

Provably Optimal Quantum Circuits with Mixed-Integer Programming

Harsha Nagarajan*

*Applied Mathematics and Plasma Physics (T-5),
Los Alamos National Laboratory, Los Alamos, New Mexico, USA*

Zsolt Szabó

*ARC Centre of Excellence for Engineered Quantum Systems,
Macquarie University, Sydney, NSW 2109, Australia and
School of Mathematical and Physical Sciences, Macquarie University, NSW 2109, Australia*

We present a depth-aware optimization framework for quantum circuit compilation that unifies provable optimality with scalable heuristics. For exact synthesis of a target unitary, we formulate a mixed-integer linear program (MILP) that *linearly* handles global-phase equivalence and uses explicit parallel scheduling variables to certify depth-optimal solutions for small-to-medium circuits. Domain-specific valid inequalities—including identity ordering, commuting-gate pruning, short-sequence redundancy cuts, and Hermitian-conjugate linkages—significantly tighten the relaxation and accelerate branch-and-bound, yielding speedups up to $43\times$ on standard benchmarks. The framework supports hardware-aware objectives, enabling fault-tolerant priorities (e.g., T -count) and NISQ-era penalties (e.g., entangling gates). For approximate synthesis, we propose three fidelity-driven objectives: (i) exact—but non-convex—phase-invariant fidelity maximization; (ii) a linear surrogate that maximizes the real trace overlap, yielding a tight lower bound to fidelity; and (iii) a convex quadratic function that minimizes the circuit’s Frobenius error.

To scale beyond exact MILP, we propose a novel rolling-horizon optimization (RHO) that rolls primarily in time, caps the active-qubit set, and enforces per-qubit closure while globally optimizing windowed segments. This preserves local context, reduces the effective Hilbert-space dimension, and enables iterative improvements without ancillas. On a 142-gate seed circuit, RHO yields 116 gates—an 18.3% reduction from the seed—while avoiding the trade-off between myopic passes and prohibitive solve times. Empirically, our exact compilation framework achieves certified depth-optimal decompositions on standard targets, high-fidelity Fibonacci-anyon weaves, and a 36% gate-count reduction on multi-body parity circuits. All methods are implemented in the open-source package `QuantumCircuitOpt`, providing a single optimization framework that bridges exact certification and hardware-aware, scalable synthesis.

I. INTRODUCTION

Quantum computation holds the potential to revolutionize information processing and the solution of complex problems. A major obstacle to its practical realization, however, is the compilation of quantum algorithms into executable low-level instructions for physical hardware. This task—known as *quantum compilation*—involves translating high-level descriptions of quantum algorithms into ordered sequences of native gates compatible with a given device, a process complicated by the constraints of current quantum hardware (restricted qubit connectivity, gate infidelities, decoherence) and the fragility of quantum states that are used to store and process information. Within this context, the research community has focused heavily on T gates, motivated by fault-tolerant quantum computing practices [1, 2]. In surface-code architectures, non-Clifford operations such as the T gate are typically implemented via magic-state distillation, which dominates both spatial and temporal resource costs. Consequently, minimizing T -count—and its parallelized counterpart, T -depth—remains a central objective in quantum circuit

optimization. In contrast, on NISQ-era devices without error correction, two-qubit gates (e.g., CNOT) are typically the dominant contributors to error and latency; minimizing their count and depth is therefore critical for performance. Given these hardware-specific cost models, efficient circuit compilation is essential to realizing a practical quantum advantage.

Heuristic approaches have demonstrated strong practical efficacy in reducing quantum circuit size and depth across hardware platforms. QFAST combines combinatorial search with numerical optimization to find low-depth implementations for moderate-scale circuits at high fidelity [3]. Hardware-accelerated data-flow engines on Field Programmable Gate Arrays (FPGAs) extend variational synthesis to 3–9 qubits while maintaining near-unit fidelities via parallel optimization [4]. Graph-theoretic methods based on the ZX-calculus systematically simplify circuits by merging phase gadgets and optimizing non-Clifford structure [5]. More recently, AlphaTensor-Quantum, a deep reinforcement learning framework, has reformulated circuit identity discovery as tensor optimization problems, demonstrating empirical improvements in T -gate count reduction through learned policies [6]. Despite their scalability—especially ZX rewriting and AlphaTensor-Quantum—and non-trivial T -gate savings, these methods provide neither optimality guar-

* Corresponding author: harsha@lanl.gov

antees nor a posteriori certificates on solution quality.

Exact synthesis algorithms provide optimality certificates only for constrained settings (small instances or restricted gate sets). The meet-in-the-middle search yields depth-optimal Clifford+ T decompositions for small multi-qubit circuits [7], while number-theoretic synthesis achieves near-optimal single-qubit Z -rotation approximations in Clifford+ T via Diophantine equation solutions [8]. Polynomial-time optimizers also exist for T -depth minimization in specific circuit structures [9]. However, the general problem is provably intractable: T -count and T -depth minimization for exact unitary synthesis are both NP-hard problems [9], with the hardness stemming from the combinatorial explosion in valid gate sequences subject to quantum mechanical constraints. This computational barrier explains the field’s reliance on heuristic approaches for circuits beyond a trivial scale.

Decades of mathematical advances in mixed-integer programming (MIP) have established it as a natural framework for high-dimensional, combinatorial design problems like quantum circuit compilation. Its applicability extends beyond circuit synthesis—qubit placement and routing, for instance, can be formulated as integer linear programs that jointly optimize initial layout and SWAP insertion [10]. MIP modeling provides four key advantages for quantum compilation: (i) Unified modeling of discrete choices and continuous algebraic constraints, (ii) Mature solvers with advanced presolve, cutting planes (or cuts), and parallel branch-and-bound, (iii) Flexible objectives beyond T -count (e.g., depth minimization, entangling-gate penalties, hardware-aware weights, fidelity), and (iv) Provable optimality through solution certificates or bounded optimality gaps for tractable instances.

Nagarajan *et al.* introduced a MIP formalism for circuit optimization in the open-source package **QuantumCircuitOpt** (QCOpt), providing a rigorous approach to circuit design with optimality guarantees [11]. Subsequently, [12] explored continuous/nonlinear formulations and relaxations that accelerate search while preserving solution quality. Together, these works establish a baseline for exact certification and an extensible modeling stack that we build upon here.

We contribute to QCOpt in five directions. (1) We formulate depth as a principal objective via explicit scheduling/precedence variables and per-depth qubit-disjointness, certifying depth-optimal solutions at modest scales. (2) We handle global-phase invariance *linearly* in a real embedding, eliminating non-convex phase constraints. (3) We propose a catalog of novel valid inequalities—identity ordering, commuting/equivalent-pattern pruning, short-sequence redundancy cuts, and Hermitian-conjugate linkages—that significantly tighten the MILP and accelerate branch-and-bound-based solvers. (4) For approximate synthesis, we formulate fidelity-driven objectives: an exact (non-convex) phase-invariant fidelity in real encoding, a linear real-part surrogate, and a piecewise-linear outer approximation

of Frobenius error; we also show that phase-optimized Frobenius minimization is equivalent to fidelity’s real-part maximization, and (5) For scale, we introduce a rolling-horizon algorithm that rolls primarily in time, caps the active-qubit set, and enforces per-qubit closure, preserving the tightened constraints and enabling iterative improvements on larger circuits.

Our framework is inherently adaptable to diverse hardware cost models through objective and gate-weight selection, accommodating both NISQ-era regimes—where two-qubit gate errors dominate—and fault-tolerant architectures, where non-Clifford resources (notably T gates) are the principal cost metric. The proposed formulation targets fundamental *unitary synthesis* rather than post-hoc circuit optimization: given a target unitary and an elementary gate set, the mixed-integer program selects an optimal gate ordered sequence that implements the target exactly (modulo global phase) or within a specified approximation tolerance. Hardware topology is incorporated natively by restricting multi-qubit gate placement to edges of the device coupling graph, thereby enforcing connectivity by construction rather than via post-processing.

These capabilities—together with our depth-optimization objective and catalog of domain-specific valid inequalities—yield provably optimal solutions for small-to-moderate circuits while preserving strong empirical performance at larger scales. This ensures immediate practical value across both near-term platforms and fault-tolerant architectures, with all details in the sections that follow.

II. MATHEMATICAL FORMULATION

Mathematical Preliminaries: For any integer $n \geq 1$, we denote $[n] := \{1, 2, \dots, n\}$ as the set of the first n positive integers, and thus $[n] \setminus \{1\} = \{2, \dots, n\}$. Let $\mathbb{I}_n$ be an $n \times n$ identity matrix. Throughout, $A^\dagger = \overline{A}^T$ denotes the Hermitian-conjugate; for unitary $U \in \mathbb{C}^{n \times n}$, $U^\dagger U = U U^\dagger = \mathbb{I}_n$.

Consider a quantum register of Q qubits with associated Hilbert space $\mathcal{H} = (\mathbb{C}^2)^{\otimes Q} \cong \mathbb{C}^{2^Q}$. For any qubit subset $S \subseteq [Q]$, define the unitary group on S and its special unitary subgroup as

$$\mathcal{U}(2^{|S|}) = \left\{ U \in \mathbb{C}^{2^{|S|} \times 2^{|S|}} \mid U^\dagger U = \mathbb{I}_{2^{|S|}} \right\}, \quad (1)$$

$$\mathcal{SU}(2^{|S|}) = \left\{ U \in \mathcal{U}(2^{|S|}) \mid \det U = 1 \right\}. \quad (2)$$

Let $\mathbb{G} \subset \bigcup_{S \subseteq [Q]} \mathcal{U}(2^{|S|})$ be a finite elementary gate set with fixed parameters. For each gate $g \in \mathbb{G}$ acting on subset $S_g \subseteq [Q]$, its extension to the full Hilbert space $\mathcal{H}$ is denoted $g^{(S_g)} \in \mathcal{U}(2^Q)$ as defined in Definition 2. For each qubit $q \in [Q]$, denote $\mathbb{G}_q = \{g \in \mathbb{G} \mid g \text{ acts non-trivially on } q\}$ as the subset of gates affecting qubit q (see Definition 1).

Quantum circuit compilation seeks an ordered sequence of gates $(U_p)_{p=1}^P$ where each U_p corresponds to the full Hilbert space representation of some gate from $\mathbb{G}$, such that:

$$\prod_{p=1}^P U_p = e^{i\phi} \tilde{T} \quad \phi \in [0, 2\pi), \quad (3)$$

where $\tilde{T} \in \mathcal{U}(2^Q)$ is the given target unitary, and P is the maximum allowable gate count. Specifically, for each position $p \in [P]$, there exists $g_p \in \mathbb{G}$ such that $U_p = g_p^{(S_{g_p})}$. The circuit is structured across P positions $p \in [P]$, with depth D of the circuit as defined in Definition 3.

The optimization landscape for quantum compilation encompasses multiple physically significant metrics, each addressing critical constraints of near-term and fault-tolerant quantum hardware:

- *Gate count:* Minimize P , the total number of elementary operations, to shorten the gate sequence and reduce computational time and error rates.
- *Entangling gate count:* Minimize entangling operations (e.g., CNOT, CZ), which typically have error rates 2–10 $\times$ higher than single-qubit gates across platforms, thereby mitigating crosstalk and other interaction-induced errors.
- *Non-Clifford resources:* In fault-tolerant architectures, minimize T gates (and other non-Clifford gates), each requiring costly magic-state distillation; this is particularly important in error-corrected computation due to their higher resource demands.
- *Circuit depth:* Minimize the circuit depth D (overall execution time) by maximal parallelism, thereby mitigating decoherence and dephasing in the algorithm.
- *Circuit fidelity:* Under fixed resource and hardware constraints, maximize circuit fidelity with the target unitary $\tilde{T}$ to ensure optimal performance.

To address any of these optimization metrics, the combinatorial complexity of quantum gate selection while respecting hardware constraints and the continuous nature of unitary evolution within a unified optimization framework, we cast circuit synthesis as a Mixed-Integer Linear Program (MILP). The complete formalism is presented in the following subsection.

