point [34, 39]. Hybrid CV-DV systems must address both oscillator photon loss and qubit errors but typically require only one (or a few) qubits per oscillator, reducing qubit crosstalk and simplifying error correction and fault tolerance.

2.2 Hamiltonian Simulation

Hamiltonian Simulation plays a crucial role in understanding the quantum dynamics of various systems and scenarios in quantum physics, chemistry, and materials science [10]. A universal quantum simulator is built upon the time evolution operator e^{-iHt} , where H, known as **Hamiltonian**, typically represents a Hermitian operator, and t represents time (here and the rest of the paper, we take the Planck's constant as $\hbar = 1$). The goal of the Hamiltonian simulation is to decompose the operator e^{-iHt} into a sequence of basis gates for

a given hardware backend. This decomposition enables quantum computers to approximate the system's evolution over time.

A generic n-qubit-m-qumode Hamiltonian for a hybrid CV-DV quantum system can be represented as follows:

$$H = \sum_{k=0}^{4^n - 1} P_k h_k(\hat{a}_1, \hat{a}_1^{\dagger}, \hat{a}_2, \hat{a}_2^{\dagger}, \cdots, \hat{a}_m, \hat{a}_m^{\dagger})$$
(1)

The operator P_k represents an element of the Pauli basis (X, Y, Z, I)on *n* Qubits, where $k = 0, 1, ..., 4^n - 1$. The function $h_k(\hat{a}_i, \hat{a}_i^{\dagger})$ is a finite-degree polynomial in terms of the annihilation \hat{a}_i and creation operator \hat{a}_i^{\dagger} on qumode *i*.

An example is the following spin-Holstein model on N qubits and N qumodes [23]. The first term acts on the *i*-th qubit and qumode jointly, and the second term acts on the *i*-th qumode:

$$H = \sum_{i}^{N} \frac{g_i}{2} Z_i(\hat{a}_i^{\dagger} + \hat{a}_i) + \sum_{i}^{N} \frac{g_i}{2} I(\hat{a}_i^{\dagger} + \hat{a}_i)$$
(2)

The compilation goal is to map the time evolution of a Hamiltonian, i.e., e^{-iHt} , into a minimum and efficient set of basis gates on the hardware. The basis gates include single-qubit gate rotation, single-qumode gates, multi-qumode, and hybrid qubit-qumode gates. A representative gate set is shown in Table 2. Note that two-qubit gates are not included in the basis gates, as there is no direct connection between qubits.

2.3 The Challenges for Compiling A Hybrid CV-DV Hamiltonian Application

Compiling Hamiltonian simulation on a CV-DV system is more challenging than compiling that on a DV system. It must account for (a) unique qumode gate decomposition rules, (b) multi-qubit Paulistring synthesis, and (3) architecture constraints when performing Qubit/Qumode mapping and routing. Specifically, we describe three fundamental differences and challenges:

Challenge 1: Qumode-focused Gate Synthesis

We need to synthesize qumode-only gates for the Hermitian polynomial of annihilation and creation operators of qumodes. For simple Hamiltonian terms, we can perform pattern matching to identify the right basis gate to implement them. For instance, for the Hamiltonian in Equation 2, through Trotterization [29], we can convert e^{-iHt} into a product of two terms $e^{-i\frac{g_i}{2}Z_i(\hat{a}_i^{\dagger}+\hat{a}_i)t}$ and $e^{-i\frac{g_i}{2}(\hat{a}_i^{\dagger}+\hat{a}_i)t}$ for each *i*.

The first term can be pattern-matched to the Control Displacement gate $e^{\sigma_z(\alpha a^{\dagger} - \alpha^* a)}$ in Table 2, by setting the displacement parameter $\alpha = -i\frac{g_i}{2}$. The second term can be pattern matched to the displacement gate $e^{\sigma_z(\alpha a^{\dagger} - \alpha^* a)}$ by parameterizing the displacement $\alpha = -i\frac{g_i}{2}$.

Туре	Gate Name	Definition
Oubit	x, y Rotation	$r_{\varphi}(\theta) = \exp\left[-i\frac{\theta}{2}(\cos\varphi\sigma_x + \sin\varphi\sigma_y)\right]$
Qubit	z Rotation	$r_z(\theta) = \exp\left(-i\frac{\theta}{2}\sigma_z\right)$
	Phase-Space Rotation	$\mathbf{R}(\theta) = \exp\left[-i\theta a^{\dagger}a\right]$
Qumode	Displacement	$D(\alpha) = \exp\left[\left(\alpha a^{\dagger} - \alpha^* a\right)\right]$
	Beam-Splitter	$BS(\theta,\varphi) = \exp\left[-i\frac{\theta}{2}\left(e^{i\varphi}a^{\dagger}b + e^{-i\varphi}ab^{\dagger}\right)\right]$
	Conditional Phase-Space Rotation	$CR(\theta) = \exp\left[-i\frac{\theta}{2}\sigma_z a^{\dagger}a\right]$
	Conditional Parity	$CP = \exp\left[-i\frac{\pi}{2}\sigma_z a^{\dagger}a\right]$
Hybrid	Conditional Displacement	$CD(\alpha) = \exp\left[\sigma_z\left(\alpha a^{\dagger} - \alpha^* a\right)\right]$
	Conditional Beam-Splitter	$\mathrm{CBS}(\theta,\varphi) = \exp\left[-i\frac{\theta}{2}\sigma_z\left(e^{i\varphi}a^{\dagger}b + e^{-i\varphi}ab^{\dagger}\right)\right]$
	Rabi Interaction	$\operatorname{RB}(\theta) = \exp\left[-i\sigma_x\left(\theta a^{\dagger} - \theta^* a\right)\right]$

Table 2: Basis gates in the Hybrid CV-DV System, where σ_i terms represent Pauli terms acting on Qubits, e.g., σ_z is the Pauli-Z operator. *a* and a^{\dagger} are the annihilation and creation operators acting on Qumode. Between different Qubits and Qumode is the tensor product \otimes . We omit the \otimes symbol following the convention in the physics literature [20]. Single-qubit or single-qumode gates typically take around 20 ns [11]. A two-qumode gate or hybrid qubit-qumode gate typically runs in the range of 400–1000 ns [5, 11].

While we have a comprehensive basis gate set of Qubit and Qumode operators, not every term in a Hamiltonian (after Trotterization) can be directly mapped to a basis gate or a combination of basis gates in the quantum hardware. For instance, the term $(a^{\dagger})^2 a^2$ is a Kerr non-linearity [20] that performs a simulation of the Kerr Effect in optics, and it is not easy to be directly implemented in the hardware. It is, in fact, a complex non-linear term requiring a product formula and multiple steps of gate decomposition. In the past, such decomposition was done manually by physicists or theorists. This manual decomposition approach may be time-consuming and error-prone. Furthermore, the hardware vendors typically provide an overcomplete gate set for more flexible and robust selections of gate operations. This further complicates the manual decomposition process.

Challenge 2: Multi-Qubit Pauli-string Synthesis

Since we aim to support the simulation of a generic hybrid CV-DV Hamiltonian, we need to consider qubit-only terms – Paulistring terms. The simulation of Pauli-string terms [18, 25, 26, 36] in Hamiltonians has been extensively studied for DV systems. A Paulistring representation denotes tensor products of Pauli-matrices. Unlike qubits that are connected in a DV system, qubits are typically not directly connected in a hybrid CV-DV system, which is built upon either superconducting or trapped-ion devices [28]. Instead, qubits are connected to qumodes, and qumodes are connected (more details discussed in Challenge 3). A previous idea that uses a sequence of (control) displacement gates to cancel out the effect on a qumode and then "kick back" the phase to the qubits has success-fully implemented R_{ZZ} , CNOT, and Toffoli gates. Inspired by this idea, we develop a method to synthesize **an arbitrary multi-qubit Pauli-string** Hamiltonian by leveraging a multi-qubit Controlled Displacement gate and a trajectory of Displacement gates with respect to different eigenvectors. Moreover, this technique could be extended to create rules for qumode operations controlled by arbitrary multi-qubit Pauli-strings.

Challenge 3: Limited Connectivity Constraints

After gate synthesis from a Hamiltonian, there are three types of gates: two-qumode gates, single-qubit/qumode gates, and hybrid qubit-qumode gates. These gates are generated at the logical level. When mapped to the physical hardware layer, there may be long-range interactions. Just like DV architectures, in the hybrid CV-DV architectures, qubits and qumodes are often not all-to-all connected. Most current hybrid CV-DV systems are designed with either only qumodes connected among themselves, or qubits connected among themselves, but not both at the same time, due to crosstalk issues [28]. We focus on the architecture where qumodes are connected, and qubits can be indirectly connected via qumodes. An example of a superconducting CV-DV device is shown in Fig. 1. The reason is that a qumode has an infinite energy level and can store the information of a qubit, but not vice versa. Also, for the simulation of Fermion-Boson-mixture, the connectivity among gumodes naturally allows the modeling of bosonic interactions.

Oubit/qumode mapping and routing are tightly connected with circuit synthesis described in Challenge 1 and 2. First of all, the synthesis rules allow certain flexibilities, such as the commutation of gates, which can be leveraged to improve the mapping and routing stage to reduce the number of intrinsic SWAP gates. Moreover, the synthesis approach often requires ancilla gumodes, and most importantly, in certain cases, such as compiling multi-qubit entanglement operations with control displacement, it does not require the ancilla qumode to be in the vacuum (ground) state. Rather, it can return the ancilla qumode back to its original state after the sequence is completed for synthesizing a qubit/qumode operation of interest. That means any qumode can be used to assist synthesis and also can be leveraged to improve the mapping and routing efficiency in the hardware circuit compilation stage. This is different from mapping and routing in the traditional DV system, where only SWAP insertion is considered. In certain cases, it can be modeled as a traveling salesman problem (TSP) as described in our Section 3.4.1. We develop a synergistic mapping and routing approach that considers the flexibilities in circuit synthesis.

Summary: Overall, our work is **the first comprehensive compiler framework that addresses the compilation of Hamiltonian simulation on hybrid CV-DV architectures**. It addresses both the gate synthesis and hardware mapping problems, as well as identifies and models the unique compilation problems that only arise in the hybrid CV-DV architecture rather than in the traditional DV context. We show our workflow in Fig. 2, which shows the process of how a given Hamiltonian is decomposed into physical circuits step by step by our methods. We present our design details in Section 3 corresponding to each of the three challenges.

