Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.083187

ARTICLE

Quantum-Inspired Optimization with Hamming-Distance Reinforcement for Hypercube-Encoded Reversible Circuit Synthesis

Yu-Chi Jiang[1,2,*]

[1]Department of Computer Science and Information Engineering, National University of Tainan, Tainan, 700301, Taiwan
[2]Department of Communication Engineering, National Central University, Taoyuan, 320317, Taiwan

*Corresponding Author: Yu-Chi Jiang. Email: ycjiang@mail.nutn.edu.tw

Received: 30 March 2026; Accepted: 23 July 2026; Published: 15 September 2026

ABSTRACT: Quantum logic reversible synthesis is a fundamental operation in quantum computing. One of the most challenging issues in this field resides in navigating the immense search space to synthesize the most compact circuit configurations, which are critical for realizing reliable, noise-free, and error-free quantum computing systems. To address this challenge, this study proposes a novel hypercube-encoded quantum-inspired optimization framework to formulate the synthesis task as a trajectory-finding process. This structure-informed domain knowledge transformation delivers exceptional search direction guidance, moving away from blind, black-box exploration. Specifically, by mapping the reversible functions onto the hypercube architecture, the framework embeds explicit dual Hamming-distance (HMD) guidance metrics into a global-best guided quantum-inspired tabu search (GQTS) engine. To minimize computational cost and enhance search efficiency, the framework incorporates a domain-informed initialization and couples a streamlined two-particle configuration with a global-best mechanism, thereby amplifying the efficiency of the underlying quantum-inspired updating mechanism to escape local optima and rapidly converge once a successfully synthesized superior path is locked. Under an online step relaxation mechanism, the framework preserves exceptional structural optimization flexibility without altering the underlying hypercube representation. The framework's significance is rigorously evaluated against both rigid hypercube rule-based methods and generic randomized search-based heuristics, using gate count and exact-optimality as primary evaluation metrics. Extensive ablation studies first validate the individual and synergistic contributions of each core algorithmic component. Experimental results demonstrate that for the complete set of 3-bit reversible functions, the proposed HMD-guided GQTS (HMD-GQTS) framework achieves over 98% exact-optimal circuits in a single fine-tuned sweep, with selective retries attaining 100% exhaustive optimal coverage. Furthermore, for typical 4-bit benchmark instances, the method consistently delivers competitive gate counts, matching or improving upon previous hypercube-based outcomes. Through the seamless integration of structure-informed guidance and coordinated optimization mechanisms, the framework preserves exceptional structural flexibility and algorithmic robustness, offering an effective, low-cost, and highly scalable avenue for reversible circuit synthesis.

KEYWORDS: Quantum logic reversible synthesis; reversible circuit synthesis; quantum-inspired optimization; hypercube model; Hamming distance; logic optimization; electronic design automation

1 Introduction

Reversible circuits are a cornerstone of quantum computation because quantum circuits are governed by unitary transformations and are therefore reversible in nature [1,2]. Reversible circuits are also widely

regarded as an enabling basis for low-power and energy-aware next-generation computing technologies [3,4]. However, the same reversibility constraint that enables exact information preservation also makes synthesis substantially more difficult. Given a reversible function, one must construct a functionally exact circuit while minimizing the gate count. Since there is not yet a comprehensive algorithm that can synthesize the most compact circuit for all reversible functions, the synthesis framework remains diversified to cope with different problem characteristics. From this perspective, search optimization is attractive because it provides a flexible way to organize the synthesis process, especially when exact solutions are difficult to obtain efficiently and the design setting may call for adjustments in guidance, evaluation, or gate-library choices.

Quantum-inspired optimization [5,6] is therefore a natural candidate for this formulation. By introducing the simulation of quantum properties, such as the uncertainty of superposition, while remaining practically executable on classical machines, quantum-style search becomes appealing for improving search efficiency [7]. This optimization line has also been applied to reversible circuit synthesis [8] and, more recently, to automatic quantum circuit optimization [9]. In the present study, the importance of this framework lies in offering a tunable optimization mechanism that can be integrated with a reversible circuit structure.

To make such an optimization process effective, it still requires guidance that reflects the internal structure of reversible functions rather than relying only on generic black-box fitness evaluation. A prior hypercube-based viewpoint for quantum Boolean circuit synthesis [10] provides an important foundation in this direction. That study showed that the hypercube can serve as a structurally meaningful representation of a reversible function, enabling one to observe circuit evolution, evaluate current synthesis states, and interpret swap actions in a visually and mathematically coherent way.