A. Mixed-Integer Programming Formalism

A Mixed-Integer Linear Program (MILP) optimizes a linear objective subject to affine constraints over mixed

continuous-binary variables. The canonical form is:

$$\begin{aligned} \min_{\mathbf{x}, \mathbf{z}} \quad & \mathbf{c}^\top \mathbf{z} \\ \text{s.t.} \quad & A\mathbf{x} + B\mathbf{z} = \mathbf{b} \\ & \mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U \\ & \mathbf{z} \in \{0, 1\}^{n_z} \end{aligned} \quad (4)$$

where $A \in \mathbb{R}^{m \times n_x}$, $B \in \mathbb{R}^{m \times n_z}$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^{n_z}$, and $\mathbf{x}^L, \mathbf{x}^U \in (\mathbb{R} \cup \{-\infty, +\infty\})^{n_x}$ constitute the problem data. Continuous variables $\mathbf{x} \in \mathbb{R}^{n_x}$ represent real-encoded unitary entries, while binary variables $\mathbf{z} \in \{0, 1\}^{n_z}$ encode gate selection and depth assignments. The constraint matrix $[A \ B]$ enforces quantum-mechanical validity through linear relationships derived in detail in Section IID. When constraints or the objective function contain quadratic (bilinear) terms, the formulation becomes a Mixed-Integer Quadratic Program (MIQP).

Although solving MILPs to optimality is NP-hard in the worst case, modern commercial solvers (Gurobi [13], CPLEX [14]) routinely handle large, structured instances via branch-and-cut: they explore a branch-and-bound tree while repeatedly solving tightened linear-programming (continuous) relaxations and adding cutting planes. Decades of advances—presolve, cut separation, primal heuristics, and sophisticated branching—have yielded performance improvements outpacing Moore’s-law hardware gains [15]. Our contribution is a tight MILP formulation for quantum circuit synthesis that exactly captures the compilation problem’s combinatorial and continuous structure. While this exact representation theoretically guarantees optimal circuits, its $\Theta(4^Q)$ scaling with qubit count Q limits practical applicability. We therefore couple the formulation with quantum-specific enhancements that preserve optimality while enabling synthesis of larger circuits, demonstrating that rigorous MILP modeling combined with domain-aware acceleration yields a scalable exact synthesis framework for non-trivial quantum circuits.

B. Real Encoding of Complex Matrices

To represent complex matrices using real arithmetic while preserving the algebraic structure of matrix operations, we employ a real encoding where each complex entry is mapped to a 2×2 real block. Define the complex-to-real map

$$\mathcal{R} : \mathbb{C} \rightarrow \mathbb{R}^{2 \times 2}, \quad \mathcal{R}(a + ib) = \begin{bmatrix} a & -b \\ b & a \end{bmatrix}. \quad (5)$$

For $A \in \mathbb{C}^{n \times n}$ with entries A_{ij} , the elementwise block encoding $\mathcal{R}(A) \in \mathbb{R}^{2n \times 2n}$ is the $n \times n$ array of 2×2 blocks defined by

$$(\mathcal{R}(A))_{[2i-1:2i, 2j-1:2j]} = \mathcal{R}(A_{ij}) \quad \forall i, j \in [n]. \quad (6)$$

Equivalently, with E_{ij} the matrix units, $\mathcal{R}(A) = \sum_{i,j} E_{ij} \otimes \mathcal{R}(A_{ij})$. Up to a fixed permutation Π , this coincides with the canonical form: $\mathcal{R}(A) = \Pi \begin{bmatrix} \text{Re } A & -\text{Im } A \\ \text{Im } A & \text{Re } A \end{bmatrix} \Pi^\top$.

Basic properties. We write $A^\dagger = \overline{A}^\top$ for the conjugate transpose. The map $\mathcal{R}$ is an injective $*$ -algebra homomorphism:

$$\mathcal{R}(A+B) = \mathcal{R}(A) + \mathcal{R}(B), \quad \mathcal{R}(AB) = \mathcal{R}(A)\mathcal{R}(B), \quad (7)$$

$$\mathcal{R}(A^\dagger) = \mathcal{R}(A)^\top, \quad \mathcal{R}(\mathbb{1}_n) = \mathbb{1}_{2n}.$$

Consequently, if A is unitary, then $\mathcal{R}(A)^\top \mathcal{R}(A) = \mathcal{R}(A^\dagger A) = \mathcal{R}(\mathbb{1}_n) = \mathbb{1}_{2n}$, so $\mathcal{R}(A)$ is orthogonal. Moreover, the following proposition holds true:

Proposition 1. $\det \mathcal{R}(A) = |\det A|^2$.

Proof. By Schur decomposition $A = QTQ^\dagger$ with Q unitary and T upper triangular. Using multiplicativity, $\det \mathcal{R}(A) = \det \mathcal{R}(Q) \det \mathcal{R}(T) \det \mathcal{R}(Q^\dagger)$. Since Q is unitary, $\det \mathcal{R}(Q) = \det \mathcal{R}(Q^\dagger) = 1$. For triangular T with diagonal $(\tau_1, \dots, \tau_n)$, $\det \mathcal{R}(T) = \prod_{k=1}^n \det \mathcal{R}(\tau_k) = \prod_{k=1}^n |\tau_k|^2 = |\det T|^2 = |\det A|^2$. $\square$

Hence we have $A \in SU(n) \Rightarrow \det \mathcal{R}(A) = 1$.

For a *global complex phase* $\lambda = e^{i\phi}$ ($|\lambda| = 1$), with $\lambda = r + is$ for $r, s \in \mathbb{C}$, then

$$\mathcal{R}(\lambda A) = (\mathbb{1}_n \otimes \mathcal{R}(\lambda)) \mathcal{R}(A) = \mathcal{R}(A) (\mathbb{1}_n \otimes \mathcal{R}(\lambda)). \quad (8)$$

Thus a global-phase λ acts blockwise as the planar rotation $\mathcal{R}(e^{i\phi}) = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix}$, so any phase-invariant constraints in the complex model translate directly under this encoding.

Proposition 2. Define $J_n = \mathbb{1}_n \otimes \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} = \mathcal{R}(i \mathbb{1}_n)$. Then

$$\text{Re}(\text{Tr } A) = \frac{1}{2} \text{Tr}(\mathcal{R}(A)), \quad \text{Im}(\text{Tr } A) = -\frac{1}{2} \text{Tr}(J_n \mathcal{R}(A)).$$

Proof. Write $A_{ij} = a_{ij} + ib_{ij}$. By (5), $\mathcal{R}(A)$ has (i, i) -block $\begin{pmatrix} a_{ii} & -b_{ii} \\ b_{ii} & a_{ii} \end{pmatrix}$, whose trace is $2a_{ii}$. Summing over i gives $\text{Tr}(\mathcal{R}(A)) = 2 \sum_i a_{ii} = 2 \text{Re}(\text{Tr } A)$.

Using this result, for the imaginary part, we have $J_n \mathcal{R}(A) = \mathcal{R}(i \mathbb{1}_n) \mathcal{R}(A) = \mathcal{R}(iA)$, hence $\text{Tr}(J_n \mathcal{R}(A)) = \text{Tr}(\mathcal{R}(iA)) = 2 \text{Re}(\text{Tr}(iA)) = 2 \text{Re}(i \text{Tr}(A)) = -2 \text{Im}(\text{Tr } A)$. $\square$

Inverse map: For any $B \in \mathbb{R}^{2n \times 2n}$ with block structure $B_{2i, 2j} = B_{2i-1, 2j-1}$ and $B_{2i-1, 2j} = -B_{2i, 2j-1}$, $\forall i, j \in [n]$, define

$$(\mathcal{R}^{-1}(B))_{ij} = B_{2i-1, 2j-1} + i B_{2i, 2j-1}.$$

Then $\mathcal{R} : \mathbb{C}^{n \times n} \rightarrow \mathbb{R}^{2n \times 2n}$ is a bijection onto the set of real matrices with this structure, with $\mathcal{R}^{-1}(\mathcal{R}(A)) = A$ for all A and $\mathcal{R}(\mathcal{R}^{-1}(B)) = B$ for all such B .

C. Linearization of Bilinear Terms

The optimal circuit compilation problem's constraints in Eq. (3) contain multilinear terms and are therefore not directly MILP-representable. We obtain a MILP by replacing each binary-continuous product with its McCormick convex-hull linearization. For $z \in \{0, 1\}$ and $x \in [-1, 1]$, the set

$$S = \{(x, z, \beta) \mid \beta = zx, z \in \{0, 1\}, x \in [-1, 1]\} \quad (9)$$

has convex hull

$$\text{conv}(S) = \left\{ (x, z, \beta) \mid \begin{array}{l} \beta \leq z, \quad \beta \geq -z, \\ \beta \leq x + 1 - z, \quad \beta \geq x - 1 + z, \\ x \in [-1, 1], z \in [0, 1] \end{array} \right\}.$$

Crucially, when z is binary and $x \in [-1, 1]$, $\text{conv}(S)$ exactly enforces $\beta = zx$ due to the boundedness of x . The exactness of this linearization also extends to all bilinear binary-binary products ($z_1, z_2 \in \{0, 1\}$). Multilinear terms are linearized recursively by introducing auxiliary variables for each binary-continuous factor, preserving exactness [16]. The complete MILP reformulation (with auxiliaries) is in Section IID.

D. Variables and Constraints

Given a register of Q qubits and a finite set of elementary gates $\mathbb{G}$ with fixed entries, we seek to construct a quantum circuit composed of at most P gates, placed at positions $p \in [P]$, with an overall depth not exceeding a prescribed maximum depth D (the latter is defined in Definition 3 and particularly relevant in depth-optimization tasks).

Definition 1 (Trivial vs. non-trivial action on qubits). Let $\mathcal{H} = \mathbb{C}^{2^Q}$ and let $U \in \mathcal{U}(2^Q)$. For a qubit subset $S \subseteq [Q]$, we say that U acts trivially on S iff there exists a unitary $W \in \mathcal{U}(2^{Q-|S|})$ such that

$$U = P_S^\dagger (\mathbb{1}_{2^{|S|}} \otimes W) P_S, \quad (10)$$

where $P_S \in \mathcal{U}(2^Q)$ permutes tensor factors so that the qubits in S occupy the first $|S|$ positions (the property is independent of the particular choice of P_S). If no such W exists, then U acts non-trivially on S .

In particular, for a single qubit $q \in \{1, \dots, Q\}$, U acts trivially on q iff

$$U = P_{\{q\}}^\dagger (\mathbb{1}_2 \otimes W) P_{\{q\}} \quad \text{for some } W \in \mathcal{U}(2^{Q-1}). \quad (11)$$

Definition 2 (Gate extension to $\mathcal{H}$). For an elementary gate $U \in \mathcal{U}(2^{|S|})$ acting non-trivially on qubit subset $S \subseteq \{1, \dots, Q\}$, its extension to full Hilbert space $\mathcal{H}$ is the unitary operator $U^{(S)} \in \mathcal{U}(2^Q)$ defined by

$$U^{(S)} = P_S^\dagger (U \otimes \mathbb{1}_{2^{Q-|S|}}) P_S, \quad (12)$$

where $P_S \in \mathcal{U}(2^Q)$ is the permutation that moves the qubits in S (in increasing index order) to the first $|S|$ tensor slots and preserves the relative order of the remaining qubits. In particular, when $|S| = 1$ (i.e., $S = \{q\}$), $U^{\{q\}}$ reduces to $U \otimes \mathbb{1}_{2^{Q-1}}$ in the basis where qubit q is positioned first.

Definition 3 (Circuit depth). *Given a quantum circuit $\mathcal{C}$ with gate sequence $g_1, \dots, g_P$ acting on Q qubits, depth $D(\mathcal{C})$ is the smallest integer D for which there exists an assignment of gates to depths $\delta(p) \in [D]$ such that: (i) $\delta(p) \leq \delta(p+1) \forall p < P$, and (ii) For every qubit $q \in [Q]$ and depth $d \in [D]$, at most one gate acting non-trivially on q is assigned to depth d .*

Definition 3 captures the minimal number of sequential time steps required to execute a circuit $\mathcal{C}$ on hardware, where gates acting on disjoint qubits may execute in parallel within the same depth. To formalize this definition in the MIP context, we introduce several key variables and constraints:

Constraints: Gate count minimization

To facilitate an MILP formulation, we introduce (i) binary selection variables $z_{g,p} \in \{0,1\}$ indicating that gate $g \in \mathbb{G}$ is chosen at position $p \in [P]$, and (ii) real matrix variables $G_p \in [-1,1]^{2^{Q+1} \times 2^{Q+1}}$ denoting the fixed real block representation $G_p \equiv \mathcal{R}(U_p)$ of the unitary at position p (see Section II B). Using the canonical extension of gates $g \in \mathbb{G}$ to the full Hilbert space $\mathcal{H}$ (Definition 2), the gate-assignment constraints are

$$\sum_{g \in \mathbb{G}} z_{g,p} = 1, \quad \forall p \in [P], \quad (13a)$$

$$G_p = \sum_{g \in \mathbb{G}} z_{g,p} \cdot \mathcal{R}(g), \quad \forall p \in [P]. \quad (13b)$$

Constraint (13a) ensures exactly one gate occupies each position, and constraint (13b) links the binary choice to the (embedded) gate so that G_p equals the selected element of $\mathbb{G}$ acting on $\mathcal{H}$, encoded as a real matrix.