2.4 Current State of CV-DV Technologies

CV-DV systems can be implemented with superconducting, trappedion, or neutral atom devices. In superconducting devices, transmons are used as two-level systems (DV), and microwave resonators storing photons are used as harmonic oscillators (CV). In trapped-ion devices, collective motional modes are oscillators, while ions' internal degrees of freedom are used as qubits. With neutral atoms, atomic motional modes in the optical tweezer are oscillators, and neutral atoms' internal degrees of freedom are used as qubits.

All three technologies have been demonstrated successfully in labs. Superconducting devices typically operate in GHz, trapped-ion in MHz, and neutral atom in KHz. Among them, superconducting devices offer advantages compared to the other two architectures, such as fast and reliable basis gate sets. For instance, microwavecontrolled beam-splitter gates permit fast, high-fidelity oscillator SWAPs, usually in 100 ns, with 99.92% fidelity in the single photon subspace [30]. Hence, the superconducting architecture is our primary evaluation platform in this work, and our work can be extended to other hybrid CV-DV architectures.

3 Our Compilation Framework

As shown in Equation 1, the Hamiltonian consists of contributions from Pauli operators (X, Y, Z, I) and qumode operators (a_i^{\dagger}, a_i) . We synthesize them into a physical circuit with basis gates while considering hardware connectivity constraints in CV-DV systems.

3.1 Qumode Gate Synthesis

3.1.1 Pattern Matching. For sequences involving first or secondorder Hermitian polynomials of annihilation/creation operators, we first try to map them to the basic gates in the hybrid CV-DV gate set (Table 2) through pattern matching. The corresponding basic gates are applied directly if the operators align with specific gate patterns with parameterization – most of the bosonic gates have continuous parameters. These sequences can then be synthesized into implementable gate operations. For example, common terms like $a^{\dagger}a$ and $(a^{\dagger} - a)$ can be mapped to the Phase-Space Rotation gate and the Displacement gate, respectively, as shown below:

$$e^{(-tia^{\dagger}a)} \to R(t), \text{ where } R(\theta) = e^{-i\theta a^{\dagger}a}$$

$$e^{(3ia^{\dagger}+3ia)} \to D(3i), \text{ where } D(\alpha) = e^{\alpha a^{\dagger}-\alpha^{*}a}$$
(3)

The basic gate set in Table 2 defines the matching rules, which allow further extension for wider functionally or architecturespecific gates. However, it only works for simple and fine-grained qumode operator sequences. For complex and large-scale annihilation/creation operator polynomials, we propose the following recursive decomposition process to synthesize them.

3.1.2 Template Matching Decomposition and Approximation. Most Hamiltonians are not directly implementable on quantum hardware and require further decomposition and approximation. Two common methods for Hamiltonian decomposition are Trotterization and the Baker Campbell Hausdorff (BCH) formula, both widely used in quantum circuit synthesis [1, 20, 27, 32]. **Trotterization:** The Trotter-Suzuki decomposition approximates the time evolution operator by breaking it into simplified exponentials:

$$e^{(M+N)t} \approx \left(e^{Mt'}e^{Nt'}\right)^n,\tag{4}$$

where *M* and *N* are parts of the Hamiltonian, t' = t/n, and *n* is the number of Trotter steps. This method is particularly effective for approximating long-time evolution by transforming it into discrete, simplified steps. The purpose of Trotterization is to break a Hamiltonian into a **sum** of terms, which later can be used for approximation as products of small matrix exponentials.

Baker Campbell Hausdorff (BCH): The BCH formula is used to decompose the time evolution operator with non-commuting terms, allowing for a systematic approximation:

$$e^{[M,N]t^2} \approx e^{Mt} e^{Nt} e^{-Mt} e^{-Nt}.$$
(5)

where [M, N] = MN - NM is the commutator. This approach is most suitable for short-time evolution or small non-commutative contributions. BCH is useful, as it can be used to help create a **product** of terms.

Both Trotterization and BCH can be approximated to an arbitrary precision with a large enough order. For the BCH formula, we use the second-order approximation. Trotter will affect the number of gates by repeating the same gate sequence n times, but each gate has a smaller parameter and has a smaller duration. The overall evolution time is the same after Trotterization.

Consider the following commutator (assuming A and B are qumode operators):

$$[\sigma_z \otimes A, I \otimes B] = \sigma_z \otimes AB - \sigma_z \otimes BA = \sigma_z[A, B].$$
(6)

Now also consider that $([A, B] + \{A, B\})/2 = AB$, where $\{A, B\} = AB + BA$ is the anticommutator, the implication is that **we can implement a product of terms of** \hat{a} and \hat{a}^{\dagger} **operators** if $\sigma_z \{A, B\}$ can be implemented. Fortunately, Kang *et al.* [20] provides an implementation of $\sigma_z \{A, B\}$ on CV-DV architecture, as well as another version of [A, B] implementation, which is different from ours in Equation 6. These altogether set a foundation for the template-rewriting-based approach we propose in this paper. We list all these rules in Table 3. With basic **sums** and **products** rules, we can recursively break a Hamiltonian down into sums and products until we reach a point where the term of interest matches a basis gate template in Table 2. We describe the recursive template matching process in the next section.

In Table 3, For a qubit ⊗qumode system, the Hamiltonian pattern \mathcal{B}_M :

$$\mathcal{B}_M = \begin{pmatrix} 0 & M \\ M^{\dagger} & 0 \end{pmatrix},$$

as a Hermitian operator acting on the combined Hilbert space $\mathcal{H}_2 \otimes \mathcal{H}_{\Lambda+1}$, where \mathcal{H}_2 is the qubit Hilbert space and $\mathcal{H}_{\Lambda+1}$ is the qumode Hilbert space.

Certain decomposition rules in Table 3 represent that the target operator is in a block encoding form. For instance, we only need the MN component in the top-left of the Hamiltonian below.

$$\begin{pmatrix} 2MN & 0\\ 0 & -NM - (NM)^{\dagger} \end{pmatrix}.$$
 (7)

ISCA '25, June 21–25, 2025, Tokyo, JapanZihan Chen, Jiakang Li, Minghao Guo, Henry Chen, Zirui Li, Joel Bierman, Yipeng Huang, Huiyang Zhou, Yuan Liu, and Eddy Z. Zhang

Rules	Operator Template	Conditions	Decomposition Output	Reference	Precision
1	$\exp(Mt + Nt) \approx \operatorname{Trotter}(Mt, Nt)$		$(\exp(Mt/k)\exp(Nt/k))^k$	Trotterization	Approx
2	$\exp([Mt, Nt]) \approx BCH(Mt, Nt)$		$\exp(Mt)\exp(Nt)\exp(-Mt)\exp(-Nt)$	BCH	Approx
3	$\exp(t^2[M,N])$	M, N Hermitian	$\exp([it\sigma_i N, it\sigma_i M])$	[20]	Exact
4	$\exp(-it^2\sigma_i\{M,N\})$	M, N Hermitian	$\exp([it\sigma_j M, it\sigma_k N])$	[20]	Exact
5	$\exp(-it^2\sigma_z[M,N])$		$\exp([(itN, it\sigma_z M])$	This paper	Exact
6	$\exp(t^2\sigma_z((MN-(MN)^{\dagger})))$	[M, N] = 0	$\exp([X \cdot it\mathcal{B}_N \cdot X, it\mathcal{B}_M])$	[20]	Exact
7	$\exp(it^2\sigma_z((MN+(MN)^{\dagger})))$	[M, N] = 0	$\exp([S \cdot it\mathcal{B}_M \cdot S^{\dagger}, X \cdot it\mathcal{B}_N \cdot X])$	[20]	Exact
8	$\exp\left(-2it\begin{pmatrix}MN & 0\\ 0 & -MN\end{pmatrix}\right)$	M, N Hermitian	$\exp(-it\sigma_{z}[M,N] - it\sigma_{z}\{M,N\})$	This paper	Exact
9	$\exp\left(2it^2\begin{pmatrix}MN & 0\\ 0 & -MN\end{pmatrix}\right)$	$[M, N] = 0$ $MN = (MN)^{\dagger}$	$\exp([(S \cdot it\mathcal{B}_M \cdot S^{\dagger}, X \cdot it\mathcal{B}_N \cdot X])$	[20]	Exact
10	$\exp\left(2it\mathcal{B}_{MN} ight)$	[M, N] = 0	$X \cdot \exp(t\sigma_y(MN - (MN)^{\dagger}) + it\sigma_x(MN + (MN)^{\dagger})) \cdot X$	[20]	Exact
11	$\exp\left(it \begin{pmatrix} 2MN & 0\\ 0 & -NM - (NM)^{\dagger} \end{pmatrix}\right)$	$MN = (MN)^{\dagger}$	$\exp([S \cdot it\mathcal{B}_M \cdot S^{\dagger}, X \cdot it\mathcal{B}_N \cdot X])$	[20]	Exact
12	$\mathcal{B}_a = \exp\left(2i\alpha \begin{pmatrix} 0 & a \\ a^{\dagger} & 0 \end{pmatrix}\right)$	$\alpha = \alpha^*$	$\exp(i(\pi/2)a^{\dagger}a)\exp(i(\alpha(a^{\dagger}+a))\otimes\sigma_y)\exp(-i(\pi/2)a^{\dagger}a)\exp(i(\alpha(a^{\dagger}+a))\otimes\sigma_x)$	[20]	Approx
13	$\mathcal{B}_{a^{\dagger}} = \exp\left(2ilpha \begin{pmatrix} 0 & a^{\dagger} \\ a & 0 \end{pmatrix} ight)$	$\alpha = \alpha^*$	$\exp(i(\pi/2)a^{\dagger}a)\exp(i(\alpha(a^{\dagger}+a))\otimes\sigma_y)\exp(-i(\pi/2)a^{\dagger}a)\exp(-i(\alpha(a^{\dagger}+a))\otimes\sigma_x)$	This paper	Approx
14	$e^{(P_1P_2\cdots P_n)(\alpha a_k^{\dagger} - \alpha^* a_k)}$		Multi-qubit-controlled displacement: Right hand side (RHS) of Equation (11) first line	[28]	Exact
15	$e^{2i\alpha^2 P_1 P_2 \cdots P_n}$		Multi-Pauli Exponential: Right hand side (RHS) of Equation (9) first line	This Paper	Exact
16	All Native Gates RHS in Table 2		All Native Gates Left Hand Side (LHS) Table 2	[28]	Exact

Table 3: Decomposition rules for operators in hybrid CV-DV systems with corresponding conditions. X and S represent Pauli-X and phase gates, implemented using the single-qubit rotation gates in Table 2. All rules are exact except rules 1, 2, 12, and 13, which are subject to Trotter and BCH approximation errors with respect to the order of Trotter or BCH.