Inspired by that perspective, the present paper adopts the hypercube as the encoding foundation of reversible synthesis and extracts two guidance indicators with complementary structural roles based on Hamming distance (HMD) aggregation: the total Hamming distance (THD), which captures global mismatch, and the adjacent Hamming distance (AHD), which summarizes local discrepancy over the adjacent neighborhood. These indicators are then embedded into a quantum-inspired optimization process to guide the search toward shorter exact circuits.

Based on this idea, this study introduces the first framework to reformulate hypercube-centered reversible synthesis from a deterministic, rule-based constructive approach into a global, trajectory-based heuristic optimization framework. To navigate this vast permutation space efficiently, this study proposes a set of novel strategies that provide both adaptive guidance and boundary flexibility to ensure search convergence. Specifically, our approach introduces domain-informed topological steering via Hamming-distance measures to provide effective evaluation guidance, implements a multi-stage search adjustment to accelerate convergence, and incorporates a dynamic boundary mechanism to overcome non-synthesizable structural bottlenecks. Through these coordinated advancements, the proposed method successfully extends the hypercube-based synthesis line toward a next-generation optimization paradigm.

The three main contributions of this paper are summarized as follows.

1. This study introduces the first framework to reformulate hypercube-centered synthesis from a rule-based constructive approach [10] into a trajectory-based heuristic optimization framework. By embedding domain-informed, non-uniform initializations and a competitive winner-loser updating mechanism, the proposed method successfully achieves an adaptive, structure-guided global search.
2. The structure-informed domain knowledge incorporates dual Hamming-distance indicators (THD and AHD) as explicit guidance signals rather than relying on black-box search, directly coupling the internal geometric properties of the discrete permutation space with the Q-matrix evolution. This integration

provides structurally meaningful quality information at both global and local levels to precisely steer the candidate synthesis trajectories.

3. A flexible search optimization architecture equipped with a novel two-phase step-relaxation mechanism is proposed. This mechanism dynamically scales the circuit cost budget during runtime to prevent trajectories from becoming trapped in non-synthesizable local dead ends, ensuring robust convergence under tight gate constraints.

The remainder of this paper is organized as follows. Section 2 reviews the most relevant literature on reversible circuit synthesis and optimization based approaches. Section 3 introduces the reversible circuit and gate-library concepts required by the proposed method. Section 4 presents the proposed method, including the hypercube-reversible correspondence, the THD/AHD-guided formulation, and the hypercube encoded GQTS procedure. Section 5 reports the experimental setting and benchmark results. Finally, Section 6 concludes the paper.

2 Related Work

Quantum computing has faced several challenges in the phase of practical circuit realization, including fidelity assessment [11] and hierarchical scheduling [12]. Before tackling these challenging topics, understanding the reversibility of quantum computing is a fundamental prerequisite for realizing quantum logic circuits. Reversible logic is important from both quantum computing and low-energy computing perspectives. Since quantum circuits are implemented by unitary transformations [13], they are reversible by nature; hence, reversible circuit synthesis is a basic issue in quantum circuit realization and optimization [1,2]. From the thermodynamic viewpoint, the low-energy motivation of reversible computing is rooted in Landauer's principle, which relates irreversible information loss to heat dissipation [14], and Bennett's result that logically reversible computation can in principle avoid this information-loss-induced dissipation [15]. This motivation remains visible in recent circuit-level studies, including reversible metal-oxide-semiconductor (MOS) current-mode implementations [3] and ultra-energy-efficient reversible quantum-dot cellular automata (QCA) sequential circuits with explicit energy-dissipation analysis [4].

The literature on reversible circuit synthesis is broad and methodologically diverse. Early design automation studies developed exact search and gate-cost-aware synthesis formulations [16], transformation-based synthesis [17], and template-based optimization for reversible Toffoli networks [18]. These method families are summarized in the survey of Saeedi and Markov [19]. More recent work further extends this line through multi-commodity network formulations for optimal synthesis [20], optimization of reversible logic networks through gate sharing [21], and asymptotically optimal constructive algorithms for large reversible circuits [22]. These methods typically rely on fixed synthesis principles or construction rules, which makes adaptation less direct when the search objective or gate library changes.