Additionally, let variables $\hat{G}_p$ represent the cumulative unitary up to position p , also with real values in $[-1,1]^{2^{Q+1} \times 2^{Q+1}}$. To maintain the correct circuit-product structure, we impose:

$$\begin{aligned} \hat{G}_1 &= G_1, \\ \hat{G}_p &= \hat{G}_{p-1} \cdot G_p, \quad \forall p \in [P] \setminus \{1\}. \end{aligned} \quad (14)$$

The recursive set of constraints in (14) is inherently bilinear in continuous-binary products, and thus, we use the successive McCormick linearization technique described in Section II C to handle them efficiently.

Constraints: Circuit depth minimization In addition to the variables and constraints introduced above, for the task of circuit-depth minimization (Definition 3)

we introduce assignment binaries $b_{p,d} \in \{0,1\}$ indicating that the gate at position p is scheduled at depth d (i.e., $b_{p,d} = 1$ iff the gate at position p is assigned to depth d). The depth-related constraints are:

$$\sum_{d=1}^D b_{p,d} = 1, \quad \forall p \in [P], \quad (15a)$$

$$0 \leq \sum_{d=1}^D d(b_{p,d} - b_{p-1,d}) \leq 1, \quad \forall p \in [P] \setminus \{1\}, \quad (15b)$$

$$\sum_{g \in \mathbb{G}_q} \sum_{p=1}^P z_{g,p} \cdot b_{p,d} \leq 1, \quad \forall q \in [Q], d \in [D]. \quad (15c)$$

Constraint (15a) assigns each position to exactly one depth; (15b) enforces a stepwise, monotonic depth ordering so that the depth at position $p+1$ is either equal to or exactly one greater than that at p ; and (15c) ensures that, at any depth, at most one gate acts on a given qubit q (per-depth qubit disjointness). The bilinear products $z_{g,p} \cdot b_{p,d}$ in (15c) are linearized exactly via the McCormick construction described in Section II C.

E. Target Constraint and Global-Phases

The final condition that must be imposed is that the resulting circuit matches the target unitary $\tilde{T}$, i.e., $\hat{G}_P = \mathcal{R}(e^{i\phi} \tilde{T})$ where $\phi \in [0, 2\pi)$ is arbitrary. At first glance, this may appear to be a non-convex constraint, as it involves matching two unitaries up to a global-phase (GP). However, the following characterization of the constraint on the last cumulative unitary $\hat{G}_P$ is sufficient to ensure that the circuit reproduces the target up to an arbitrary global phase:

$$\text{GP Target: } \hat{G}_P = \mathcal{R}((r + is) \cdot \tilde{T}), \quad (16)$$

where $r, s \in [-1,1]$ are newly introduced real variables that represent $e^{i\phi}$ in rectangular form. This constraint avoids the ϕ -dependency in Eq. (16) while preserving solution equivalence up to global phase.

Proposition 3. *Let $\tilde{T} \in \mathcal{U}(2^Q)$ and let $\hat{G}_P \in \mathcal{U}(2^{Q+1})$ denote the cumulative product of the selected elementary unitary gates. If, in addition, the linear constraint in (16) holds, then $|r + is| = 1$ (equivalently $r^2 + s^2 = 1$).*

Proof. Using unitarity of $\hat{G}_P$ and $\tilde{T}$, $\mathbb{1}_{2^{Q+1}} = \hat{G}_P^\dagger \hat{G}_P = (\mathcal{R}((r - is)\tilde{T}^\dagger))(\mathcal{R}((r + is)\tilde{T})) = |r + is|^2 \mathcal{R}(\tilde{T}^\dagger \tilde{T}) = |r + is|^2 \mathbb{1}_{2^{Q+1}}$. Thus $\mathbb{1}_{2^{Q+1}} = |r + is|^2 \mathbb{1}_{2^{Q+1}}$ implies $|r + is|^2 = 1$, i.e., $r^2 + s^2 = 1$. $\square$

Consequently, when (16) holds with $\tilde{T}$ and $\hat{G}_P$ unitary, the non-convex identity $r^2 + s^2 = 1$ follows automatically and need not be imposed explicitly. Thus, the matching of $\hat{G}_P$ to $\tilde{T}$ up to a global phase is enforced entirely by the linear GP conditions.

In principle, the correct condition is equality up to a global phase, as presented above. In practice, however, if both $\tilde{T}$ and all generated circuits lie in $\mathcal{SU}(2^Q)$, one may impose the stricter condition with $r = 1$, $s = 0$:

$$\text{Exact Target: } \hat{G}_P = \mathcal{R}(\tilde{T}), \quad (17)$$

which enforces exact equality in $\mathcal{SU}(2^Q)$. This ignores the residual action of the center $\mathcal{Z}_{2^Q} = \{e^{2\pi i k/2^Q} \mathbb{1}_{2^Q} : k = 0, \dots, 2^Q - 1\}$, but since global phases are physically irrelevant, this approximation has no impact on the compiled circuit's action on quantum states. We adopt this simplification in our numerical implementation for efficiency, with results presented in Section V A.

F. Compilation Objectives: Gate Count

Using the variables introduced above, we define objective functions for common compilation goals.

Objective: Hardware-aware gate count

Minimize a weighted count of non-identity gates:

$$\text{minimize } \sum_{p=1}^P \sum_{g \in \mathbb{G} \setminus \{\mathbb{1}\}} w_g z_{g,p}, \quad (18)$$

where $w_g \geq 0$ is a fixed per-gate weight parameter derived from hardware calibration or design priorities. Setting all $w_g \equiv 1$ recovers plain gate count; assigning larger w_g to non-Clifford gates, especially T , targets fault-tolerant resource costs (T -count), while weighting multi-qubit/entangling gates more heavily reflects NISQ-era hardware coupler error rates. Binary values on w_g recover subset counts (e.g., only non-Clifford or only multi-qubit), and excluding $\mathbb{1}$ from the objective ensures unused positions incur no cost.

Objective: Circuit depth

To minimize the circuit depth $D(\mathcal{C})$, we optimize the schedule by minimizing the depth at the final position P . This is achieved via the objective:

$$\text{minimize } \sum_{d=1}^D d \cdot b_{P,d}. \quad (19)$$

Because (15) enforces a nondecreasing schedule that can increase by at most one per position, the circuit depth equals the depth of the final position, i.e., $D(\mathcal{C})$ as per Definition 3. Minimizing this objective, therefore, yields a valid schedule with maximal parallelization of gates and hence minimal overall circuit depth.

G. Compilation Objectives: Circuit Fidelity

Exact circuit synthesis is often infeasible and, in many applications, unnecessary: it typically suffices to approximate a target unitary $\tilde{T}$ within a tolerance $\varepsilon > 0$ in

a phase-invariant metric (e.g., the fidelity). For any fixed, finite universal gate set, such as Clifford+ T , the Solovay–Kitaev theorem guarantees that generic unitaries can be approximated to arbitrary precision with sequence length polylogarithmic in $1/\varepsilon$ (see, e.g., [17]). In other settings, one may prefer alternative discrete families—such as Klein’s icosahedral group augmented by the “Super Golden T -gate” [18]—or seek to realize the canonical H , X , and T gates by braiding or weaving Fibonacci anyons on topological quantum computers [19, 20]. In these regimes, the design goal shifts: rather than minimizing gate count per se, one fixes a resource budget (total count, depth, T -count, etc.) and maximizes circuit fidelity.

We quantify performance using the phase-invariant fidelity [21],

$$F((U_p)_{p=1}^P) \equiv \frac{1}{2^{2Q}} \left| \text{Tr}(\tilde{T}^\dagger \prod_{p=1}^P U_p) \right|^2. \quad (20)$$

This is the squared, normalized Hilbert–Schmidt overlap between the implemented unitary $\prod_p U_p$ and the target $\tilde{T}$. It obeys $0 \leq F(\cdot) \leq 1$, with $F(\cdot) = 1$ if and only if $\prod_p U_p = e^{i\phi} \tilde{T}$ for some global-phase ϕ , and it is invariant under rephasing of either argument.

Exact fidelity objective

When MIP solvers support non-convex objectives, we optimize the phase-invariant fidelity (20) directly. Using real-encoding from Prop. 2, define:

$$\begin{aligned} \alpha &\equiv \frac{1}{2^{Q+1}} \text{Tr}(\mathcal{R}(\tilde{T})^\dagger \hat{G}_P) = \text{Re} \left(\frac{1}{2^Q} \text{Tr}(\tilde{T}^\dagger \prod_{p=1}^P U_p) \right), \quad (21) \\ \beta &\equiv -\frac{1}{2^{Q+1}} \text{Tr}(J_{2^Q} \mathcal{R}(\tilde{T})^\dagger \hat{G}_P) = \text{Im} \left(\frac{1}{2^Q} \text{Tr}(\tilde{T}^\dagger \prod_{p=1}^P U_p) \right), \quad (22) \end{aligned}$$

where $J_{2^Q} = \mathbb{1}_{2^Q} \otimes \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$. Since $F(\cdot) = \alpha^2 + \beta^2$, exact fidelity maximization requires

$$\max \alpha^2 + \beta^2, \quad (23)$$

while subject to all the base model’s linear constraints. Owing to the modulus-squared trace in our decision variables, objective (23) is non-convex, which precludes direct treatment by convex optimization methods. While theoretically exact, this formulation becomes computationally intractable beyond small instances due to combinatorial non-convexity. We therefore investigate two linear, MILP-amenable surrogates of (20). Their comparative empirical performance is reported in Section V.

Linearized fidelity (real-part surrogate)

We now present a MILP-amenable surrogate by maximizing only the real trace overlap of the exact objective in (23):

$$\max \alpha \quad (24)$$

Since $\beta^2 \geq 0$, $\alpha^2 \leq F(\cdot)$ provides a strict lower bound for $F(\cdot)$. Critically, this surrogate becomes asymptotically tight as $F(\cdot) \rightarrow 1$: when $F(\cdot) > 0.999$, residual phase errors (β) are typically negligible under $SU(2^Q)$ encoding, ensuring $\alpha \approx \sqrt{F(\cdot)}$.

Frobenius Error Objective

Next, we present an equivalent characterization of the linearized fidelity objective (24) while enabling scalable implementation by parametrizing the mismatch between the compiled circuit and the target unitary. Introduce a deviation matrix $E \in \mathbb{R}^{2^{Q+1} \times 2^{Q+1}}$ (elementwise bounds $-\epsilon \leq E_{ij} \leq \epsilon$) via

$$\hat{G}_P = \mathcal{R}(\tilde{T}) + E. \quad (25)$$

A natural phase-invariant fidelity objective surrogate (in the real embedding) is to minimize the Frobenius error

$$\min \|E\|_F^2 = \min \text{Tr}(EE^\top), \quad (26)$$

which is a *convex* quadratic objective with the existing base model's linear constraints. We now present Proposition 4, which characterizes the relationship between Frobenius-error minimization (26) and fidelity maximization (23) objectives.

Proposition 4. *Consider a circuit compilation problem with fixed maximum depth P , fixed gate set $\mathbb{G}$, and target unitary $\tilde{T}$. Let $\mathcal{S}$ denote the feasible set of circuits $(U_P)_{P=1}^P$ such that $\hat{G}_P = \mathcal{R}(\tilde{T}) + E$. Then:*

- (i) $\|E\|_F^2 = 2^{Q+2}(1-\alpha)$, where α is defined as in (21).
- (ii) Minimizing $\|E\|_F^2$ is equivalent to maximizing α , the real trace overlap of the fidelity objective $F(\cdot)$, but not necessarily to maximizing $F(\cdot)$ when $\max_{\mathcal{S}} F(\cdot) < 1$.