In a qubit-qumode system, an operator $O = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ can be expressed as:

 $O = |0\rangle \langle 0| \otimes A + |0\rangle \langle 1| \otimes B + |1\rangle \langle 0| \otimes C + |1\rangle \langle 1| \otimes D.$

Projecting onto the subspace associated with $|0\rangle$ by setting the ancilla qubit to $|0\rangle$, the effective operator becomes $\langle 0|O|0\rangle = A$, assuming C = 0. Terms involving $|0\rangle \langle 1|$ and $|1\rangle \langle 1|$ vanish due to orthogonality. Thus, by initializing the ancilla and zeroing specific components (e.g., C = 0), we can isolate and simplify blocks of the operator matrix for easier decomposition. If we take the matrix exponential of O, let B = C = 0, since A is on the diagonal, we can also implement e^A via e^O via the block encoding.

3.1.3 Rule-Based Recursive Template Matching for Hamiltonian Decomposition. Section 3.1.1 introduces template matching rules for synthesizing operators into gates, while Section 3.1.2 extends these rules to decompose operator sequences. To decompose complex Hamiltonians, recursive steps and multiple decomposition paths are needed. To enable scalable and automated Hamiltonian simulation, we transform the decomposition task into a Template Matching compilation. This recursive process applies gate synthesis and decomposition rules iteratively to input sequences of annihilation and creation operators so as to reduce a Hamiltonian to basic gates in the Hybrid CV-DV gate set.

Next, we use an example, Hamiltonian in Equation 8, to illustrate the decomposition process. We first apply Rule 1, the Trotterization formula, to approximate it as a product of two exponential operators. As Rule 1 and Rule 2 are common **intermediate steps** in many decomposition Rules, we omit their detailed discussion here.

$$\exp(-iHt) = \exp\left(-i(\omega a^{\dagger}a + \frac{\kappa}{2}(a^{\dagger})^{2}a^{2})t\right)$$
$$\approx \left(\exp(-i\omega a^{\dagger}a\frac{t}{k}) \cdot \exp(-i\frac{\kappa}{2}(a^{\dagger})^{2}a^{2}\frac{t}{k})\right)^{k}$$
(8)

Rule 1 separates the linear term $a^{\dagger}a$ and nonlinear term $(a^{\dagger})^2 a^2$. Each term is then decomposed into basic gates in the Hybrid CV-DV gate set. For the linear term $a^{\dagger}a$, it is synthesized as a Phase-Space Rotation gate using the rules from Section 3.1.1.

Decomposing the nonlinear term $(a^{\dagger})^2 a^2$ is challenging due to the lack of a matching basis gate. This would incur recursively applying the template matching rules from Section 3.1.2 or Table 3 to simplify it. The process follows a recursive search tree, where nodes represent decomposition states (current terms and their settings), and edges correspond to synthesis rules from Tables 2-3.

For a complex term, there may exist multiple ways to break it into subterms, M and N in Table 3. Fig. 3 shows three possible methods for splitting $(a^{\dagger})^2 a^2$ into M and N. This introduces additional complexity or opportunity to the decomposition process due to the multipath selection.

To decompose $(a^{\dagger})^2 a^2$, we select a splitting method for M, N, then apply the template matching rules from Table 3. First, we check the conditions for each rule and filter out those that don't apply. Next, we identify the operator templates for M and N, such



Figure 3: Decomposing $(a^{\dagger})^2 a^2$ into three child states, i.e., splitting it into subterms *M* and *N* using three methods.

as MN, [M, N], $\{M, N\}$, $MN \pm (MN)^{\dagger}$, or \mathcal{B}_{MN} , and proceed with the decomposition accordingly.

We use Depth-First Search (DFS) with backtracking to traverse the recursive decomposition tree, exploring paths deeply and storing valid decomposition paths for evaluation. Dead ends occur when no rule further reduces the operator sequence. The algorithm then backtracks to the last valid node, prunes unnecessary branches, and explores alternative paths.

Handling Dead Ends: For instance, if $M = a^{\dagger}$ and $N = a^{\dagger}a^2$, Rule 11 from Table 3 applies as $MN = (MN)^{\dagger}$. While *M* matches a basic gate, $N (a^{\dagger}a^2)$ requires further decomposition. The path reaches a dead end after trying three splitting methods for *N* without success. The algorithm then discards this branch and backtracks to try an alternative decomposition, such as $M = (a^{\dagger})^2$ and $N = a^2$.

Step 1: For $M = (a^{\dagger})^2$ and $N = a^2$, the condition $MN = (MN)^{\dagger}$ holds, allowing Rule 11 to be applied. This decomposes the term into two subterms, $(S \cdot it\mathcal{B}_M \cdot S^{\dagger})$ and $(X \cdot it\mathcal{B}_N \cdot X)$.

Step 2: One of subterm $(a^{\dagger})^2$ is further decomposed into $M = a^{\dagger}$ and $N = a^{\dagger}$, satisfying the condition [M, N] = 0 and matching the Block Encoding template. Rule 10 is applied to produce two new subterms: $it\sigma_x(MN + (MN)^{\dagger})$ and $t\sigma_u(MN - (MN)^{\dagger})$.

Step 3: At this stage, $M = a^{\dagger}$ and $N = a^{\dagger}$ again. Rule 6 is applied, decomposing the term into $(S \cdot it\mathcal{B}_M \cdot S^{\dagger})$ and $(X \cdot it\mathcal{B}_N \cdot X)$, further simplifying the operators.

Step 4: The Block Encoding template is then decomposed into a sequence of simplified annihilation and creation operators using Rule 13.

Step 5: Finally, the resulting sequence is synthesized into Phase-Space Rotation and Displacement gates from the hybrid CV-DV gate set. The complete 5-step recursive Template Matching decomposition and gate synthesis process is illustrated in Fig. 4.

Discussion: Our mechanism is designed to be **extension-friendly**, **reuse-friendly**, and **customization-friendly**. When expanding two rule sets, such as adding new architecture-specific gates or decomposition rules, our template matching and implementation mechanism remains unaffected. Additionally, we can customize cost metrics, such as fidelity or decomposition approximation, by extending our template-matching into a cost-optimized framework.

Hamiltonian decomposition and gate synthesis are highly challenging tasks. To the best of our knowledge, our work is the first to automatically decompose the six general Hamiltonian models described in Section 4 for the CV-DV system. Our evaluation indicates



Figure 4: Recursive decomposition and gate synthesis of $(a^{\dagger})^2 a^2$ into a sequence of basic gates. The red circle indicates the template matching rules from the Decomposition Rules Set (Table 3), and the blue square represents the template matching rules from the hybrid CV-DV gate set (Table 2).

that the decomposition and synthesis of a given Hamiltonian operator have limited viable pathways. As a result, the trade-offs between latency, gate count, and error rate are not a primary concern at this stage, given the scarcity of successful decomposition pathways. However, providing support to choose among different decomposition and synthesis pathways should be a feasible extension. We leave this extension of our compiler as our future work.

3.2 Multi-qubit Pauli-string Synthesis

We propose a scheme to synthesize an arbitrary multi-qubit Paulistring on hybrid CV-DV platforms. It is inspired by *phase kickback* in DV systems, where the phase of the control qubit is influenced by the operation on the target qubits. In our approach, we use qumodes as a medium to pass entanglement among qubits through a sequence of Conditional Displacement gates, i.e., qubit-controlled qumode Displacement, and unconditional Displacement gates such that the desired operation is achieved on qubits while the effects on qumodes cancel.

We propose, for the first time, the multi-Pauli exponential for the CV-DV system in this paper, that is, an arbitrary Pauli-string exponential for a hybrid CV-DV system built upon displacement and multi-qubit control displacement gates.

The detailed implementation follows a structured decomposition:

$$U = D^{k}(i\alpha) CD^{(k,P_{1\cdots n})}(-\alpha) D^{k}(-i\alpha) CD^{(k,P_{1\cdots n})}(\alpha)$$

= $e^{2i\alpha^{2}P_{1}P_{2}\cdots P_{n}}$ (9)

where $P_1P_2 \cdots P_n$ represents the desired multi-qubit Pauli string $P_1 \otimes P_2 \otimes \cdots \otimes P_n$, $D^k(m)$ is a displacement of m to the k-th qumode, and $CD^{(k,P_1\dots n)}(-\alpha)$ and $CD^{(k,P_1\dots n)}(\alpha)$ are controlled displacement of $\pm \alpha$. During the operation, displacement effects on qumodes return them to the original states, kicking back an overall phase to the qubits, akin to phase kickback in DV systems. This is built on Liu *et al.*[28], which used ancillary qumodes to implement CNOT and R_{ZZ} and single-qubit gates. We extend their approach to cover arbitrary multi-qubit operations with machine compilation.

An example with one qubit and one qumode is as follows.

$$U = D^{k}(i\alpha)CD^{k,P_{1}}(\alpha, -\alpha)D(-i\alpha)CD^{k,Z}(-\alpha, \alpha)$$

= $e^{2i\alpha^{2}Z}$ (10)

If we set P_1 as the Pauli-Z operator, $CD^{k,Z}(m)$ applies a qubitcontrolled Displacement gate to the *k*-th qumode, with D(m) for $|0\rangle$ and D(-m) for $|1\rangle$ with respect to the state of the first qubit, this exactly implements $e^{2i\alpha^2 Z}$.

Our new multi-Pauli exponential in Equation (9) makes use of a multi-qubit controlled CD gate proposed by Liu *et al.*[28], as below.

$$CD^{(k,P_1P_2\cdots P_n)}(\alpha) = U_{seq}^{\dagger}D(i^n\alpha)U_{seq} - e^{(P_1P_2\cdots P_n)(\alpha a_k^{\dagger} - \alpha^* a_k)}$$
(11)

where $U_{seq} = \prod_{j=1}^{n} e^{i\pi P_{j}a_{k}^{\dagger}a_{k}/2}$ is a sequence of Control Parity gates (in Table 2) conjugated by Clifford gates, with the *j*-th qubit controlling the *k*-th qumode. It acts as a Control Parity gate for $P_{j} = Z$. For $P_{j} = X$, the Control Parity gate must be conjugated with Hadamard (*H*) gates, and for $P_{j} = Y$, with *H* and S^{\dagger} gates.