When the search space becomes too large for direct constructive control or exhaustive optimization, metaheuristic search becomes a practical alternative. Given the diversity of reversible functions and synthesis objectives, no single constructive strategy is known to perform best across all practical settings. In that sense, search-driven methods offer a flexible and adaptive alternative for navigating the discrete synthesis space. Genetic search has been applied to reversible circuit synthesis over the NCT library [23], metaheuristic reordering has been integrated into binary decision diagram (BDD)-based reversible logic optimization [24], and ant colony optimization (ACO) based search has also been explored for reversible circuit synthesis improvement [25,26]. Within the quantum-inspired line, the quantum-inspired tabu search (QTS) algorithm has been applied directly to reversible logic circuit synthesis [8], has been established as an effective optimizer for classical combinatorial problems [7], and has been further extended to reversible circuit optimization [9]

and quantum design automation [27]. These studies suggest that QTS provides a viable optimization framework for circuit synthesis in which the search space is large.

To effectively navigate such a large discrete space, relying solely on robust optimization techniques is often insufficient; exploiting the underlying problem structure is equally critical to making the search process meaningful. This philosophy aligns with the emerging paradigm of informed machine learning [28], where data patterns, physical laws, and domain-specific heuristics are embedded to reduce the search space and avoid invalid trajectories. Particularly, structure-informed multi-scale hierarchical machine learning models [29] are increasingly adopted to achieve rapid performance in complex science and engineering problems. Analogous to these complex scientific applications, quantum Boolean circuit synthesis also faces the challenge of navigating a massive search space. While a prior study [10] successfully introduced the hypercube as a valuable structural representation for reversible functions, it primarily relied on fixed constructive rules. To bridge the gap between search heuristics and structural constraints, the present work is the first to elevate this hypercube viewpoint into a truly structure-informed optimization framework. By retaining the hypercube encoding and integrating its Hamming distance-style guidance into a tunable QTS-based engine, our method explicitly uses topological domain rules to restrict the massive permutation space. This structure-guided search allows the optimizer to avoid invalid trajectories and effectively converge on highly compact and consistent circuit configurations.

3 Background

This section introduces the background knowledge and notation usage in reversible circuit synthesis, including the reversible function model, the gate library for synthesis, and the hypercube interpretation used throughout the paper.

3.1 Characteristics of Reversible Circuits

A reversible circuit on n lines realizes a bijection on the binary state space $F : \{0,1\}^n \to \{0,1\}^n$, and therefore preserves information exactly [30]. Equivalently, every output pattern is produced by exactly one input pattern, so the function can be written as a permutation of the 2^n basis states. Let $V_n = \{0,1\}^n$ be the set of all n-bit binary vectors. A reversible Boolean function is a bijection $F : V_n \to V_n$. After identifying each bit string with its integer index in $\{0, \ldots, 2^n - 1\}$, a reversible function can be represented by a permutation vector $F = [F(0), F(1), \ldots, F(2^n - 1)]$.

A reversible circuit C realizes the reversible function F by being composed of a sequence of reversible gates g_i, denoted as $C = g_L \circ g_{L-1} \circ \cdots \circ g_1$, where L is the circuit length, and the composition ensures that C remains bijective. These structural constraints distinguish reversible synthesis from ordinary logic synthesis. The number of inputs and outputs must match, fan-out is not allowed in the ideal reversible model, and every gate choice must preserve a one-to-one mapping throughout the synthesis process. As a result, reversible circuit synthesis is naturally permutation-oriented: one does not only care whether the final Boolean relation is correct, but also whether intermediate transformations remain reversible and resource-efficient.

3.2 Generalized Toffoli Gate for Reversible Circuit

The generalized Toffoli (GT) interpretation is defined as follows. A n-bit totally controlled GT gate g consists of a set of control bits S_c and a target bit t, denoted as $g = (S_c,\ t)$, where $S_c \subseteq \{1, \ldots, n\} \backslash \{t\}$ is the control set. A generalized Toffoli gate toggles the target bit only when the control literals match a prescribed pattern. The λ records the controlled match status, which consists of two parts, the state value and its control polarity in position k. Let $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, and t be the target line. For the state value x_k in position k (the k_{th} qubit), the polarity of each control literal is denoted as p_k, where 1 is positive and 0 is negative.

The $\lambda(x_k, p_k)$ represents the match status of all the control literals in each qubit k. The action of the circuit g input x at each line i can be expressed as Eq. (1). Only when all control records match in λ, the circuit flips the target bit by the negation law $x_t' = x_t \oplus 1$. Fig. 1 provides the standard gate example of a NOT gate (special case for $S_c = \varnothing$), with one control a CNOT gate, and with two or more controls, a multi-control Toffoli gate. In the totally controlled case used to interpret hypercube edge swaps, all non-target bits act as controls, and the target toggles only when the non-target patterns are matched.

$$g(x_i, t) = \begin{cases} x_i, & i \neq t, \\ x_t \oplus \prod_{k \in S_c} \lambda(x_k, p_k), & i = t, \end{cases} \tag{1}$$