Proof. (i): Using the orthogonality of $\hat{G}_P$ and $\mathcal{R}(\tilde{T})$, and expanding the Frobenius norm using $E = \hat{G}_P - \mathcal{R}(\tilde{T})$:

$$\begin{aligned} \|E\|_F^2 &= \text{Tr}((\hat{G}_P - \mathcal{R}(\tilde{T}))(\hat{G}_P - \mathcal{R}(\tilde{T}))^\top) \\ &= 2 \cdot \text{Tr}(\mathbb{1}_{2^{Q+1}}) - \text{Tr}(\hat{G}_P \mathcal{R}(\tilde{T})^\top) - \text{Tr}(\mathcal{R}(\tilde{T}) \hat{G}_P^\top) \end{aligned}$$

By trace properties (see (Prop. 2).), both cross terms equal $\text{Tr}(\mathcal{R}(\tilde{T})^\top \hat{G}_P)$, and by equation (21), so:

$$\begin{aligned} \|E\|_F^2 &= 2 \cdot \text{Tr}(\mathbb{1}_{2^{Q+1}}) - 2 \cdot \text{Tr}(\mathcal{R}(\tilde{T})^\top \hat{G}_P) \\ &= 2 \cdot 2^{Q+1} - 2 \cdot (2^{Q+1}\alpha) \\ &= 2^{Q+2}(1-\alpha) \end{aligned}$$

This completes the proof of (i).

(ii): From (i), $\|E\|_F^2 = 2^{Q+2}(1-\alpha)$ is a strictly decreasing function of α . Therefore, minimizing $\|E\|_F^2$ is equivalent to maximizing α . However, since $F(\cdot) = \alpha^2 + \beta^2$, maximizing α does not necessarily maximize $F(\cdot)$ when β can vary independently. This independence arises because the orthogonality constraint $\hat{G}_P \hat{G}_P^\top = \mathbb{1}$ permits multiple values of β for fixed α (and hence fixed $\|E\|_F^2$), depending on the alignment of E with $J_{2^Q} \mathcal{R}(\tilde{T})$. $\square$

Remark 1. *From the identity (i) of Proposition (4) and fidelity definition $F(\cdot) = \alpha^2 + \beta^2$, we obtain a tight lower bound:*

$$F(\cdot) \geq \left(1 - \frac{\|E\|_F^2}{2^{Q+2}}\right)^2, \quad (27)$$

with equality iff $\beta = 0$. This bound is asymptotically tight as $F(\cdot) \rightarrow 1$: when $F(\cdot) > 0.999$, residual phase errors (β) can typically become negligible under $SU(2^Q)$ encoding.

Although (26) is convex quadratic, to retain a *linear* objective, and thus a MILP formulation that scales, we globally under-approximate each quadratic term E_{ij}^2 by a finite family of first-order (tangent) approximations to $x \mapsto x^2$ on an uniform grid $\{a_k\}_{k=1}^K \subset [-\epsilon, \epsilon]$ (see Table IV for speedups). Introduce auxiliaries $\hat{E}_{ij} \geq 0$ and write the objective with tangent constraints as

$$\begin{aligned} \min \quad & \sum_{(i,j) \in [2^{Q+1}] \times [2^{Q+1}]} \hat{E}_{ij} \\ \text{s.t.} \quad & \hat{E}_{ij} \geq 2a_k E_{ij} - a_k^2, \quad \forall (i,j), k \in [K]. \end{aligned} \quad (28)$$

Consequently, for any feasible solution of (28), the optimal objective value is a valid *lower bound* to the true quadratic objective (26). Tightness, also a surrogate for Fidelity, improves monotonically with K .

Alternative objectives based on other matrix norms (e.g., ℓ_1 and ℓ_∞) were evaluated against the Frobenius-based quadratic outer-approximation surrogate for measuring compilation error. Across benchmark circuits, they yielded negligible fidelity gains while increasing wall-clock time; accordingly, we report results using the quadratic outer-approximation metric, which offered the best quality-time trade-off.

Neither linear surrogate ((24) or (28)) guarantees optimality for fidelity maximization in (20), but both achieve near-optimal fidelities ($F(\cdot) > 0.999$) significantly faster than non-convex or brute-force methods. Phase-invariant alternatives (e.g., minimizing $\|\hat{G}_P - e^{i\phi} T\|_F$ over ϕ) yield equivalent linear formulations. We evaluate the scalability of these approaches versus the exact MIQP (23) in Section V.

III. SPEEDING UP THE COMPILATION

The MILP formulation in Section II (the “base” formulation), exactly models the quantum circuit synthesis problem, theoretically guaranteeing global optimality for any transpilation task with a finite gate set. However, the combinatorial structure of quantum circuits induces significant computational challenges: the formulation exhibits inherent symmetry (multiple equivalent gate sequences yielding identical unitaries), a weak continuous (linear-programming) relaxation due to the bilinear nature of quantum operations, and exponential growth in

variable count with respect to qubit count. These properties lead to slow convergence in branch-and-cut algorithms, as evidenced by empirical results showing impractical solve times for circuits exceeding 3 qubits or depth 10.

To strengthen the formulation, we derive three families of valid constraints that preserve the global optimal solution while improving the integrality gap and reducing symmetry in the branch-and-bound tree. While theoretical analysis confirms their validity, the practical efficacy of these enhancements depends critically on the specific circuit and must be evaluated empirically. Our experimental results in Section V demonstrate that these domain-specific valid constraints reduce solve times in state-of-the-art solvers by up to two orders of magnitude for non-trivial benchmarks.

A. Symmetry Constraints

The first set of valid constraints works by reducing the search space by eliminating valid symmetrical optimal solutions. One immediate example arises from the placement of identity gates. Given an optimal compilation with identity gates, an equally optimal solution can be obtained by shifting these identities to other positions. To eliminate such symmetry, we introduce the constraint:

$$z_{1,p-1} \leq z_{1,p}, \quad \forall p \in [P] \setminus \{1\}. \quad (29)$$

We refer to this as the *Identity Gate Symmetry* constraint.

Similarly, consider two gates g_i and g_j that commute with each other. If these gates appear consecutively, swapping their order yields another equally optimal solution. To break this symmetry, we impose:

$$z_{g_i,p-1} + z_{g_j,p} \leq 1, \quad \forall p \in [P] \setminus \{1\}. \quad (30)$$

thereby reducing the search space by eliminating similar solutions.

A generalization of this commutativity constraint involves evaluating pairs of gates and identifying equivalent products. The problem is then constrained so that only one of the equivalent pairs is allowed, while the other is made infeasible. For example, this well-known relationship of the Pauli gates X and Z together with the Hadamard: $X \cdot H = H \cdot Z$ when all act on the same qubit, they are equivalent. By constraining the possibility that Z can follow a Hadamard, i.e., $z_{H,p} + z_{Z,p+1} \leq 1$, we can reduce the symmetries in the search space of optimal solutions.

Similarly, symmetry reduction via *equivalent triplet* elimination is crucial when distinct three-gate sequences implement the same unitary. This approach is particularly critical for topological quantum compilation via Fibonacci anyon braiding [22], where the braid group B_n (governing exchanges of n anyons with $n-1$ generators σ_i) satisfies the Yang-Baxter relation [23, 24]:

$$\sigma_i \cdot \sigma_{i+1} \cdot \sigma_i = \sigma_{i+1} \cdot \sigma_i \cdot \sigma_{i+1}, \quad \forall i = 1, \dots, n-2,$$

which induces local two-way redundancies. To prune symmetric branches without excluding optimal solutions, we enforce a canonical choice by forbidding one representative. Concretely, for consecutive positions $(p, p+1, p+2)$ we impose the valid inequality

$$z_{\sigma_i,p} + z_{\sigma_{i+1},p+1} + z_{\sigma_i,p+2} \leq 2, \\ \forall i \in \{1, \dots, n-2\}, \quad \forall p \in \{1, \dots, P-2\}, \quad (31)$$

which excludes $(\sigma_i \cdot \sigma_{i+1} \cdot \sigma_i)$ while retaining feasibility through the equivalent $(\sigma_{i+1} \cdot \sigma_i \cdot \sigma_{i+1})$. For Fibonacci anyon models where such local symmetries proliferate with n , this symmetry-breaking technique significantly accelerates MIP's branch-and-bound convergence while preserving optimality.

Remark 2. *Eliminating one representative of two locally equivalent canonical patterns (e.g., commuting gates or triplet sequences) is valid for order-insensitive objectives (total gate count (18); phase-invariant fidelity (23), (26)), but generally invalid for depth minimization objective in (19), where gate order affects parallelism via (15b)–(15c).*

Counterexample. Define $T_1 := T \otimes \mathbb{1}_2$, $T_2 := \mathbb{1}_2 \otimes T$, $\text{CNOT}_{1,2} \in \mathcal{U}(2)$ with (control on qubit 1, target on qubit 2). Since T_1 is diagonal and acts only on the control, T_1 commutes with $\text{CNOT}_{1,2}$, hence

$$T_1 \cdot \text{CNOT}_{1,2} \cdot T_2 = \text{CNOT}_{1,2} \cdot (T \otimes T).$$

Without symmetry breaking, schedule $\text{CNOT}_{1,2}$ at depth 1 and $(T \otimes T)$ in parallel at depth 2 is “optimal”. If a symmetry-breaking rule enforces $T_1 \prec \text{CNOT}_{1,2}$, then T_1 is at depth 1, $\text{CNOT}_{1,2}$ at depth 2, and T_2 must be at depth 3. Thus the enforced local symmetry elimination strictly increases optimal depth; such symmetry constraints are invalid for depth minimization.

B. Redundancy-elimination Constraints

To eliminate redundant gate sequences while maintaining computational tractability, we impose valid constraints exclusively for sequences of length $2 \leq k \leq 5$. For a gate set $\mathbb{G}$ and any sequence $(g_1, \dots, g_k) \in \mathbb{G}^k$ satisfying the algebraic identity $g_1 \cdots g_k = g'$ for some $g' \in \mathbb{G}$, the following constraint is enforced for all starting positions $p \leq P - k + 1$:

$$\sum_{i=1}^k z_{g_i,p+i-1} \leq k-1, \quad \forall k \in [5] \setminus \{1\}. \quad (32)$$

This prevents the inclusion of the redundant sequence $(g_1, \dots, g_k)$ at consecutive positions p to $p+k-1$, as its effect is algebraically equivalent to the single gate g' .

We consider $k \leq 5$ (up to quintuplets) to cover the standard minimal library $\{\text{CNOT}, H, T\}$, identities canonical relations such as control-target reversal $(H \otimes H) \cdot \text{CNOT}_{1,2} \cdot (H \otimes H) = \text{CNOT}_{2,1}$, $T^2 = S$, $X^2 = \mathbb{I}$,

$X \cdot H \cdot Z = H$, $H_2 \cdot \text{CNOT}_{1,2} \cdot H_2 = \text{CZ}_{1,2}$, as well as cyclic relations of weaving operators in topological quantum computation with Fibonacci anyon models [19, 25].

To summarize, symmetry constraints eliminate equivalent optimal circuits by identifying equivalence classes and restricting to a canonical representative, while redundancy constraints remove provably dominated solutions that correspond to non-minimal gate sequences implementing identical unitaries. The combined application of both constraint classes induces a valid restriction of the feasible region of discrete solutions that preserves solution optimality while substantially reducing the branch-and-bound search space through elimination of both symmetric solutions and dominated sequences.

C. Hermitian-conjugate Constraints

The constraints introduced in this section—*Hermitian-conjugate constraints (HC)*—do not exclude any integer-feasible compilation; they encode algebraic identities that every valid compiled circuit already satisfies. As noted in Section II A, MILP solvers proceed via LP relaxations in which gate-selection variables may take fractional values; at such fractional points these identities generally fail. Enforcing them therefore leaves the convex hull of integer solutions unchanged while cutting off infeasible fractional solutions, yielding a tighter LP relaxation. A stronger relaxation improves the root-node dual bound, reduces the integrality gap, and typically shrinks the branch-and-bound tree. Consequently, these identities serve as valid constraints that accelerate convergence.