Essentially, this method combines unconditional Displacement and Control Parity gates to construct an arbitrary multi-qubit Paulistring term, requiring only a single ancillary qumode. The approach is **agnostic to the state of the ancilla qumode**, with the ancilla qumode returning to its initial state after the operation. This flexibility supports **efficient resource allocation** and **specific optimization goals**, benefiting mapping and routing.

The example in Fig. 5 illustrates how an ancilla qumode travels in the physical circuit, visiting every qubit A, B, \dots, N , to implement a CD gate for all qubits involved in the Pauli-string.



Figure 5: An ancilla qumode interacts with every qubit on the path from A to N via Control Parity (CP) gates, performs an unconditional Displacement gate, and then travels back to A's location while performing CP gates. This implements a CD gate in Equation 11. Note that the ancilla qumode can be any qumode, and the CP gates can run in any order. We just pick the qumode below qubit A to illustrate this idea.

Note that conjugating the Displacement gate with conjugated CP gates, or applying it to a polynomial of annihilation and creation

operators, yields multi-qubit controlled qumode gates. These form a series of sub-rules, which are also included in our rule database (Table 3) for conciseness.

3.3 DSL Design and Implementation

After level-1 compilation, Hamiltonians are decomposed into sequences of basic gates from the hybrid CV-DV gate set. Level-2 compilation then applies hardware constraints to generate physical circuits. To streamline this process, we introduce **CVDV-QASM**, an OpenQASM-like DSL that integrates Pauli-strings and CV-DV gate sequences. Our mapping and routing stage parses CVDV-QASM into quantum circuits. Syntax support for Pauli-string terms like "pauli($\pi/4$) YYZI" enables CP gate commutation and qumode allocation flexibility during physical circuit mapping while preserving semantics and providing crucial information to the lower compilation stack. An example is shown in Fig. 6:

```
// Pauli String with Parameter
pauli(pi/4) YIYZXXIIIIIII;
pauli(pi/4) XZYZXYIIIIIII;
// Phase Space Rotation Gate
R(pi/4) qm[1];
R(pi/4) qm[2];
// Control Displacement Gate
CD(pi/4) q[2], qm[1];
CD(pi/4) q[3], qm[1];
// Displacement Gate
D(pi/4) qm[2];
D(pi/4) qm[2];
```

Figure 6: Hybrid CV-DV circuit DSL example. pauli represents the parameter of a Pauli String; R, CD, D represent the gate type; qm[i] and q[i] represent the qumode and qubit respectively. Pauli String sequence will be further decomposed into basic gates in the compiler output of physical circuits.

This DSL has a similar input format of Bosonic Qiskit [40], except that our DSL supports Pauli string representation. As described before, Pauli strings are decomposed with an ancilla qumode, while hardware constraints, such as qumode interactions, are managed by inserting SWAP gates. The final circuit after our mapping and routing consists only of basic CV-DV gates and not Pauli-string gates. Detailed discussions on hardware constraints are in Section 3.4.

3.4 Tackling Limited Connectivity Constraints

In hybrid CV-DV architectures shown in Fig. 1, qumodes have limited connectivity and interact with qubits, adding complexity to multi-qumode interactions. While a SWAP operation in DV systems typically requires 3 CNOT gates, in CV systems, it begins with a Beam-Splitter gate $BS(\pi, 0)$ with parameters $\theta = \pi$ and $\varphi = 0$:

$$BS(\pi,0) |\Phi_a,\Phi_b\rangle = e^{-i\frac{\pi}{2}(a^{\dagger}a+b^{\dagger}b)} |\Phi_b,\Phi_a\rangle$$
(12)

To cancel the phase change from the Beam-Splitter (BS) gate, two Phase Space rotation gates, $e^{-i\frac{\pi}{2}\hat{n}_a}$ and $e^{-i\frac{\pi}{2}\hat{n}_b}$, are added. Thus, a **qumode SWAP gate** consists of **one Beam-Splitter (BS) gate** and **two Phase Space rotation gates**, with the latter having very low latency compared with a BS gate. This qumode SWAP primitive enables qumode movement for interactions with other qumodes or qubits. For qubit interactions involving qumodes, the qumodes can be moved to qubits. Multi-qubit interactions require primitives like Controlled Parity gates, discussed in Section 3.2.

In physical implementations, connectivity constraints between qumodes, qubits, and qumode-qubit pairs can be summarized as the following three mapping challenges:

- Qumode-qumode Mapping. Interactions are limited to adjacent qumodes. For non-adjacent qumodes, qumode SWAP gates are used, with routing optimized to minimize such SWAPs.
- **Qubit-qumode Mapping.** Each qumode interacts only with its associated qubit. For interactions with other qumodes, adjacency is established by moving qumodes, similar to qumode-qumode mapping.
- **Qubit-qubit Mapping.** Qubits interact indirectly via an ancilla qumode, which is moved between qubits to mediate interactions and complete gate operations.

3.4.1 Optimized Ancilla Qumode Routing for Qubit-qubit Interactions. As direct qubit-qubit interactions are limited by hardware constraints, we use ancilla qumodes to mediate interactions via phase kickback (Equation 10). The main challenge is to find the ancilla qumode and optimize its path to interact with target qubits efficiently. This Optimized Ancilla Qumode Routing Problem seeks to minimize path costs, measured by qumode SWAP operations, by finding the shortest path for a specific ancilla qumode to visit all qubits in $S \subseteq V$ on an undirected graph G = (V, E).

Fig. 7 compares an arbitrary alphabetical routing path with an optimized ancilla qumode routing path, highlighting its significant impact on the overall SWAP cost:



Figure 7: In the Optimized Ancilla Qumode Routing example, the red path represents alphabetical routing $(A \rightarrow B \rightarrow C \rightarrow D)$ with a SWAP cost of 4. The blue path shows optimized routing $(D \rightarrow C \rightarrow A \rightarrow B)$, using the Qumode at D as the ancilla, reducing the SWAP cost to 3.

The Optimized Ancilla Qumode Routing Problem can be reformulated as a relaxed Hamiltonian Path Problem, similar to a modified *Traveling Salesman Problem* (TSP). Unlike the closed-path TSP, this problem allows revisiting vertices and does not require returning to the starting vertex.

We construct a complete graph G' = (V, E') with the same vertex set *V* as the original graph *G*. Each edge in *E'* is weighted by w_{ij} , the shortest path cost between v_i and v_j in *G*, mapped to the

edge sequence in G. As shortest paths in G are polynomial-time computable, constructing G' is also polynomial in complexity.

Given the NP-hardness of the Hamiltonian Path Problem, we propose a multi-level solution strategy for scalability, as heuristic solutions can be more practical for larger problem instances.

We use *Christofides Algorithm*, which constructs a minimum spanning tree, finds a minimum-weight perfect matching among the odd-degree vertices, and combines them to form an Eulerian circuit, which can guarantee a near-optimal solution with polynomial time complexity, making it suitable for large-scale qumode routing scenarios. However, we also explore the other Heuristicbased algorithms, such as *Threshold Accepting Algorithm*, which can perform better in large-scale Pauli Strings compilation.

Also, for larger-scale problems, *Heuristic Algorithms* can provide **efficient** solutions, also making them well-suited for **specific physical circuits optimization goal** in Hybrid CV-DV systems to provide high extensibility and scalability.

Dynamic Oubit Floating: While the only native SWAP gates are for qumodes when addressing the connectivity constraints. Qubit movement can also be considered. As discussed by Liu et al.[28], CNOT gates can be implemented in the CV-DV architecture where qubits are not directly connected. We can essentially implement a qubit-qubit SWAP using 3 CNOT gates. However, it would require the usage of ancilla qumode, as the qubits are indirectly interacting via the ancilla gumodes. However, since gubit-gubit SWAP is not native, it takes 12 control displacement gates and 12 gumode-SWAP gates. It is almost 24 times the cost of the beamsplitter (qumode-SWAP) gate. It could only be useful when there are frequent interactions on a set of qubits, while the set of qubits is scattered very far from each other. This depends on the active-qubit pattern in the Pauli-string terms and the gubit-controlled gumode gates. Although it may not be useful in some applications, we still integrate this design into our compiler. We calculate the average shortest distance for active qubits within a Pauli-string, and if it exceeds a certain threshold, we trigger a clustering procedure and move qubits into a connected component. It offers more flexibility for specific long-range interactions. This is as if qubits can float around rather than being at their fixed locations. We name this approach as floating qubit.

Given hardware constraints and optimization flexibility, using an ancilla qumode or using qubit floating for multi-qubit gate synthesis offers three key advantages: **Resource Flexibility**: Any available qumode can serve as the ancilla without requiring a specific initial state, enabling efficient resource utilization. **Operational Efficiency**: The ancilla qumode can be reused across multiple gates, with Controlled Parity gates applied in any order before and after Displacement gates. **Optimization Potential:** Ancilla selection, Ancilla/Qubit routing, and task assignment can be optimized for resource reuse and circuit partitioning.

3.4.2 Mapping and Routing for General Case. Now we handle the case when the CVDV-QASM representation contains both Pauli strings and other CV-DV basis gates. Pauli-strings are processed separately. The general idea is to keep a working frontier – the set of gates (Pauli-string gates) whose dependence has been resolved, as well as a scheduled_worklist data structure containing gates that have already been scheduled so far. Then we repeat the following.

- (1) Execute any gate, including Pauli-string gates that are executable from the frontier. We prioritize the scheduling of Pauli-string gates because, by Trotterization, each Pauli-string gate (block) can run in any order. We use different ranking functions to choose the set of Pauli-strings to schedule next, and the ranking function either prioritizes the non-identity qubit number or the time stamp of the latest finished gate in that set of active qubits, to improve the parallelism. Then for the rest of the CV-DV basis gates, they are executable if their involved qubits/qumodes are connected. Once we schedule a sequence of gates, we add them to the scheduled_worklist.and update the frontier.
- (2) For the rest of the gates in the frontier, pick SWAPs that will help at least one gate in the frontier. Rank all the possible SWAPs. Then select one SWAP and add it to the scheduled_worklist.
- (3) Go back to Step 1 or terminate if there is no gate in the work frontier.