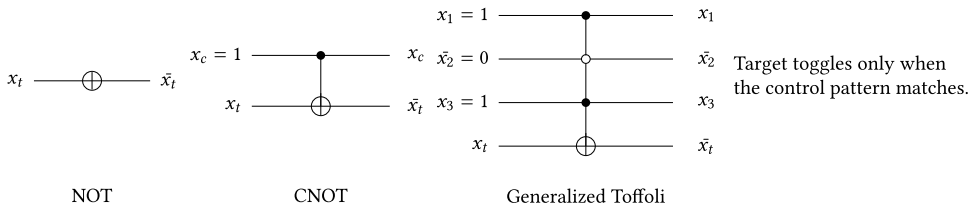

Figure 1: In an n-bit reversible function, a GT gate consists of a single target bit and a variable number of control bits. The target bit toggles if and only if the specified control pattern is satisfied, ensuring that at most one bit is flipped per operation. The NOT gate represents a special case where the control set is empty.

3.3 Hypercube Structure Reversible Function Correspondence

The n-bit reversible function is isomorphic to the n-dimensional hypercube [10,31]. The n-dimensional hypercube is the graph $Q_n = (V_n, E_n)$, where $(u, v) \in E_n$ if and only if u and v differ in exactly one bit position, e.g., $u = \{0, 1, 0\}$, $v = \{1, 1, 0\}$. Let e_j denote the j-th unit vector, e.g., $e_0 = \{1, 0, 0\}$, $e_1 = \{0, 1, 0\}$, $e_2 = \{0, 0, 1\}$. Consequently, every edge in Q_n can be expressed in the form $(u, u \oplus e_j)$, implying each vertex $u \in V_n$ has n adjacent vertices, each differing by a single bit. The 3-bit hypercube illustration is displayed in Fig. 2.

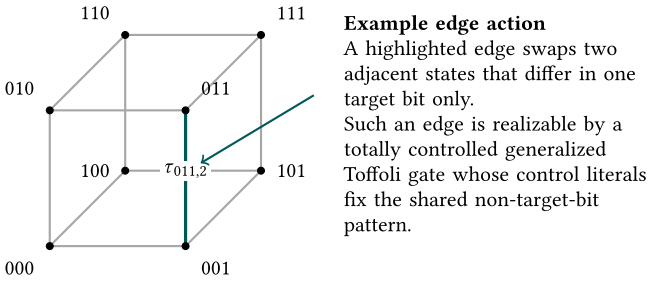

Figure 2: Three-dimensional hypercube model for 3-bit reversible synthesis. Each vertex is a basis state, and each edge represents an admissible adjacent transposition. The highlighted edge illustrates how a single hypercube edge corresponds to one reversible gate action.

For a vertex $u \in V_n$ and bit index j, the adjacent transposition $\tau_{u,j}$ is defined to swap the function values (output states) stored at the two adjacent input addresses, u and $u \oplus e_j$. Thus, a reversible function may be viewed as a labeling of hypercube vertices with output states, where each valid synthesis move corresponds to an adjacent transposition on the hypercube. This mapping serves as the fundamental encoding for our proposed method. Furthermore, if two vertices u and $v = u \oplus e_j$ differ only on the target bit j, then the

transposition $\tau_{u,j}$ is realizable by a totally controlled GT gate whose control literals are set to fix all bits, other than the target bit j, to the shared pattern of u and v. Therefore, each hypercube edge swap corresponds to a valid reversible gate action. This is exactly why the GT gate library is analytically convenient for hypercube-based synthesis: every valid edge move corresponds to a clearly interpretable reversible gate action.

For the 3-bit case, this correspondence can be listed exhaustively. Fig. 3 labels all twelve edges of the three-dimensional hypercube and shows the totally controlled GT circuit associated with each edge. This explicit mapping is useful because it makes the hypercube representation and the circuit implementation view completely consistent at the level of individual synthesis moves.