The following Hermitian-conjugate (HC- γ) identity, applied near the end of the circuit, was found significantly effective for the circuit compilation problem as modeled in this paper. Given the recursion relation in (14) and the exact target relation $\hat{G}_P = \mathcal{R}(\tilde{T})$, for any integer $1 \leq \gamma \leq P$,

$$\hat{G}_{P-\gamma} = \mathcal{R}(\tilde{T}) \left(\prod_{t=P-\gamma+1}^P G_t \right)^\dagger. \quad (\text{HC-}\gamma)$$

This follows directly from unitarity: right-multiplying by the Hermitian conjugate of the trailing product of gates yields the identity.

In this work we enforce HC-1 ($\gamma = 1$) and HC-2 ($\gamma = 2$). Because the gate-selection constraint (13) chooses exactly one $g \in \mathbb{G}$ at each position, we have $G_t^\dagger = \sum_{g \in \mathbb{G}} z_{g,t} \mathcal{R}(g)^\dagger$. Therefore HC-1 is linear in the decision variables $z_{g,t}$. By contrast, HC- γ with $\gamma \geq 2$ involves products, which are bilinear in $\{z_{g,t}\}$; depending on solver capabilities, these constraints can be imposed directly or via standard convexifications (see Section II C).

Global-phase generalization of HC-1 identity

The Hermitian-conjugate conditions must be generalized to reflect the fundamental fact that unitaries are defined only up to a global phase, as in (16). Although this generalization is nontrivial for arbitrary γ , we now present the Hermitian-conjugate identity HC-1, whose corresponding phase-invariant version is:

$$\hat{G}_{P-1} = \mathcal{R}(r + is) \cdot \sum_{g \in \mathbb{G}} z_{g,P} \cdot \mathcal{R}(\tilde{T} \cdot g^\dagger), \quad (33)$$

where let $\tilde{T} \cdot g^\dagger = A_g + iB_g$ for fixed matrices $A_g, B_g \in \mathbb{R}^{2^Q \times 2^Q}$ whose entries satisfy $|(\tilde{T} \cdot g^\dagger)_{ij}| \leq 1$ since both $\tilde{T}$ and $g^\dagger$ are unitary.

The product of (r, s) with the binary variables $\{z_{g,P}\}$ renders (33) *nonlinear*. We eliminate this nonlinearity via an exact disjunctive reformulation. Let $\mathbb{J}_{2^Q}$ denote the $2^Q \times 2^Q$ all-ones matrix. Then the phase-sensitive condition in (33) is exactly equivalent to the entry-wise linear inequalities stated in the following Proposition 5. Here $|\cdot|$ denotes the matrix whose (i, j) -th entry is the absolute value of the difference between the corresponding matrix entries, and $\mathcal{R}^{-1}(\cdot)$ is the inverse real-to-complex map defined in Section II B.

Proposition 5. *Let $\mathbb{G}$ be a finite set of unitary gates. Suppose $r, s \in [-1, 1]$ satisfy $r^2 + s^2 = 1$, and $|(\tilde{T} \cdot g^\dagger)_{ij}| \leq 1$ for all i, j . Then the nonlinear constraint in (33) is equivalent to the set of element-wise linear inequalities for all $g \in \mathbb{G}$:*

$$\begin{aligned} |\text{Re}(\mathcal{R}^{-1}(\hat{G}_{P-1})) - (rA_g - sB_g)| &\leq 2 \cdot (1 - z_{g,P}) \cdot \mathbb{J}_{2^Q}, \\ |\text{Im}(\mathcal{R}^{-1}(\hat{G}_{P-1})) - (rB_g + sA_g)| &\leq 2 \cdot (1 - z_{g,P}) \cdot \mathbb{J}_{2^Q}, \end{aligned}$$

Proof. Assume first that the nonlinear constraint (33) is valid. Then there exists a unique $g^* \in \mathbb{G}$ with $z_{g^*,P} = 1$ and $z_{g,P} = 0$ for $g \neq g^*$, so $\hat{G}_{P-1} = \mathcal{R}(r + is) \cdot \mathcal{R}(\tilde{T} \cdot (g^*)^\dagger)$. By the properties of the real encoding $\mathcal{R}$ (from Section II B), this implies $\mathcal{R}^{-1}(\hat{G}_{P-1}) = (r + is)(\tilde{T} \cdot (g^*)^\dagger)$, and thus $\text{Re}(\mathcal{R}^{-1}(\hat{G}_{P-1})) = rA_{g^*} - sB_{g^*}$ and $\text{Im}(\mathcal{R}^{-1}(\hat{G}_{P-1})) = rB_{g^*} + sA_{g^*}$. For $g = g^*$, the right-hand sides of the linear constraints vanish, and the aforementioned equalities hold exactly. For $g \neq g^*$, $z_{g,P} = 0$ gives a right-hand side of $2 \cdot \mathbb{J}_{2^Q}$. Since $|(\tilde{T} \cdot g^\dagger)_{ij}| \leq 1$ and $r^2 + s^2 = 1$ is implied due to Proposition 3, the modulus of each entry of $(r + is)(\tilde{T} \cdot g^\dagger)$ is bounded by 1. Thus, both $\mathcal{R}^{-1}(\hat{G}_{P-1})$ and $(r + is)(\tilde{T} \cdot g^\dagger)$ have real and imaginary parts with entries in $[-1, 1]$, making the element-wise differences bounded by 2. Conversely, if the linear constraints hold with a unique g^* having $z_{g^*,P} = 1$, then for g^* the constraints enforce exact equality, recovering the nonlinear constraint, while for other g the constraints are automatically satisfied due to the 2-bound. Thus, the reformulation is exact. $\square$

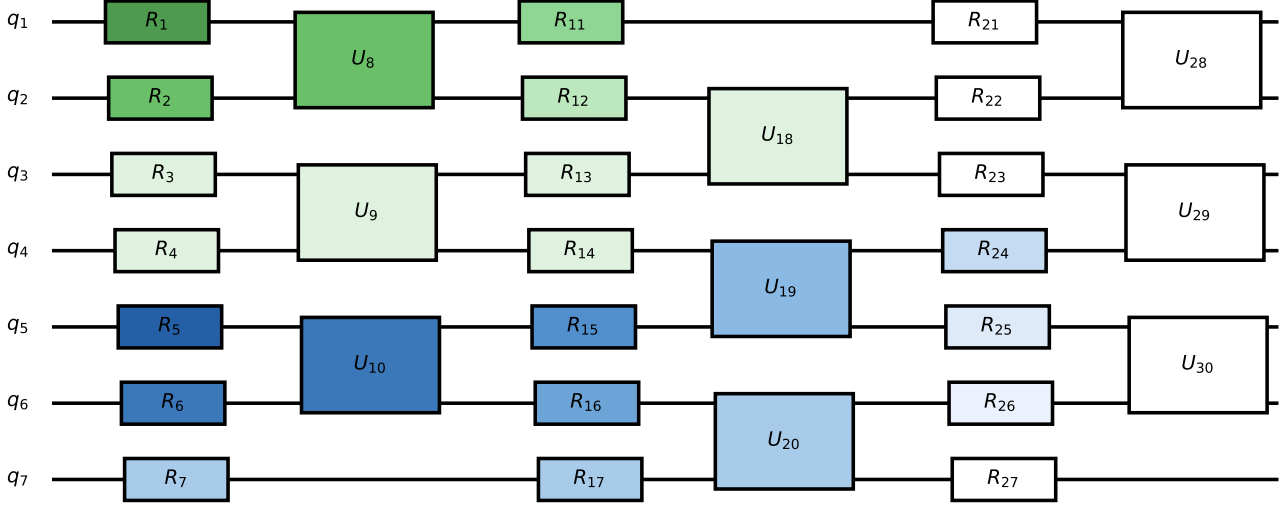


FIG. 1. Hardware-efficient brickwork ansatz on 7 qubits (cf. [26, 27]), illustrating the identification of optimization blocks in the rolling-horizon optimization (RHO). Single- and two-qubit gates are slightly offset horizontally to indicate that, while typically parallel, any fixed virtual time ordering is admissible; for exposition, we adopt a top-to-bottom order. Green shading denotes the gate insertion order in the *first* rolling-horizon window (dark→light); blue shading analogously denotes the *second* window.

IV. ROLLING-HORIZON OPTIMIZATION FRAMEWORK

The computational complexity of exact quantum circuit synthesis via MILP exhibits exponential dependence on qubit count Q , with formulation size scaling as $\Theta(4^Q)$. This inherent intractability renders global optimization impractical beyond modestly sized circuits, despite the inclusion of valid strengthening constraints above.

To address this complexity barrier while preserving solution quality, **QC0pt** employs a rolling-horizon optimization (RHO) wrapper for the specific task of gate-count minimization. The global problem is decomposed into a sequence of windows (horizons) of subproblems that are solved independently (with a potential for parallelization); their solutions are then concatenated to yield a hardware-feasible, near-optimal circuit. This improves tractability at the cost of forfeiting global optimality guarantees.

Traditional rolling-horizon methods advance strictly along the circuit’s time axis, optimizing gates in chronological order. Our RHO generalizes this by rolling the horizon over subsets of qubits (with short temporal windows), thereby partitioning both temporal and spatial dimensions and reducing complexity.

RHO in our setting presupposes an existing (typically suboptimal) target circuit to refine. That is, given an initial gate list that implements the full target $\tilde{T}$ (e.g., from a heuristic transpiler), we do not synthesize from scratch; instead, we iteratively improve contiguous subcircuits while preserving overall functionality. The two design levers are the *window-selection policy* (how we

choose the next block to optimize) and the *acceptance policy* (how much of the optimized block we lock in before rolling forward). In practice, the bottleneck step is selecting the first optimization block: this choice determines how the horizon will roll and how quickly the qubit set grows.

After optimizing the first block, we accept only an initial prefix of that optimized window (the acceptance window) and then retarget the remaining problem by effectively left-multiplying with the Hermitian-conjugate of what we accepted. If B_{acc} denotes the product of the accepted gates (in forward time order), the full target is updated as

$$\tilde{T} \leftarrow B_{\text{acc}}^\dagger \tilde{T}. \quad (34)$$

For example, if in the first iteration we accept a single $\text{CNOT}_{1,2}$, then the new full target becomes $\text{CNOT}_{1,2} \cdot \tilde{T}$. Setting the acceptance window equal to the optimization window eliminates overlap between chunks but tends to hurt quality; accepting only a minimal prefix improves quality at the expense of more iterations.

The first block we seek should satisfy several properties. (i) It must involve at most `max_qubits` distinct qubits to keep the effective Hilbert space small. (ii) Its length must not exceed `window_length`. (iii) It should *roll primarily in time*: we extend the window chronologically and only enlarge the qubit set when forced to by multi-qubit gates, thereby avoiding unnecessary growth in subsystem size. (iv) It should satisfy a *closure* condition on included qubits: once a gate on qubit q enters the window up to index i , all gates acting on q that occur before i must also be included. This prevents inconsis-

tent partial histories on any involved qubit and yields well-posed subproblems.

Operationally, our RHO policy: finds the earliest feasible block that obeys the cardinality bound on qubits, the length bound on gates, the temporal preference (extend in time first, add qubits only when necessary), and the per-qubit closure rule; optimizes that block; accepts a prefix; retargets the remainder; and repeats. This policy balances stability (small qubit footprint with preserved local context) with progress (steady forward motion in time) and underlies the routines with four key functions described in detail below:

a. gates_on_qubits_up_to This function takes a target gate sequence t , a set of qubits q , and an index i . It selects all gates from the beginning of the sequence up to position i that act on any qubit in q . The function iterates through the sequence, checking the qubits each gate acts upon, and appends matching gates to a list.

Algorithm 1 *

GATES_ON_QUBITS_UP_TO(t, q, i)

```

1: Input: gate list, qubit set, index
2: Output: gates acting on  $q$  up to  $t[i]$ 
3: selected_gates  $\leftarrow []$ 
4: for  $j = 1$  to  $i$  do
5:   gate  $\leftarrow t[j]$ 
6:   gate_qubits  $\leftarrow \text{EXTRACTQUBITS}(\text{gate})$ 
7:   if  $\text{INTERSECTION}(\text{gate\_qubits}, q) \neq \emptyset$  then
8:     APPEND(selected_gates, gate)
9: return selected_gates
```

b. recursive_gates_on_qubits_up_to Building on the previous function, this recursively expands the set of selected qubits. Initially calling *gates_on_qubits_up_to*, it updates the qubit set q based on qubits involved in selected gates and repeats until no new qubits are found in order to fulfill the closure condition.