Once the program completes, the scheduled_worklist implies the order of the compiled physical gates. For the cost function of ranking candidate SWAP gates, we use a method similar to the Sabre qubit mapper [24]. For the cost function of ranking Pauli-strings, as mentioned above, we use either the minimum-active-qubit-number metric or the minimum-total-depth metric (sum up the depths of all active qubits in that Pauli-string) to parallelize as much as possible.

We also design a specific coupling map data structure for hybrid CV-DV systems. Currently, there are three versions of hybrid CV-DV quantum processors: superconducting, trapped ion, and neutral atom. We do not consider the neutral atom architecture as it does not allow connectivity among qubits. Our coupling map can support both superconducting and trapped ion CV-DV architecture.

4 Evaluation

4.1 Experimental Setup

Our evaluation focuses on two key aspects: Pauli-string Hamiltonian (qubit-only) in a hybrid CV-DV architecture and general compilation support for Hamiltonian simulation. First, we analyze the performance of compiling qubit-only Pauli-string Hamiltonians within the hybrid CV-DV architecture, evaluating our solutions for generating multi-qubit interactions via ancilla qumode(s). Second, we demonstrate, for the first time, the complete synthesis and compilation of given Hamiltonian models that may consist of both Fermions and Bosons, or just Bosons. This is for evaluating both the gate synthesis component and the hardware component.

Metrics. We employ three primary metrics to assess the compiler's performance: **1-op Gate**, **2-op Gate**, and **Depth**. The **1-op Gate** and **2-op Gate** metrics count the number of single-operand and two-operand gates, respectively, reflecting interactions such as qubit-qumode or qumode-qumode operations. The **Depth** metric measures the number of layers of quantum gates after compilation, representing the complexity of the compiled circuit. Using data from Table 2, we approximate the cost of a 2-op gate as equivalent to 20x that of a 1-op Gate and set 1-op gate latency to 1 when evaluating depth. For general Hamiltonian simulation compilation, the metric depth is for the final physical circuits that satisfy connectivity constraints rather than for the logical circuit. **Benchmarks.** We select benchmarks of various sizes and applications from practical, real-world Hamiltonian models. For the fermion-only Hamiltonian, using the PySCF [41] software package, we constructed the Hamiltonians for seven distinct molecules: LiH, BeH₂, ethylene, NH₃, C₂, N₂, and H₂O, under minimal basis STO-3G.

We also consider different spin-orbital-to-electron ratios, resulting in 20 benchmarks that comprise a total of 600 to 1900 Pauli strings. For example, $H_2O(8,12)$ and $H_2O(10,14)$ represent water molecules with 8 electrons and 12 spin orbitals, and 10 electrons and 14 spin orbitals, respectively. The size of the molecular system affects the overall complexity of the simulation, and different encoding schemes can lead to variations in the number of required operations and the resulting circuit depth.

We also include six general models, such as the Kerr nonlinear oscillator Hamiltonian, the \mathbb{Z}_2 -Higgs model, the Bose-Hubbard model, the Hubbard-Holstein model, the Heisenberg model, and the electronic-vibration coupling Hamiltonians. They could include interactions of boson matters or fermion-boson matters.

For each of the Hamiltonians, the fermion creation/annihilation operators must first be converted into qubit operators, in order to convert them into the form in Equation 1 before we perform further synthesis and scheduling. We employ both Jordan-Wigner [19] (JW) and Bravyi-Kitaev [2] (BK) encoding schemes to map fermionic operators to qubit operators [44]. JW is highly regular, but its operator length grows linearly with the system size N(O(N)). In contrast, the Bravyi-Kitaev encoding employs a more intricate tree-based structure to store parity information, thereby reducing the Pauli-weight, but with less regularity.

The Kerr nonlinear oscillator Hamiltonian (H_1) is a single-qumode Hamiltonian that includes a Kerr nonlinear term, adding complexity to its decomposition and compilation:

$$H_1 = \omega a^{\dagger} a + \frac{\kappa}{2} (a^{\dagger})^2 a^2.$$
 (13)

The \mathbb{Z}_2 -Higgs model (H_2) is a hybrid Hamiltonian featuring multimode interactions, including qumode-qumode and qubit-qubit interactions, making it representative of hybrid CV-DV systems:

$$H_2 = -g \sum_{i=1}^{L-1} X_{i,i+1} + U \sum_{i=1}^{L} \hat{n}_i^2 - J \sum_{i=1}^{L-1} \left(\hat{a}_i^{\dagger} Z_{i,i+1} \hat{a}_{i+1} + \text{h.c.} \right), \quad (14)$$

where $\hat{n}_i = a_i^{\mathsf{T}} a_i$ is the bosonic number operator, and $X_{i,i+1}$ and $Z_{i,i+1}$ are Pauli X and Z operators linking sites *i* and *i* + 1, respectively, each can be represented using one qubit operator. The term "h.c." represents the hermitian conjugate of the term right before the addition sign.

The Bose-Hubbard model (H_3) is a lattice model whose structure varies with different lattice configurations, making it suitable for higher-dimensional physical systems. Also, it is a pure Hamiltonian with only qumode-qumode interaction (may need ancilla qubits for higher order annihilation and creation operators), making it representative of pure qumode Hamiltonian systems:

$$H_3 = -t \sum_{i,j} (b_i^{\dagger} b_j + b_j^{\dagger} b_i) + \frac{U}{2} \sum_i \hat{n}_i (\hat{n}_i - 1) - \mu \sum_i \hat{n}_i, \quad (15)$$

where $\hat{n}_i = b_i^{\dagger} b_i$, and b_i^{\dagger} and b_i are creation and annihilation operators.

The Hubbard-Holstein model (H_4) involves both fermionic and bosonic operators, adding complexity due to multiple mappings of fermionic operators to qubit Pauli operators. Here, b_i^{\dagger} (b_i) denotes the bosonic creation (annihilation) operator at site *i*, while $c_{i,\sigma}^{\dagger}$ ($c_{i,\sigma}$) represents the fermionic counterpart. The number operator $\hat{n}_{i,\sigma} = c_{i,\sigma}^{\dagger} c_{i,\sigma}$ counts fermions with spin $\sigma = \uparrow, \downarrow$ at site *i*. We applied Jordan-Wigner encoding, but, due to space constraints, do not expand it into Pauli and Bosonic annihilation operator form:

$$H_4 = -t\sum_{i,j,\sigma} c^{\dagger}_{i,\sigma} c_{j,\sigma} + U\sum_i \hat{n}_{i,\uparrow} \hat{n}_{i,\downarrow} + \sum_i b^{\dagger}_i b_i + g\sum_{i,\sigma} \hat{n}_{i,\sigma} (b^{\dagger}_i + b_i),$$
(16)

The Hamiltonian H_5 describes the interaction between discrete electronic states (qubits) and vibrational modes (qumodes), capturing complex electron-phonon coupling effects in molecular systems. Such models are essential for simulating quantum dynamics in onedimensional (1D) chromophore arrays, where electronic excitations are strongly coupled to vibrational environments. Applications span light-harvesting complexes, organic semiconductors, and photosynthetic systems [45]. The full Hamiltonian is decomposed as:

$$H_5 = \sum_{\gamma=1}^{N} \left[H_0^{(\gamma)} + H_1^{(\gamma)} + H_2^{(\gamma)} \right]$$
(17)

Here, $H_0^{(\gamma)}$ represents local non-interacting terms, $H_1^{(\gamma)}$ captures dispersive coupling between qubits and modes, and $H_2^{(\gamma)}$ includes inter-chromophore interactions and vibrationally modulated couplings. Each component is defined below:

$$H_{0}^{(\gamma)} = \omega_{\gamma_{0}} b_{\gamma_{0}}^{\dagger} b_{\gamma_{0}} + \omega_{\gamma_{1}} b_{\gamma_{1}}^{\dagger} b_{\gamma_{1}} - \frac{\omega_{q\gamma_{0}}}{2} \sigma_{\gamma_{0}}^{z}$$
(18)

$$H_1^{(\gamma)} = -\frac{\chi_{\gamma_0}}{2} b_{\gamma_0}^{\dagger} b_{\gamma_0} \sigma_{\gamma_0}^z + \frac{g_{cd,\gamma_0}}{2} (b_{\gamma_0} + b_{\gamma_0}^{\dagger}) \sigma_{\gamma_0}^z$$
(19)

$$\begin{split} H_{2}^{(\gamma)} &= \frac{g_{cd,\gamma_{1}}}{2} (b_{\gamma_{1}} + b_{\gamma_{1}}^{\dagger}) \sigma_{\gamma_{0}}^{z} \\ &+ \frac{g_{\gamma_{0},(\gamma-1)_{0}}}{4} \left(\sigma_{\gamma_{0}}^{x} \sigma_{(\gamma-1)_{0}}^{x} + \sigma_{\gamma_{0}}^{y} \sigma_{(\gamma-1)_{0}}^{y} \right) \\ &+ \frac{g_{\gamma_{0},(\gamma+1)_{0}}}{4} \left(\sigma_{\gamma_{0}}^{x} \sigma_{(\gamma+1)_{0}}^{x} + \sigma_{\gamma_{0}}^{y} \sigma_{(\gamma+1)_{0}}^{y} \right) \\ &+ \frac{g_{\gamma_{0},(\gamma-1)_{0},\gamma_{1}}}{4} \left(\sigma_{\gamma_{0}}^{x} \sigma_{(\gamma-1)_{0}}^{x} + \sigma_{\gamma_{0}}^{y} \sigma_{(\gamma-1)_{0}}^{y} \right) (b_{\gamma_{1}} + b_{\gamma_{1}}^{\dagger}) \\ &+ \frac{g_{\gamma_{0},(\gamma+1)_{0},\gamma_{1}}}{4} \left(\sigma_{\gamma_{0}}^{x} \sigma_{(\gamma+1)_{0}}^{x} + \sigma_{\gamma_{0}}^{y} \sigma_{(\gamma+1)_{0}}^{y} \right) (b_{\gamma_{1}} + b_{\gamma_{1}}^{\dagger}) \end{split}$$
(20)

The Heisenberg model (H_6) describes qubit-qubit interactions in a spin chain or lattice system. Here, H_6 represents a chain of Nqubits coupled via exchange interactions in the x, y, z directions. The absence of qubit-qumode and qumode-qumode interactions simplifies the model, making it ideal for understanding pure qubit dynamics. The Heisenberg Hamiltonian has broad applications in condensed matter physics, quantum magnetism, and quantum computing, serving as a foundation for entanglement, spin transport, and quantum phase transitions, which is given by:

$$H_6 = -\frac{1}{2} \sum_{j=1}^{N} (J_x X_j X_{j+1} + J_y Y_j Y_{j+1} + J_z Z_j Z_{j+1} + hZ_j)$$
(21)

These Hamiltonians exemplify hybrid CV-DV systems and present significant compilation challenges. For the first time, we have fully decomposed, synthesized, and compiled their physical circuits for hybrid CV-DV quantum computers.