Algorithm 2 *

RECURSIVE_GATES_ON_QUBITS_UP_TO(t, q, i)

```

1: Input: gate list, qubit set, index
2: Output: extended gate list acting on qubits up to  $t[i]$ 
3: selected_gates  $\leftarrow \text{GATES\_ON\_QUBITS\_UP\_TO}(t, q, i)$ 
4: new_q  $\leftarrow \text{EXTRACTQUBITS}(\text{selected\_gates})$ 
5: if  $\text{LENGTH}(\text{new\_q}) > \text{LENGTH}(q)$  then
6:   return  $\text{RECURSIVE\_GATES\_ON\_QUBITS\_UP\_TO}(t,$ 
7:      $\text{new\_q}, i)$ 
7: else
8:   return selected_gates
```

c. find_first_block Identifies the first feasible block of gates to optimize, constrained by a specified window length and maximum qubit count. It incrementally evaluates sequences until one exceeds the constraints, returning the largest feasible block found.

Algorithm 3 *

FIND_FIRST_BLOCK($t, \text{window_length}, \text{max_qubits}$)

```

1: Input: target gate sequence, window size, max number
  of qubits in a window
2: Output: best gate block satisfying constraints
3:  $q \leftarrow \text{EXTRACTQUBITS}(t[1])$ 
4:  $i \leftarrow 1$ 
5: best_sequence  $\leftarrow []$ 
6: while true do
7:   seq  $\leftarrow \text{RECURSIVE\_GATES\_ON\_QUBITS\_UP\_TO}(t, q, i)$ 
8:    $q \leftarrow \text{EXTRACTQUBITS}(\text{seq})$ 
9:    $\text{cond1} \leftarrow i > \text{LENGTH}(t)$ 
10:   $\text{cond2} \leftarrow \text{LENGTH}(\text{seq}) > \text{window\_length}$ 
11:   $\text{cond3} \leftarrow \text{LENGTH}(q) > \text{max\_qubits}$ 
12:  if  $\text{cond1 or cond2 or cond3}$  then
13:    return best_sequence
14:  else if  $\text{LENGTH}(\text{seq}) = \text{window\_length}$  then
15:    return seq
16:  else
17:    best_sequence  $\leftarrow \text{seq}$ 
18:     $i \leftarrow i + 1$ 
```

d. rolling_horizon This function drives optimization by sequentially selecting and optimizing blocks, updating the gate sequence until all gates are processed.

To illustrate how the first block is found under rolling horizon, we use the 7-qubit hardware-efficient ansatz in Fig. 1 (cf. [26, 27]). Although single-qubit rotations are often treated as parallel, in practice, any fixed virtual time ordering is admissible; in the figure, we intentionally offset them slightly to make this explicit. In this example, we adopt a simple acceptance policy—accept the entire optimized window—and set the window length to 12 and per-window qubit cap to 4.

The search for the first block begins at the earliest gate, R_1 on qubit 1 (dark green). The next gate on that qubit is U_8 ; inserting U_8 triggers the closure condition, which simultaneously pulls in R_2 , so both enter the window. The algorithm then adds R_{11} and R_{12} (now 5 gates). The next candidate is U_{18} , but closure requires adding $R_3, R_4, U_9, R_{13}, R_{14}$, bringing the total to 11. The subsequent candidate U_{19} —together with $R_5, R_6, U_{10}, R_{15}, R_{16}$ required by closure—would exceed both the qubit cap (> 4) and the window length (> 12), so these additions are rejected. The algorithm therefore optimizes the valid 11-gate window by forming the corresponding target and solving the exact MILP to minimize gate count. In Fig. 1, progressively lighter shades of green indicate the order in which gates enter this first window.

Because the acceptance policy fixes each optimized window, that portion of the target circuit is removed, and the search moves to the next “first block.” It resumes at R_5 , now the earliest remaining gate. The second block is assembled as follows: U_{10} and R_6 are added (closure), then R_{15} and R_{16} ; adding U_{19} enlarges the active-qubit set to qubits 4–6, and adding U_{20} pulls in R_7 and R_{17} , activating qubit 7. Subsequent rounds add R_{24}, R_{25} ,

Algorithm 4 *

```

ROLLING_HORIZON(target_sequence, window_length, max_qubits, elementary_gates, accept_window, optimizer)
1: Input: target gate sequence, window size, max number of qubits in a window, elementary gate set, optimizer, accept
   window size
2: Output: optimized gate sequence
3: output_gates  $\leftarrow []$ 
4: full_target  $\leftarrow$  target_sequence
5: while full_target  $\neq \emptyset$  do
6:   current_target  $\leftarrow$  FIND_FIRST_BLOCK(full_target, window_length, max_qubits)
7:   current_qubits  $\leftarrow$  EXTRACTQUBITS(current_target)
8:   full_target  $\leftarrow$  REMOVE(full_target, current_target)
9:   if LENGTH(current_qubits)  $\leq 1$  or LENGTH(current_target) = 1 then
10:    APPEND(output_gates, current_target)
11:   else
12:     optimized  $\leftarrow$  QCOPT(current_target, elementary_gates, optimizer)
13:     if LENGTH(full_target) = 0 then
14:       APPEND(output_gates, optimized[:])
15:     else
16:       if LENGTH(optimized)  $\geq$  accept_window then
17:         APPEND(output_gates, optimized[:accept_window])
18:         full_target  $\leftarrow$  APPEND(optimized[accept_window+1:], full_target)
19:       else
20:         APPEND(output_gates, optimized)
21: return output_gates

```

and R_{26} . When the window hits its 12-gate cap, only those gates are optimized. The construction of this second block appears in progressively lighter shades of blue in Fig. 1. This simple example makes the rolling-horizon mechanics concrete: select the earliest feasible block (respecting length, qubit cardinality, and per-qubit closure), optimize, accept, retarget, and repeat.

V. NUMERICAL EXPERIMENTS

All MILP/MIQP formulations and the rolling-horizon algorithm are implemented in the open-source package `QuantumCircuitOpt.jl`. We omit usage details here; complete documentation and a user guide are available at <https://github.com/harshangrjn/QuantumCircuitOpt.jl>. The illustrative examples presented in this section demonstrate typical transpilation use cases and provide empirical evidence motivating several implementation choices. All experiments were run on a single machine with the following setup: *CPU: AMD EPYC 7R13 (32 cores / 64 threads, base 2.56 GHz); RAM: 128 GB; OS: Windows Server 2022 (21H2); solver: Gurobi v12.0.2 (default settings), threads: 32; Julia: 1.11.5.*

Unless stated otherwise, all runs use the total gate-count objective (18) with the baseline MILP constraints (13)–(14). We denote by “best-MILP” the baseline augmented with all additional constraints—symmetry, redundancy-elimination, and the Hermitian-conjugate (HC-1) constraints (see Sec. III). Reported run-time statistics may vary with hardware (e.g., core count)

due to the parallelism of commercial MIP solvers (e.g., Gurobi) and may also depend on the solver choice. Further details on the number of qubits, elementary gate set size, and the imposed maximum circuit length for each example are provided in Appendix A.

A. Performance of Global-Phase Conditions

In Section II we introduced the global-phase equivalence target condition (Eq. (16)), which allows $\phi \in [0, 2\pi)$. We show the implementation of this formulation using the total gate-count objective under the “best-MILP” configuration, with Hermitian-conjugate (HC-1) constraints implemented in the global-phase perspective, as described in Proposition 5. Table I reports wall-clock times; the “Exact” column corresponds to the strict $\phi = 0$ condition (Eq. (17)). The exact condition is, on average, 28.3% faster than global-phase equivalence. The gap likely stems from the larger feasible region in the global-phase formulation, which introduces combinatorial symmetries and increases branch-and-bound depth, degrading MILP solver performance. Nevertheless, when all elementary gates are encoded in $SU(2^Q)$, the exact target condition (Eq. (17)) mostly outperforms the global-phase representation, and we therefore adopt it for all subsequent examples.

TABLE I. Execution time (seconds) for two separate conditions: global-phase equivalence ($\phi \in [0, 2\pi)$) vs. exact ($\phi = 0$).

Target	Global-phase (s)	Exact (s)
Controlled- $\sqrt{X}$	17.01	12.93
Controlled-Hadamard	0.46	3.92
Magic ^a [28]	10.26	3.35
iSwap	8.57	2.39
single excitation Hadamard [29]	8.30	5.26
CNOT _{1,3}	16.52	2.34
Fredkin [30]	41.58	43.30
Miller [31]	71.09	117.7
Relative Toffoli [32]	80.55	18.31
Margolus [32]	26.62	13.72
CNOT _{4,1}	31.99	28.46
Double Peres [33]	12.96	18.67
Quantum Full Adder [31]	203.64	153.6
Double Toffoli [31]	150.87	117.64
Average	48.6	38.7

B. Speed-ups from Valid Constraints

Having identified the target constraint to use going forward, we now evaluate the impact of the valid speed-up constraints. We begin with those tied directly to the target relation, namely the Hermitian-conjugate constraints: a linear HC-1 form and a quadratic MIQP variant, HC-2, as presented in Section III C. Table II shows that these constraints can substantially reduce runtime—often by factors of two to five—though the magnitude depends strongly on the target.

Each run (including the *base* column) uses total gate-count optimization with only the necessary baseline constraints (conditions (13) to (14)) plus the identity-gate symmetry constraint (29), which we regard as fundamental. The subsequent columns add only the named constraint family to that baseline. The *best MILP* column enables all three speed-up families simultaneously, namely redundancies, equivalent pairs, and HC 1 as well.

A few takeaways are worth highlighting. First, the pruning conditions (redundancies and equivalent pairs) often help the solver dramatically—up to $11\times$ on their own (see CNOT_{1,3} with equivalent pairs). Second, in 5 of the 11 targets, enabling all three speed-up families produces a combined gain that is larger than what one might expect from simply compounding their individual effects. For example, for the Fredkin gate, a naive expectation from individual columns suggests only about a $3.4\times$ improvement, yet the full combination delivers $9.5\times$. Similarly, for the Double Peres gate, the individual contributions amount to about $3.2\times$, while the combined run achieves $6.3\times$. Finally, across our test set, these three families together yield a maximum observed improvement of $43\times$ (for CNOT_{1,3}).

These speedups have a practical consequence — as seen

in the Controlled-iSwap case — because without these constraints, the raw MILP runtimes make our approach essentially infeasible, whereas with them, the problem becomes tractable, broadening the set of target gates for which the package could be genuinely useful.

C. Circuit Depth Minimization

Having evaluated the target constraints and the benefits of the three families of speed-ups, we now illustrate with a few numerical examples how QCOpt tackles *depth optimization* directly, using a gate set similar to the one employed above. All runs use our best available constraints: the base formulation (13)–(15) together with the exact-target constraint (17), plus all speed-up constraints—redundancy constraints and the Hermitian-conjugate constraints (HC1, HC2). From the symmetry family, we include only the identity-gate symmetry (29); the equivalent-pair symmetry in (30) is not a valid constraint for depth optimization, as noted in Remark 2.

We observe that running the direct depth-minimization MIP typically requires substantially more time than gate-count optimization with this formalism, which matches expectations: depth minimization is both a harder objective and it introduces more decision variables and linking constraints than the simpler gate-budget formulation.

D. Approximate Compilation with Fibonacci Anyons

We tested our approximate formalism on a topological model — computation with Fibonacci anyons — where two closely related gate pictures exist: braiding (the braid generators σ_1, σ_2) and weaving (their powers, typically σ_1^2, σ_2^2). For single-qubit compilation, one uses three anyons, yielding a two-dimensional fusion space acted on by two braiding operators. Let $\varphi := \frac{1+\sqrt{5}}{2}$ be the golden ratio. Within an $SU(2)$ representation, the braiding operators can be written compactly as

$$\sigma_1 = e^{i\pi/10} R, \quad \sigma_2 = e^{i\pi/10} FRF, \quad \text{where} \\ R = \begin{pmatrix} e^{-4i\pi/5} & 0 \\ 0 & e^{3i\pi/5} \end{pmatrix}, \quad F = \begin{pmatrix} \varphi^{-1} & \varphi^{-1/2} \\ \varphi^{-1/2} & -\varphi^{-1} \end{pmatrix}.$$

The weaving operators are $W_1 = \sigma_1^2$ and $W_2 = \sigma_2^2$, and as the braiding ones, they do not commute ($[W_1, W_2] \neq 0$). Sequences drawn from $\{\sigma_1^{\pm 1}, \sigma_2^{\pm 1}\}$ (or equivalently from their weaves and complex conjugates) are dense in $SU(2)$, so any single-qubit unitary can be approximated to arbitrary precision. Prior work has largely searched for the best short sequences by brute force: for instance, a depth-15 weave achieving a Hadamard has reported fidelity 0.999957 [19]. To mirror approximate compilation methods in Section II G, we benchmark three objectives—ordered by increasing modeling tractability—on

TABLE II. Efficacy of valid constraints on wall-clock runtime (seconds) across circuit decompositions. Speedup is reported relative to the base run (values > 1 indicate speedup). “best MILP” = base + HC-1 + redundancy + equivalent-pair constraints.