Baselines and Hardware Coupling Maps. For multi-qubit interactions on a hybrid CV-DV architecture, since there is a prior compiler study, we compare different versions of our implementation, including the Christofides algorithm and the Threshold Accepting Heuristic-based approach, to analyze their effectiveness in handling such Hamiltonian Simulation scenarios. Additionally, we investigate the use of Floating Qubits within the hybrid CV-DV architecture, exploring innovative compilation strategies under this framework. Finally, we assess the scalability and extensibility of our methods, offering valuable insights for future research and development. For the coupling map, we adopt a lattice structure for qumode connectivity as shown in Fig. 1, each qumode is connected to one qubit. Qubits are not connected.

4.2 Pauli String Synthesis using Ancilla Qumode and Floating Qubit

For Pauli string compilation, hybrid CV-DV systems introduce challenges in handling qubit-qubit interactions where direct qubit compilation is unavailable. This necessitates adaptations in our framework. One approach mediates multi-qubit interactions through gumodes, addressing the Optimized Ancilla Qumode Routing Problem. We explored two routing strategies: the Christofides Algorithm and the Threshold Accepting Algorithm, both efficient heuristics for the common traveling salesman problem (TSP). Additionally, we investigated the Floating Qubit method as an alternative architectural solution for hybrid systems. Detailed results are in Table 4. Our benchmarks involve complex and representative Hamiltonians. While the Christofides Algorithm provides an efficient polynomialtime approximation, results indicate that the Threshold Accepting Algorithm achieves 3-7% better optimization, reducing circuit depth by an average of 4.8%. This demonstrates the effectiveness of heuristics in addressing complex compilation problems. Further improvements might be possible by incorporating additional heuristic objectives, such as architecture-specific optimizations.

We also evaluated the Floating Qubit method but found it largely ineffective across our benchmark suite, achieving optimal results in only two cases. In most benchmarks, it increased the average circuit depth by 6%. This is attributed to the significant cost of Floating Qubits, which can reach up to 24x, making the depth minimization goal more challenging. However, flexible Floating Qubit strategies may still be useful in scenarios requiring high adaptability, such as resource reuse or circuit partitioning.

Table 4 shows that JW outperforms BK in physical mapping and routing. It is because JW's Pauli strings have more regularity from block to block and hence better locality, thereby reducing the number of additional routing gates. Therefore, we use JW encoding for fermion operators in all subsequent experiments.

ISCA '25, June 21–25, 2025, Tokyo, JapanZihan Chen, Jiakang Li, Minghao Guo, Henry Chen, Zirui Li, Joel Bierman, Yipeng Huang, Huiyang Zhou, Yuan Liu, and Eddy Z. Zhang

Malaanlaa	Manning	# Pauli Strings	Christofide	es Algorithm(Qumode SWAP)	Threshold(Qumode SWAP)			Threshold(Floating Qubit)		
Molecules	Mapping		1-op Gate	2-op Gate	Duration	1-op Gate	2-op Gate	Duration	1-op Gate	2-op Gate	Duration
1:11 (4 12)	BK	631	41472	30920	546597	39192	29780	518510 (5.14%)	39360	29864	520580 (4.76%)
LIII (4, 12)	JW	631	36344	27384	458441	33888	26156	432246 (5.71%)	34352	26472	444074 (3.13%)
BoH2 (6, 14)	BK	666	49520	35608	628278	46576	34136	587180 (6.54%)	48210	35184	644966 (-2.66%)
Del12 (0, 14)	JW	666	44428	33652	562822	42092	32484	535692 (4.82%)	43384	33508	585151 (-3.97%)
Ethylana (10, 16)	BK	789	54976	40688	678566	52368	39384	650664 (4.11%)	57602	43408	769017 (-13.33%)
Ethylene (10, 10)	JW	789	57312	43496	712622	54304	41992	681511 (4.37%)	58066	44608	796156 (-11.72%)
NH3 (8, 12)	BK	915	61564	45592	810401	58164	43892	773912 (4.5%)	58268	43944	769401 (5.06%)
1115 (0, 12)	JW	915	53680	39960	656519	50336	38288	615952 (6.18%)	53002	39684	699905 (-6.61%)
$C_{2}(10, 16)$	BK	1177	87200	63768	1057647	82584	61460	1011085 (4.4%)	85998	64196	1111745 (-5.11%)
02 (10, 10)	JW	1177	88960	66968	1100289	83832	64404	1057411 (3.9%)	87568	67028	1190209 (-8.17%)
N2 (10, 16)	BK	1177	88792	65004	1074193	84488	62852	1039060 (3.27%)	87078	65176	1150102 (-7.07%)
142 (10, 10)	JW	1177	88752	67056	1106544	84232	64796	1063242 (3.91%)	89512	68528	1216367 (-9.92%)
H2O (8, 12)	BK	1219	80004	59516	1048788	75860	57444	1001186 (4.54%)	75940	57484	997611 (4.88%)
1120 (0, 12)	JW	1219	70200	52696	862689	66336	50764	817900 (5.19%)	70540	52908	937287 (-8.65%)
$H_{2O}(10, 14)$	BK	1654	122864	88204	1543297	116576	85060	1469419 (4.79%)	122572	91040	1620031 (-4.97%)
H2O (10, 14)	JW	1654	111332	83068	1377143	105180	79992	1319874 (4.16%)	113576	84904	1553009 (-12.77%)
NH3 (8, 14)	BK	1734	131716	94452	1657461	124404	90796	1547533 (6.63%)	129320	93968	1700487 (-2.6%)
	JW	1734	118260	88428	1484725	111620	85108	1395552 (6.01%)	120348	90228	1622020 (-9.25%)
$C_{2}(12, 18)$	BK	1884	175612	126176	2212185	167756	122248	2141333 (3.2%)	171994	131024	2381938 (-7.67%)
C2 (12, 18)	JW	1884	155140	117292	1937459	147340	113392	1857982 (4.1%)	163376	127668	2361642 (-21.89%)

Table 4: Compilation latency results for multi-Pauli exponentials. Gate latencies are assigned according to Table 2, where single-qubit (1-op) gates have a latency of 1 unit (20 nanoseconds) and two-qubit (2-op) gates incur a latency of 20 units. The Christofides algorithm is used as the performance baseline. Christofides-based routing is used as the baseline. Duration results show absolute latency and percentage reduction relative to the baseline.

4.3 General Hamiltonian Simulation Compilation

4.3.1 End-to-end Compilation. For the evaluation of general Hamiltonian simulation, we compiled a diverse set of six Hamiltonian models. These models are not only representative in hybrid CV-DV systems but also pose significant complexity during compilation, making them ideal for evaluating our general-purpose compiler. We further tested performance across varying lattice sizes to assess scalability. The detailed results are shown in Table 5.

Table 5 shows the results at different stages of compilation. The number of Pauli strings and hybrid CV-DV gates is from the intermediate representation using our domain-specific language, **CVDV-QASM**. The "Total Gate Count" and "Duration" are from the final compiled circuits after decomposition, mapping, and routing.

These representative Hamiltonian models highlight the compiler's capability to support practical quantum computing applications, offering a valuable framework for benchmarking and reference in future research. Key considerations included successful decomposition, gate count, and acceptable depth metrics. Based on our Rule-Based Recursive Template Matching Mechanism, the framework can be further optimized for architecture-specific constraints or fidelity improvements, providing a foundation for future advancements in Hamiltonian simulation compilation.

4.3.2 Hit Rate Analysis for Decomposition Rules. We conducted a detailed decomposition analysis by measuring the "hit rate" of each rule, quantifying its relative importance during synthesis. Each unique term in a summation Σ was counted once per Hamiltonian model, regardless of index variations. For complex expressions

(e.g., $a^{\dagger}a^{\dagger}aa$, $a^{\dagger}aa^{\dagger}a$), which often require multiple recursive applications, we normalized rule usage per Hamiltonian model. This normalized frequency is defined as the rule's "hit rate."

Rule 16, which corresponds to native basis gate synthesis, appears most frequently and nearly all decomposition paths eventually terminate at a basis gate. It accounts for 68.30% of rule applications along successful paths, and 70.88% overall. To better assess the relative contribution of other rules, we exclude Rule 16 and report the normalized hit rates for Rules 1-15 in Table 6.

Rules 1 and 2 (Trotter and BCH decompositions) are heavily used during intermediate steps, together contributing over 28.80% of the successful hit rate. Rules 14 and 15, which address multi-Pauli exponentials and multi-qubit-controlled qumode displacements, also show high hit rates, highlighting the importance of qubitqumode interactions in certain Hamiltonian model. Rules 11 and 12 frequently appear in the successful paths as well, enabling bosonic operators to be decomposed into fine-grained gate-synthesizable forms via block-encoding templates, jointly contributing 10.80% to the successful hit rate.

4.3.3 Compilation Time Analysis. We evaluated the compilation time of our framework using JW-mapped Pauli strings, considering the scale of Pauli string compilation for each molecule. Results are shown in Table 7. Measured in seconds, the compilation time covers the entire process, from Pauli string decomposition to physical circuit synthesis. While it generally increases with the number of Pauli strings, C2(12, 18) shows a significantly higher time due to the increased number of electrons and spin-orbitals, leading to a more complex system than just longer Pauli strings.

Hamiltonians	# Qubits	# Qumodes	# Pauli Strings (in IR)	# Gates (in IR)	Final Gate Count	Duration
Kerr Nonlinear Hamiltonian	19 (ancilla)	1	0	409	595	3032
	20	20	19	419	696	943
Z_2 -Higgs Model	40	40	39	839	1472	1797
	60	60	59	1259	2263	2721
	20 (ancilla)	20	0	820	2935	15353
Bose-Hubbard Model	40 (ancilla)	40	0	2440	15313	62548
	60 (ancilla)	60	0	4860	36705	155872
Hubbard-Holstein Model	40	20	3100	100	688011	4509157
	20	40	96	1392	5635	28551
Electronic-Vibration Coupling	40	80	196	2852	12655	54952
	60	120	296	4312	19684	81729
Heisenberg Model	20	0	77	0	2218	4122
	40	0	157	0	4676	4398
	60	0	237	0	7200	7074

Table 5: Compilation results for general Hamiltonian simulations. For each model, the table reports (a) the number of qubits and qumodes utilized, (b) the component counts in intermediate representation, i.e., the number of Pauli strings and hybrid CV-DV gates, using our domain-specific language, CVDV-QASM, and (c) the "Total Gate Count" and "Duration" of the final compiled circuit. Operation durations are estimated by assigning 1 time unit (20 nanoseconds) to each single-qubit or single-qumode gate and 20 time units to each hybrid CV-DV gate and multi-qumode gate, according to Table 2.

Rule	Success	Total	Rule	Success	Total	Rule	Success	Total
No. 1	9.50%	3.83%	No. 6	1.35%	0.97%	No. 11	0.34%	4.47%
No. 2	19.33%	23.77%	No. 7	1.35%	0.57%	No. 12	5.40%	12.78%
No. 3	0.00%	0.00%	No. 8	8.15%	1.47%	No. 13	5.40%	11.48%
No. 4	8.15%	2.94%	No. 9	0.00%	1.47%	No. 14	15.77%	17.17%
No. 5	8.15%	1.47%	No. 10	1.35%	0.45%	No. 15	15.77%	17.17%

Table 6: Normalized hit rates for decomposition Rules 1-15. Each entry includes both the *Success Hit Rate* (calculated over successfully decomposed paths) and the *Total Hit Rate* (all search attempts, including failures).

Molecule	# Pauli Strings	Time(s)	Molecule	# Pauli Strings	Time(s)
LiH (4, 12)	631	20.39	BeH2 (6, 14)	666	33.33
Ethylene (10, 16)	789	67.21	NH3 (8, 12)	915	59.95
C2 (10, 16)	1177	184.49	N2 (10, 16)	1177	205.48
H2O (8, 12)	1219	100.33	H2O (10, 14)	1654	379.83
NH3 (8, 14)	1734	485.46	C2 (12, 18)	1884	1,152.00

Table 7: Compilation Time Analysis

5 Related Work

Existing tools, including Bosonic Qiskit [40], StrawberryField [21], Perceval [49], and Bosehedral [33], offer preliminary support for programming, simulating, and composing circuits for either domainspecific applications such as Gaussian Boson Sampling (GBS) or general bosonic circuits. Perceval allows users to build linear optical circuits from a collection of pre-defined components. Strawberry Fields and Bosonic Qikist provide Python libraries involving basic CV-DV gates, as well as simulate the programs written with these basic libraries. However, none of this provides support for the simulation of the hybrid CV-DV Hamiltonian.

While Hamiltonian simulation on DV systems has been extensively studied [18, 25, 26], compilation support for hybrid CV-DV systems remains underexplored. The unique properties of Bosonic hardware mean that Fermion-Boson interactions have not been thoroughly investigated. This involves converting Hamiltonian simulation algorithms into basis gates for both qubits and qumodes, implementing multi-qubit operations, and optimizing qubit-qumode mapping and routing to address connectivity and error mitigation challenges. Our work bridges this gap.

6 Conclusions

Our work introduces Genesis, the first comprehensive compilation framework for Hamiltonian simulation on hybrid CV-DV quantum processors. By leveraging a two-stage compilation approach: (1) decomposing hybrid Hamiltonians into universal basis gates, and (2) mapping them to hardware-constrained circuits, we enable efficient simulation of complex physical systems. Our tool has successfully compiled important Hamiltonians, including the Bose-Hubbard model, \mathbb{Z}_2 -Higgs model, Hubbard-Holstein model, and electronvibration coupling Hamiltonians critical in domains like quantum field theory, condensed matter physics, and quantum chemistry. We also provide a domain-specific language (DSL) design to support Hamiltonian simulation on a hybrid CV-DV architecture.

Acknowledgments

We thank the anonymous reviewers for their constructive and helpful feedback. This work is supported in part by grants from the Rutgers Research Council, NSF, and DOE. In particular, Y.L., H.Z., and J.B. are supported by the U.S. Department of Energy (DOE), Office of Science, Office of Advanced Scientific Computing Research (ASCR), under Award Number DE-SC0025384. Z.C., J.L., M.G., H.C., and E.Z. are supported by the DOE Award DE-SC0025563, the NSF Award CCF-2129872, and Rutgers Research Council Grant. J.B., Y.L., and H.Z. are supported in part by NSF OSI-2410675 (with a subcontract to NC State University from Duke University). H.Z. is also supported in part by NSF grants PHY-2325080 and OMA-2120757 (with a subcontract to NC State University from the University of Maryland). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of our sponsors.

References

- Junaid Aftab, Dong An, and Konstantina Trivisa. 2024. Multi-product Hamiltonian simulation with explicit commutator scaling. arXiv preprint arXiv:2403.08922 (2024).
- [2] Sergey B Bravyi and Alexei Yu Kitaev. 2002. Fermionic quantum computation. Annals of Physics 298, 1 (2002), 210–226.

ISCA '25, June 21-25, 2025, Tokyo, JaparZihan Chen, Jiakang Li, Minghao Guo, Henry Chen, Zirui Li, Joel Bierman, Yipeng Huang, Huiyang Zhou, Yuan Liu, and Eddy Z. Zhang

- [3] Colin D. Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M. Sage. 2019. Trapped-ion quantum computing: Progress and challenges. *Applied Physics Reviews* 6, 2 (May 2019). doi:10.1063/1.5088164
- [4] Philippe Campagne-Ibarcq, Alec Eickbusch, Steven Touzard, Evan Zalys-Geller, Nicholas E Frattini, Volodymyr V Sivak, Philip Reinhold, Shruti Puri, Shyam Shankar, Robert J Schoelkopf, et al. 2020. Quantum error correction of a qubit encoded in grid states of an oscillator. *Nature* 584, 7821 (2020), 368–372.
- [5] Benjamin J. Chapman, Stijn J. de Graaf, Sophia H. Xue, Yaxing Zhang, James Teoh, Jacob C. Curtis, Takahiro Tsunoda, Alec Eickbusch, Alexander P. Read, Akshay Koottandavida, Shantanu O. Mundhada, Luigi Frunzio, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf. 2023. A high on-off ratio beamsplitter interaction for gates on bosonically encoded qubits. arXiv:2212.11929 [quant-ph] https: //arxiv.org/abs/2212.11929
- [6] Eleanor Crane, Kevin C Smith, Teague Tomesh, Alec Eickbusch, John M Martyn, Stefan Kühn, Lena Funcke, Michael Austin DeMarco, Isaac L Chuang, Nathan Wiebe, et al. 2024. Hybrid Oscillator-Qubit Quantum Processors: Simulating Fermions, Bosons, and Gauge Fields. arXiv preprint arXiv:2409.03747 (2024).
- [7] Brennan de Neeve, Thanh Long Nguyen, Tanja Behrle, and Jonathan Home. 2020. Error correction of a logical grid state qubit by dissipative pumping. arXiv:2010.09681 [quant-ph] https://arxiv.org/abs/2010.09681
- [8] Brennan De Neeve, Thanh-Long Nguyen, Tanja Behrle, and Jonathan P Home. 2022. Error correction of a logical grid state qubit by dissipative pumping. *Nature Physics* 18, 3 (2022), 296–300.
- [9] Yu-Hao Deng, Yi-Chao Gu, Hua-Liang Liu, Si-Qiu Gong, Hao Su, Zhi-Jiong Zhang, Hao-Yang Tang, Meng-Hao Jia, Jia-Min Xu, Ming-Cheng Chen, Jian Qin, Li-Chao Peng, Jiarong Yan, Yi Hu, Jia Huang, Hao Li, Yuxuan Li, Yaojian Chen, Xiao Jiang, Lin Gan, Guangwen Yang, Lixing You, Li Li, Han-Sen Zhong, Hui Wang, Nai-Le Liu, Jelmer J. Renema, Chao-Yang Lu, and Jian-Wei Pan. 2023. Gaussian Boson Sampling with Pseudo-Photon-Number-Resolving Detectors and Quantum Computational Advantage. *Phys. Rev. Lett.* 131 (Oct 2023), 150601. Issue 15. doi:10.1103/PhysRevLett.131.150601
- [10] Yulong Dong, K Birgitta Whaley, and Lin Lin. 2022. A quantum hamiltonian simulation benchmark. npj Quantum Information 8, 1 (2022), 131.
- [11] Alec Eickbusch, Volodymyr Sivak, Andy Z. Ding, Salvatore S. Elder, Shantanu R. Jha, Jayameenakshi Venkatraman, Baptiste Royer, S. M. Girvin, Robert J. Schoelkopf, and Michel H. Devoret. 2022. Fast universal control of an oscillator with weak dispersive coupling to a qubit. *Nature Physics* 18, 12 (Oct. 2022), 1464–1469. doi:10.1038/s41567-022-01776-9
- [12] Richard P Feynman. 1982. Simulating physics with computers. International journal of theoretical physics 21 (1982), 467–488. Issue 6/7. doi:10.1007/BF02650179
- [13] Christa Flühmann and Jonathan P Home. 2020. Direct characteristic-function tomography of quantum states of the trapped-ion motional oscillator. *Physical review letters* 125, 4 (2020), 043602.
- [14] Christa Flühmann, Thanh Long Nguyen, Matteo Marinelli, Vlad Negnevitsky, Karan Mehta, and JP Home. 2019. Encoding a qubit in a trapped-ion mechanical oscillator. *Nature* 566, 7745 (2019), 513–517.
- [15] C. Flühmann, T. L. Nguyen, M. Marinelli, V. Negnevitsky, K. Mehta, and J. P. Home. 2019. Encoding a qubit in a trapped-ion mechanical oscillator. *Nature* 566, 7745 (Feb. 2019), 513–517. doi:10.1038/s41586-019-0960-6
- [16] Jeffrey M Gertler, Brian Baker, Juliang Li, Shruti Shirol, Jens Koch, and Chen Wang. 2021. Protecting a bosonic qubit with autonomous quantum error correction. *Nature* 590, 7845 (2021), 243–248.
- [17] P. C. Haljan, K.-A. Brickman, L. Deslauriers, P. J. Lee, and C. Monroe. 2005. Spin-Dependent Forces on Trapped Ions for Phase-Stable Quantum Gates and Entangled States of Spin and Motion. *Phys. Rev. Lett.* 94 (Apr 2005), 153602. Issue 15. doi:10.1103/PhysRevLett.94.153602
- [18] Yuwei Jin, Zirui Li, Fei Hua, Yanhao Chen, Henry Chen, Yipeng Huang, and Eddy Z. Zhang. 2023. Tetris: A compilation Framework for VQE Applications. arXiv:2309.01905 [quant-ph]
- [19] Pascual Jordan and Eugene Paul Wigner. 1993. Über das paulische äquivalenzverbot. Springer.
- [20] Christopher Kang, Micheline B Soley, Eleanor Crane, SM Girvin, and Nathan Wiebe. 2023. Leveraging hamiltonian simulation techniques to compile operations on bosonic devices. arXiv preprint arXiv:2303.15542 (2023).
- [21] Nathan Killoran, Josh Izaac, Nicolás Quesada, Ville Bergholm, Matthew Amy, and Christian Weedbrook. 2019. Strawberry Fields: A Software Platform for Photonic Quantum Computing. *Quantum* 3 (March 2019), 129. doi:10.22331/q-2019-03-11-129
- [22] Alexei Kitaev and William A Webb. 2008. Wavefunction preparation and resampling using a quantum computer. arXiv preprint arXiv:0801.0342 (2008).
- [23] Johannes Knörzer, Tao Shi, Eugene Demler, and J Ignacio Cirac. 2022. Spin-Holstein models in trapped-ion systems. *Physical Review Letters* 128, 12 (2022), 120404.
- [24] Gushu Li, Yufei Ding, and Yuan Xie. 2019. Tackling the qubit mapping problem for NISQ-era quantum devices. Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems (2019), 1001–1014.