Target	base	base + HC-1		base + HC-1 + HC-2		base + redundancies		base + equivalent pairs		best MILP	
	time (s)	time (s)	speed-up	time (s)	speed-up	time (s)	speed-up	time (s)	speed-up	time (s)	speed-up
Toffoli (with 2-qubit gates) [31]	10.78	12.36	0.9x	5.64	1.9x	10.22	1.1x	4.10	2.6x	2.81	3.8x
CNOT _{1,3}	101.39	52.53	1.9x	31.90	3.2x	25.03	4.1x	9.19	11.0x	2.34	43.3x
Fredkin	381.45	389.53	1.0x	212.70	1.8x	259.21	1.5x	160.87	2.4x	40.13	9.5x
Miller	138.31	138.88	1.0x	108.98	1.3x	134.30	1.0x	180.10	0.8x	117.70	1.2x
Relative Toffoli	182.30	94.99	1.9x	67.15	2.7x	76.79	2.4x	68.71	2.7x	18.31	10.0x
Margolus	294.27	87.85	3.3x	124.73	2.4x	243.73	1.2x	37.42	7.9x	13.72	21.4x
Quantum Fourier Transform	481.91	279.58	1.7x	89.62	5.4x	202.19	2.4x	321.47	1.5x	72.50	6.6x
Controlled-iSwap [34]	52090.81	16008.00	3.3x	14501.24	3.6x	7447.50	7.0x	10399.80	5.0x	1367.02	38.1x
Double Peres	116.73	91.58	1.3x	62.87	1.9x	121.49	1.0x	44.50	2.6x	18.67	6.3x
Quantum Full Adder	2348.01	348.47	6.7x	231.44	10.1x	1054.85	2.2x	231.06	10.2x	153.60	15.3x
Double Toffoli	1583.08	1020.68	1.6x	315.30	5.0x	986.62	1.6x	179.38	8.8x	117.64	13.5x

TABLE III. Solver times for the task of circuit depth optimization.

Target	Solve time [s]
Toffoli (with 2-qubit gates)	1.48
CNOT _{1,3}	770.66
Fredkin	82.60
Miller	114.50
Relative Toffoli	123.13
Margolus	208.10
Quantum Fourier Transform	893.79
Double Peres	23.97
Quantum Full Adder	232.60
Double Toffoli	375.91

the depth-15 instances for the Hadamard, X, and T gates; results appear in Table IV.

First, we solve the exact, phase-invariant fidelity MIQP (23), maximizing $(F(\cdot) = \alpha^2 + \beta^2)$. This non-convex objective is computationally heavier but yields certificates of optimality. On the depth-15 Fibonacci-anyon benchmarks, the MIQP recovers the best-known Hadamard and (X)-gate weaves within 200 s, substantially faster than exhaustive search (the only alternative that, to our knowledge, guarantees optimality but scales exponentially). Although mixed-integer optimization is NP-hard in the worst case, these results show that a carefully tailored model with valid constraints can solve practically relevant instances; the (T)-gate remains harder (≈ 6900 s), motivating surrogate objectives. The MIQP’s certified fidelities serve as upper bounds for the surrogate objectives evaluated below.

Second, we evaluate the linearized fidelity surrogate (24), which maximizes the real trace overlap (α) (a linear objective). When the phase quadrature ($\beta \approx 0$), maximizing (α) is equivalent to maximizing fidelity, and indeed it matches the MIQP on the (X)-gate and is near-optimal on the others. However, its runtimes are not uniformly better—and degrade notably for the (T) target (Table IV)—motivating a third, more scalable objective.

Third, we evaluate the Frobenius-error objective (26),

realized via the linear outer approximation (28) using K tangents per error term on a uniform grid in $[-\epsilon, \epsilon]$. This yields an MILP. By Prop. 4, minimizing $\|E\|_F^2$ is exactly equivalent to maximizing α , while offering a significant run time advantage. With $K = 5$ tangents per entry, decreasing ϵ tightens the outer approximation; runtimes increase monotonically while fidelities typically improve (Table IV). At $\epsilon = \frac{1}{8}$, the surrogate reaches $F = 0.999957$ for the Hadamard in 8.25 s and $F = 0.999990$ for the X-gate in 20.42 s, essentially matching the MIQP certified optima at a fraction of the time. For the T -gate, the surrogate yields high-quality but suboptimal fidelity—e.g., $F = 0.999739$ at $\epsilon = \frac{1}{16}$ versus 0.999917 from the MIQP—reflecting that controlling $\|E\|_F^2$ (equivalently, α) cannot always suppress a residual phase quadrature β .

For reference, the exact-fidelity MIQP produced the following certified depth-15 weaves:

$$\begin{aligned}
 H &\simeq \sigma_1^{-4} \sigma_2^{+2} \sigma_1^{-2} \sigma_2^{+2} \sigma_1^{-2} \sigma_2^{-2} \sigma_1^{+2} \sigma_2^{-4} \sigma_1^{-2} \sigma_2^{+2} \sigma_1^{+2} \sigma_2^{-2} \sigma_1^{-2}, \\
 X &\simeq \sigma_2^{+2} \sigma_1^{-4} \sigma_2^{+2} \sigma_1^{-4} \sigma_2^{+2} \sigma_1^{-4} \sigma_2^{+2}, \\
 T &\simeq \sigma_1^{+2} \sigma_2^{-2} \sigma_1^{+2} \sigma_2^{+4} \sigma_1^{-2} \sigma_2^{+2} \sigma_1^{-2} \sigma_2^{-4} \sigma_1^{+2} \sigma_2^{-2} \sigma_1^{-4} \sigma_2^{-2}.
 \end{aligned}$$

E. RHO with 3-body Interactions

We test the rolling-horizon optimizer on circuits built from three-body ZZZ interactions placed on hyperedges of a graph. Such 3-local Ising terms are natural in spin models with multi-spin couplings [35], and also appear as parity checks in subsystem/topological quantum error-correction architectures that use three-qubit XXX/ZZZ measurements [36].

Throughout, we take the phase-rotation convention

$$R_{ZZZ}(\theta) = \exp(-i\frac{\theta}{2} Z \otimes Z \otimes Z), \quad (35)$$

so that a single $R_{ZZZ}(\theta)$ may be compiled as a short parity ladder: compute the Z -parity of two controls into an accumulator with two CNOTs, apply a single-qubit $R_Z(\theta)$ on the accumulator, and uncompute. In a cir-

TABLE IV. Approximate-compilation results for depth-15 Fibonacci weaves across three targets: the number of outer approximators is fixed $K = 5$, varying the entrywise error bound ϵ on the complex error matrix E (i.e., $|E_{ij}| \leq \epsilon$).

Method	H -gate		X -gate		T -gate	
	time (s)	fidelity	time (s)	fidelity	time (s)	fidelity
outer-approximation ($\epsilon = 1$):	1.06	0.945974	1.38	0.932852	1.01	0.973666
outer-approximation ($\epsilon = 1/2$):	1.83	0.987268	1.68	0.985019	1.49	0.979530
outer-approximation ($\epsilon = 1/4$):	6.86	0.992066	4.49	0.995352	9.67	0.998256
outer-approximation ($\epsilon = 1/8$):	8.25	0.999957	20.42	0.999990	13.29	0.998153
outer-approximation ($\epsilon = 1/16$):	98.46	0.999921	56.12	0.999990	846.51	0.999739
linearized fidelity (MILP):	217.66	0.999921	76.99	0.999990	30293	0.999739
exact fidelity (MIQP):	198.41	0.999957	191.06	0.999990	6897.8	0.999917

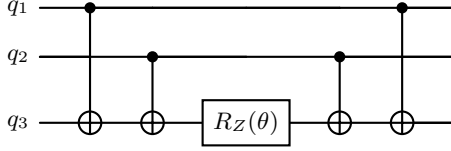


FIG. 2. Parity-ladder compilation of a three-body interaction $R_{ZZZ}(\theta)$ in a quantum circuit.

circuit diagram, therefore, one interaction term is shown in Fig. 2.

We consider the full hypergraph on 5 qubits, $K_5^{(3)}$, where there are $\binom{5}{3} = 10$ hyperedges, and apply the maximally entangling phases $R_{ZZZ}(\pi/2)$ in lexicographic order to each 3-subset $\{i, j, k\} \subset \{1, \dots, 5\}$ (with $1 < 2 < 3 < 4 < 5$ ordering). As a naive target seed, we simply concatenated the decomposition above for each hyperedge (ten copies of a 5-gate pattern), and we restricted the elementary gate set to $\{\text{CNOT}, H, S\}$ with $S = R_Z(\pi/2)$. Running RHO with parameters `window_length` = 10, `accept_window` = 5, `max_qubits` = 4, the optimizer produced a 36-gate circuit, saving 14 gates by sharing intermediate parities across successive terms. Although this is a remarkable advantage, this sequence is still suboptimal. A second RHO pass seeded by the 36-gate solution further reduced the count to 32 gates, indicating that while exact MILP becomes intractable at these sizes, the RHO heuristic can still realize substantial improvements on larger instances.

F. RHO's Performance on a Larger Circuit

We illustrate the performance of RHO—and, in particular, its dependence on the window parameters—on a four-qubit benchmark from Ref. [37]. The circuit (Fig. 3) comprises 15 gates from $\{X, \text{CNOT} = \text{CX}, \text{CCX}, \text{CCCX}\}$; our objective is to obtain an implementation over the canonical Clifford+T set $\{H, T, \text{CNOT}\}$.

To make the instance compatible with the RHO wrap-

per, we first translate each gate to the canonical set. We represent an X as HT^4H (two Hadamards and four T gates), keep CNOT as is, and use a standard T -optimal Toffoli with 6 CNOTs, 7 T -type gates (counting $T/T^\dagger$), and 2 Hadamards. For the four-controlled NOT (CCCX), we follow a Barenco-style decomposition [38] into 6 CNOTs and 7 controlled $-\sqrt{X}$ gates; each controlled $-\sqrt{X}$ is then expressed using 2 Hadamards, 2 CNOTs, and 3 T -type gates. After local H -cancellations this yields a 43-gate realisation of CCCX over $\{H, T, \text{CNOT}\}$. Overall, the benchmark translates to a seed of 142 one- and two-qubit gates on which we run the RHO wrapper. As a heuristic baseline, transpiling the same 142-gate seed with Qiskit 2.2.0 [39] reduces it to 132 gates.

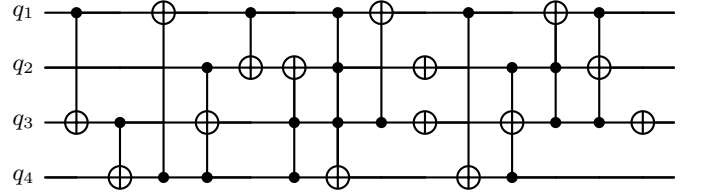


FIG. 3. Four-qubit benchmark circuit used to probe RHO parameter sensitivity.

We then explore three representative settings for the RHO parameters `window_length` (L_W), `accept_window` (A_W), and `max_qubits` (Q_W):

- *Large blocks with moderate qubit budget:* $(L_W, A_W, Q_W) = (10, 5, 3)$. This ambitious choice ran for > 24 h in total, with several windows timing out after > 4 h (solutions found but without optimality certificates). The pass reduced the seed from 142 to 122 gates. Given the exaggerated runtime, we deemed iterative passes infeasible in this configuration.
- *Very small qubit budget:* $(10, 5, 2)$. Here, most windows are too small to expose non-local simplifications, but the runtime is < 2 min. The first pass

reduced 142→130 gates, a second pass reached 126, and a third pass produced no further improvement.