- [25] Gushu Li, Yunong Shi, and Ali Javadi-Abhari. 2021. Software-Hardware Co-Optimization for Computational Chemistry on Superconducting Quantum Processors. In Proceedings of the 48th Annual International Symposium on Computer Architecture (Virtual Event, Spain) (ISCA '21). IEEE Press, 832–845. doi:10.1109/ ISCA52012.2021.00070
- [26] 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. Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (2022), 554–569. doi:10.1145/3503222.3507715
- [27] Langyu Li. 2024. Principal Trotter Observation Error with Truncated Commutators. arXiv preprint arXiv:2408.03891 (2024).
- [28] Yuan Liu, Shraddha Singh, Kevin C. Smith, Eleanor Crane, John M. Martyn, Alec Eickbusch, Alexander Schuckert, Richard D. Li, Jasmine Sinanan-Singh, Micheline B. Soley, Takahiro Tsunoda, Isaac L. Chuang, Nathan Wiebe, and Steven M. Girvin. 2024. Hybrid Oscillator-Qubit Quantum Processors: Instruction Set Architectures, Abstract Machine Models, and Applications. arXiv:2407.10381 [quantph] https://arxiv.org/abs/2407.10381
- [29] Seth Lloyd. 1996. Universal quantum simulators. Science 273, 5278 (Aug. 1996), 1073. https://www.proquest.com/docview/213562780/abstract/ 381415109DD14C66PQ/1
- [30] Yao Lu, Aniket Maiti, John WO Garmon, Suhas Ganjam, Yaxing Zhang, Jahan Claes, Luigi Frunzio, Steven M Girvin, and Robert J Schoelkopf. 2023. High-fidelity parametric beamsplitting with a parity-protected converter. *nature communications* 14, 1 (2023), 5767.
- [31] Yuwei Ma, Yuan Xu, Xianghao Mu, Weizhou Cai, Ling Hu, Weiting Wang, Xiaoxuan Pan, Haiyan Wang, YP Song, C-L Zou, et al. 2020. Error-transparent operations on a logical qubit protected by quantum error correction. *Nature Physics* 16, 8 (2020), 827–831.
- [32] Refik Mansuroglu, Felix Fischer, and Michael J Hartmann. 2023. Problem-specific classical optimization of Hamiltonian simulation. *Physical Review Research* 5, 4 (2023), 043035.
- [33] Nicolas Maring, Andreas Fyrillas, Mathias Pont, Edouard Ivanov, Petr Stepanov, Nico Margaria, William Hease, Anton Pishchagin, Thi Huong Au, Sébastien Boissier, Eric Bertasi, Aurélien Baert, Mario Valdivia, Marie Billard, Ozan Acar, Alexandre Brieussel, Rawad Mezher, Stephen C. Wein, Alexia Salavrakos, Patrick Sinnott, Dario A. Fioretto, Pierre-Emmanuel Emeriau, Nadia Belabas, Shane Mansfield, Pascale Senellart, Jean Senellart, and Niccolo Somaschi. 2023. A general-purpose single-photon-based quantum computing platform. arXiv:2306.00874 [quant-ph] https://arxiv.org/abs/2306.00874
- [34] Zhongchu Ni, Sai Li, Xiaowei Deng, Yanyan Cai, Libo Zhang, Weiting Wang, Zhen-Biao Yang, Haifeng Yu, Fei Yan, Song Liu, et al. 2023. Beating the break-even point with a discrete-variable-encoded logical qubit. *Nature* 616, 7955 (2023), 56–60.
- [35] Nissim Ofek, Andrei Petrenko, Reinier Heeres, Philip Reinhold, Zaki Leghtas, Brian Vlastakis, Yehan Liu, Luigi Frunzio, Steven M Girvin, Liang Jiang, et al. 2016. Extending the lifetime of a quantum bit with error correction in superconducting circuits. *Nature* 536, 7617 (2016), 441–445.
- [36] Jennifer Paykin, Albert T. Schmitz, Mohannad Ibrahim, Xin-Chuan Wu, and A. Y. Matsuura. 2023. PCOAST: A Pauli-based Quantum Circuit Optimization Framework. arXiv:2305.10966 [quant-ph]
- [37] Alexandru Petrescu, Hakan E Türeci, Alexey V Ustinov, and Ioan M Pop. 2018. Fluxon-based quantum simulation in circuit QED. *Physical Review B* 98, 17 (2018), 174505.
- [38] Adam L. Shaw, Pascal Scholl, Ran Finkelstein, Richard Bing-Shiun Tsai, Joonhee Choi, and Manuel Endres. 2024. Erasure-cooling, control, and hyperentanglement of motion in optical tweezers. arXiv:2311.15580 [quant-ph] https://arxiv.org/abs/2311.15580
- [39] V. V. Sivak, A. Eickbusch, B. Royer, S. Singh, I. Tsioutsios, S. Ganjam, A. Miano, B. L. Brock, A. Z. Ding, L. Frunzio, S. M. Girvin, R. J. Schoelkopf, and M. H. Devoret. 2023. Real-time quantum error correction beyond break-even. *Nature* 616, 7955 (March 2023), 50–55. doi:10.1038/s41586-023-05782-6
- [40] Timothy J Stavenger, Eleanor Crane, Kevin C Smith, Christopher T Kang, Steven M Girvin, and Nathan Wiebe. 2022. C2QA - Bosonic Qiskit. In 2022 IEEE High Performance Extreme Computing Conference (HPEC). 1–8. doi:10.1109/ HPEC55821.2022.9926318
- [41] Qiming Sun, Timothy C Berkelbach, Nick S Blunt, George H Booth, Sheng Guo, Zhendong Li, Junzi Liu, James McClain, Elvira R Sayfutyarova, Sandeep Sharma, et al. 2007. The python-based simulations of chemistry framework (pyscf). arXiv preprint arXiv:1701.08223 (2007).
- [42] James D. Teoh, Patrick Winkel, Harshvardhan K. Babla, Benjamin J. Chapman, Jahan Claes, Stijn J. de Graaf, John W. O. Garmon, William D. Kalfus, Yao Lu, Aniket Maiti, Kaavya Sahay, Neel Thakur, Takahiro Tsunoda, Sophia H. Xue, Luigi Frunzio, Steven M. Girvin, Shruti Puri, and Robert J. Schoelkopf. 2023. Dual-rail encoding with superconducting cavities. *Proceedings of the National Academy of Sciences* 120, 41 (2023), e2221736120. doi:10.1073/pnas.2221736120 arXiv:https://www.pnas.org/doi/pdf/10.1073/pnas.2221736120

- [43] G.S. Thekkadath, S. Sempere-Llagostera, B.A. Bell, R.B. Patel, M.S. Kim, and I.A. Walmsley. 2022. Experimental Demonstration of Gaussian Boson Sampling with Displacement. *PRX Quantum* 3 (May 2022), 020336. Issue 2. doi:10.1103/ PRXQuantum.3.020336
- [44] Jules Tilly, Hongxiang Chen, Shuxiang Cao, Dario Picozzi, Kanav Setia, Ying Li, Edward Grant, Leonard Wossnig, Ivan Rungger, George H. Booth, and Jonathan Tennyson. 2022. The Variational Quantum Eigensolver: A review of methods and best practices. *Physics Reports* 986 (nov 2022), 1–128. doi:10.1016/j.physrep. 2022.08.003
- [45] Nam P Vu, Daniel Dong, Xiaohan Dan, Ningyi Lyu, Victor Batista, and Yuan Liu. 2025. A Computational Framework for Simulations of Dissipative Non-Adiabatic Dynamics on Hybrid Oscillator-Qubit Quantum Devices. arXiv preprint arXiv:2502.17820 (2025).
- [46] Mattia Walschaers. 2021. Non-Gaussian Quantum States and Where to Find Them. PRX Quantum 2 (Sep 2021), 030204. Issue 3. doi:10.1103/PRXQuantum.2.030204
- [47] Christopher S Wang, Jacob C Curtis, Brian J Lester, Yaxing Zhang, Yvonne Y Gao, Jessica Freeze, Victor S Batista, Patrick H Vaccaro, Isaac L Chuang, Luigi Frunzio, et al. 2020. Efficient multiphoton sampling of molecular vibronic spectra on a superconducting bosonic processor. *Physical Review X* 10, 2 (2020), 021060.
- [48] Christopher S Wang, Nicholas E Frattini, Benjamin J Chapman, Shruti Puri, Steven M Girvin, Michel H Devoret, and Robert J Schoelkopf. 2023. Observation of wave-packet branching through an engineered conical intersection. *Physical Review X* 13, 1 (2023), 011008.
- [49] Junyu Zhou, Yuhao Liu, Yunong Shi, Ali Javadi-Abhari, and Gushu Li. 2024. Bosehedral: Compiler Optimization for Bosonic Quantum Computing. arXiv preprint arXiv:2402.02279 (2024).