- *Balanced choice:* (7,5,3). This setting provided the best trade-off. The first pass reduced 142→126, a second pass reached 116, and subsequent passes did not improve further. Each pass completed within 90 min.

Overall, we find that (i) overly aggressive windows with insufficient qubit caps incur prohibitive solve times per window; (ii) small Q_W makes passes fast but myopic; and (iii) a balanced choice (here, $L_W = 7$, $A_W = 5$, $Q_W = 3$) yields substantial reductions with manageable runtime, enabling effective multi-pass refinement. For further context, as Qiskit produces 132 gates on the same instance, whereas our best RHO setting reaches 116 (16 fewer than Qiskit, $\approx 12.1\%$ reduction; 26 fewer than the seed, $\approx 18.3\%$).

These experiments illustrate that RHO can exploit structure that is invisible to a purely local, per-term compilation, generating shorter circuits even when exact global optimization methods on the entire circuit are computationally intractable.

VI. CONCLUSIONS

In conclusion, we present a depth-aware mixed-integer programming framework for quantum circuit compilation that treats global phase with linear constraints, optimizes depth via explicit scheduling, and sharply tightens the search space with domain-specific valid inequalities. Empirically, enforcing the exact $SU(2^Q)$ target and enabling our cuts reduces wall-clock time by large factors—often an order of magnitude and up to $\sim 40\times$ —making exact certification practical on nontrivial medium-scale circuits. For approximate synthesis, linear surrogates expose accuracy–time trade-offs, while a non-convex, phase-invariant fidelity objective attains certified, best-known solutions (e.g., Hadamard with depth-15 Fibonacci-anyon weaves with $F \approx 0.999957$), and faster linearized objectives deliver the same fidelity faster. Beyond exact MIP’s reach, a rolling-horizon strategy that rolls primarily in time and enforces per-qubit closure preserves local context and yields iterative gains, including parity sharing. Empirically, RHO provides substantial gate savings on multi-body parity circuits (e.g., $50 \rightarrow 32$ gates over two passes). On a 142-gate seed circuit, RHO produces 116 gates (an 18.3% reduction, 12.1% fewer than a Qiskit baseline).

Methodologically, the results show that (i) linear handling of global phase and explicit depth scheduling make exact depth optimization practical at *modest scales*; (ii) well-chosen, domain-aware constraints can shift instances from unsolvable to tractable; and (iii) fidelity-driven approximate formulations admit both exact (non-convex) and convex/MILP surrogates with provable relationships. Practically, the same optimization stack ac-

TABLE V. Parameters used for the targets in Sec. V.

target	Q	$ \mathbb{G} $	P
<i>Two-qubit targets</i>			
Controlled- $\sqrt{X}$	2	9	7
Controlled-Hadamard	2	32	5
Magic	2	73	4
iSwap	2	9	10
single-excitation Hadamard	2	14	5
<i>Three-qubit targets</i>			
Toffoli (with 2-qubit gates)	3	9	5
CNOT _{1,3}	3	14	8
Fredkin	3	11	7
Miller	3	7	10
Relative Toffoli	3	7	9
Margolus	3	12	7
Quantum Fourier Transform	3	17	7
Controlled-iSwap	3	7	12
<i>Four-qubit targets</i>			
CNOT _{4,1}	4	6	10
Double Peres	4	9	7
Quantum Full Adder	4	11	7
Double Toffoli	4	10	7

commodates hardware-aware costs, topology constraints, and routing-aware compilations, offering a single point of control over objectives that matter in both the NISQ and fault-tolerant regimes.

Overall, the approach unifies provable optimality and principled heuristics within the open-source **QCopt** package, offering a practical path to hardware-aware compilation; promising extensions include richer cut libraries, adaptive horizons and warm starts, and tighter integration with state-of-the-art hardware-compatible gates.

ACKNOWLEDGMENTS

The authors gratefully acknowledge support from the U.S. Department of Energy (DOE) Quantum Computing Program, sponsored by the Los Alamos National Laboratory (LANL) Information Science & Technology Institute, as well as funding from LANL’s Laboratory Directed Research and Development (LDRD) program under project “20230091ER: Learning to Accelerate Global Solutions for Non-convex Optimization”. Z.S. also gratefully acknowledges stimulating discussions with Gavin Brennen and Max Hunter Gordon, and support from the Sydney Quantum Academy.

Appendix A: Auxiliary Parameters for Example Targets

Table V from this appendix collects the parameters used in Sec. V. Here Q is the number of qubits for the target, $|\mathbb{G}|$ is the cardinality of the elementary gate set

made available to the optimizer, and P is the maximum

gate count allowed for the synthesized circuit.

-
- [1] N. C. Jones, R. Van Meter, A. G. Fowler, P. L. McMahon, J. Kim, T. D. Ladd, and Y. Yamamoto, Layered Architecture for Quantum Computing, *Physical Review X* **2**, 031007 (2012).
 - [2] E. T. Campbell, B. M. Terhal, and C. Vuillot, Roads towards fault-tolerant universal quantum computation, *Nature* **549**, 172 (2017).
 - [3] E. Younis, K. Sen, K. Yelick, and C. Iancu, QFAST: Conflating Search and Numerical Optimization for Scalable Quantum Circuit Synthesis, in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, Broomfield, CO, USA, 2021) pp. 232–243.
 - [4] P. Rakyta, G. Morse, J. Nádori, Z. Majnay-Takács, O. Mencer, and Z. Zimborás, Highly optimized quantum circuits synthesized via data-flow engines, *Journal of Computational Physics* **500**, 112756 (2024), arXiv:2211.07685 [quant-ph].
 - [5] R. Duncan, A. Kissinger, S. Perdrix, and J. Van De Wetering, Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus, *Quantum* **4**, 279 (2020).
 - [6] F. J. R. Ruiz, T. Laakkonen, J. Bausch, M. Balog, M. Berekatani, F. J. H. Heras, A. Novikov, N. Fitzpatrick, B. Romera-Paredes, J. Van De Wetering, A. Fawzi, K. Meichanetzidis, and P. Kohli, Quantum circuit optimization with AlphaTensor, *Nature Machine Intelligence* **7**, 374 (2025).
 - [7] M. Amy, D. Maslov, M. Mosca, and M. Roetteler, A Meet-in-the-Middle Algorithm for Fast Synthesis of Depth-Optimal Quantum Circuits, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **32**, 818 (2013).
 - [8] N. J. Ross and P. Selinger, Optimal ancilla-free Clifford+T approximation of z-rotations, *Quantum Information and Computation* **16**, 901 (2016).
 - [9] M. Amy, D. Maslov, and M. Mosca, Polynomial-Time T-Depth Optimization of Clifford+T Circuits Via Matroid Partitioning, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **33**, 1476 (2014).
 - [10] G. Nannicini, L. S. Bishop, O. Günlük, and P. Jurcevic, Optimal Qubit Assignment and Routing via Integer Programming, *ACM Transactions on Quantum Computing* **4**, 1 (2023).
 - [11] H. Nagarajan, O. Lockwood, and C. Coffrin, QuantumCircuitOpt: An Open-source Framework for Provably Optimal Quantum Circuit Design, in *2021 IEEE/ACM Second International Workshop on Quantum Computing Software (QCS)* (IEEE, St. Louis, MO, USA, 2021) pp. 55–63.
 - [12] E. R. Henderson, H. Nagarajan, and C. Coffrin, Exploring Non-Linear Programming Formulations in QuantumCircuitOpt for Optimal Circuit Design, in *2022 IEEE/ACM Third International Workshop on Quantum Computing Software (QCS)* (IEEE, Dallas, TX, USA, 2022) pp. 36–42.
 - [13] L. Gurobi Optimization, *Gurobi optimizer reference manual* (2025).
 - [14] IBM - ILOG: CPLEX optimization studio 12.2. url: <http://www.ilog.com/products/cplex>, .
 - [15] T. Koch, T. Berthold, J. Pedersen, and C. Vanaret, Progress in mathematical programming solvers from 2001 to 2020, *EURO Journal on Computational Optimization* **10**, 100031 (2022).
 - [16] H. Nagarajan, M. Lu, E. Yamangil, and R. Bent, Tightening mccormick relaxations for nonlinear programs via dynamic multivariate partitioning, in *International conference on principles and practice of constraint programming* (Springer, 2016) pp. 369–387.
 - [17] C. M. Dawson and M. A. Nielsen, *The Solovay-Kitaev algorithm* (2005), arXiv:quant-ph/0505030.
 - [18] O. Parzanchevski and P. Sarnak, Super-Golden-Gates for PU(2), *Advances in Mathematics* **327**, 869 (2018), publisher: Elsevier BV.
 - [19] M. T. Rouabah, N. E. Belaloui, and A. Tounsi, Compiling single-qubit braiding gate for Fibonacci anyons topological quantum computation, *Journal of Physics: Conference Series* **1766**, 012029 (2021), publisher: IOP Publishing.
 - [20] B. Field and T. Simula, Introduction to topological quantum computation with non-Abelian anyons, *Quantum Science and Technology* **3**, 045004 (2018).
 - [21] S. Khatri, R. LaRose, A. Poremba, L. Cincio, A. T. Sornborger, and P. J. Coles, Quantum-assisted quantum compiling, *Quantum* **3**, 140 (2019).
 - [22] L. Hormozi, G. Zikos, N. E. Bonesteel, and S. H. Simon, Topological quantum compiling, *Physical Review B—Condensed Matter and Materials Physics* **75**, 165310 (2007).
 - [23] R. J. Baxter, *Exactly solved models in statistical mechanics* (Elsevier, 2016).
 - [24] B. Peng, S. Gulania, Y. Alexeev, and N. Govind, Quantum time dynamics employing the yang-baxter equation for circuit compression, *Physical Review A* **106**, 012412 (2022).
 - [25] S. H. Simon, N. E. Bonesteel, M. H. Freedman, N. Petrovic, and L. Hormozi, Topological Quantum Computing with Only One Mobile Quasiparticle, *Physical Review Letters* **96**, 070503 (2006).
 - [26] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, *Nature* **549**, 242 (2017).
 - [27] L. Leone, S. F. Oliviero, L. Cincio, and M. Cerezo, On the practical usefulness of the Hardware Efficient Ansatz, *Quantum* **8**, 1395 (2024).
 - [28] F. Vatan and C. Williams, Optimal quantum circuits for general two-qubit gates, *Physical Review A* **69**, 032315 (2004).
 - [29] J. M. Arrazola, O. Di Matteo, N. Quesada, S. Jahangiri, A. Delgado, and N. Killoran, Universal quantum circuits for quantum chemistry, *Quantum* **6**, 742 (2022).
 - [30] J. A. Smolin and D. P. DiVincenzo, Five two-bit quantum gates are sufficient to implement the quantum Fredkin gate, *Physical Review A* **53**, 2855 (1996).

- [31] W. Hung, Xiaoyu Song, Guowu Yang, Jin Yang, and M. Perkowski, Optimal synthesis of multiple output Boolean functions using a set of quantum gates by symbolic reachability analysis, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **25**, 1652 (2006).
- [32] D. Maslov, Advantages of using relative-phase Toffoli gates with an application to multiple control Toffoli optimization, *Physical Review A* **93**, 022311 (2016).
- [33] A. Peres, Reversible logic and quantum computers, *Physical Review A* **32**, 3266 (1985).
- [34] S. E. Rasmussen and N. T. Zinner, Simple implementation of high fidelity controlled- i swap gates and quantum circuit exponentiation of non-Hermitian gates, *Physical Review Research* **2**, 033097 (2020).
- [35] R. J. Baxter and F. Y. Wu, Exact Solution of an Ising Model with Three-Spin Interactions on a Triangular Lattice, *Physical Review Letters* **31**, 1294 (1973).
- [36] S. Bravyi, G. Duclos-Cianci, D. Poulin, and M. Suchara, Subsystem surface codes with three-qubit check operators, *Quantum Information and Computation* **13**, 963 (2013).
- [37] O. Golubitsky and D. Maslov, Reversible logic synthesis benchmarks page: 4b15g-1, https://reversiblebenchmarks.github.io/4b15g_1.html (2011), accessed: Sep. 13, 2025.
- [38] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, *Physical Review A* **52**, 3457 (1995).
- [39] G. Aleksandrowicz *et al.*, *Qiskit: An Open-source Framework for Quantum Computing* (2019).