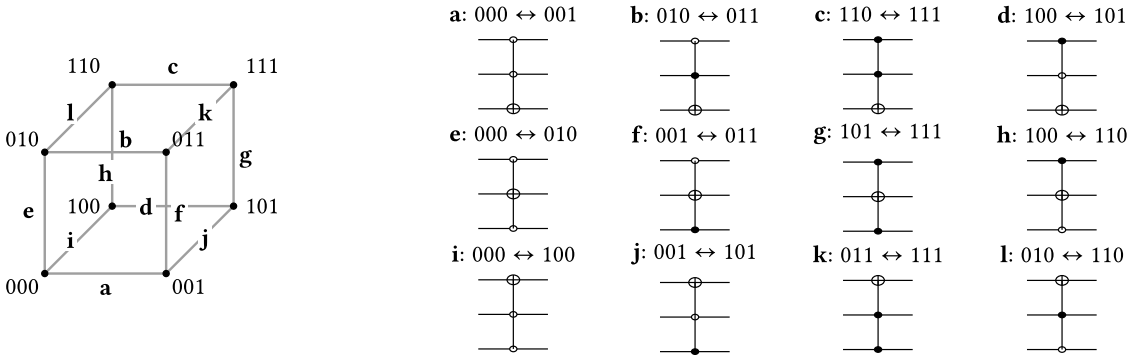

Figure 3: Complete edge-to-circuit correspondence for the 3-bit hypercube. The left panel labels the twelve hypercube edges by a through l. The right panel lists the totally controlled GT circuit associated with each edge. In every circuit, the three horizontal lines are ordered from top to bottom as x_1, x_2, and x_3.

4 Proposed Method

This study introduces the first framework that reformulates hypercube-centered reversible synthesis from a rule-based constructive approach into a trajectory-based search-driven optimization framework. To effectively explore this space, a quantum-inspired tabu search (QTS) algorithm is adopted as the core optimization engine, leveraging its proven capability in quantum and reversible circuit synthesis [8,9] and other complex combinatorial tasks [5]. The main innovation is that the hypercube structure is not used only as a representation, but is further transformed into trajectory-level search guidance through Hamming-distance observations that capture both global and local synthesis variations. Based on these observations, the proposed method adaptively updates trajectory preference, adjusts convergence behavior, and flexibly relaxes the circuit budget when necessary, making the synthesis process as the trajectory finding. By incorporating this structure-informed domain knowledge, the framework effectively guides the search direction. Concurrently, using the Q-matrix for trajectory path encoding combined with direct winner-loser updates provides exceptional flexibility, significantly boosting overall search efficiency. The remainder of this section first introduces the hypercube-based structural indicators and then presents the corresponding quantum-inspired optimization procedure.

4.1 Hypercube-Based Structural Indicators

The Hamming distance (HMD) $h(x, y)$ measures the bitwise difference between two states $x,\ y \in V_n$, as defined in Eq. (2). In the context of reversible synthesis, the total Hamming distance (THD) is defined to quantify the global discrepancy between the current function F and the identity permutation, as shown in Eq. (3). A lower THD indicates that the state mapping is closer to the target identity mapping.

To evaluate local structural properties, this study utilizes the adjacent Hamming distance (AHD) [10]. For a specific vertex u, the AHD aggregates the Hamming distances of u and its neighbors in the hypercube, providing a measure of local discrepancy as defined in Eq. (4). To further summarize local structural properties at the function level, this study defines the total adjacent Hamming distance (TAHD) as the aggregation of AHD over all vertices, as given in Eq. (5).

Within the hypercube structure, adjacent vertices naturally possess a Hamming distance of one. In the GT gate library, a single adjacent transposition (corresponding to one gate) changes $\mathrm{THD}(F)$ by at most 2 in magnitude, which follows directly from the definition of Hamming distance. Therefore, if $L^*(F)$ denotes the optimal gate count, the theoretical lower bound of the circuit length is defined in Eq. (6).

In the proposed framework, THD serves as a global progress indicator to estimate the initial search budget, while AHD (and its aggregation TAHD) provides fine-grained structural evaluation to guide the search toward promising regions of the solution space.

$$h(x, y) = \sum_{k=1}^{n} (x_k \oplus y_k) \tag{2}$$

$$\mathrm{THD}(F) = \sum_{u \in V_n} h(u, F(u)), \tag{3}$$

$$\mathrm{AHD}_F(u) = h(u, F(u)) + \sum_{j=1}^{n} h(u \oplus e_j, F(u \oplus e_j)), \tag{4}$$

$$\mathrm{TAHD}(F) = \sum_{u \in V_n} \mathrm{AHD}_F(u). \tag{5}$$

$$L^*(F) \geq \left\lceil \frac{\mathrm{THD}(F)}{2} \right\rceil. \tag{6}$$

4.2 Hypercube Structure HMD-GQTS Procedure

This study integrates Hamming distance (HMD) reinforcement guidance into a global-best guided quantum-inspired tabu search (GQTS) framework, referred to as HMD-GQTS, to realize a search-driven reversible circuit synthesis process over the hypercube structure. Accordingly, the synthesis process can be interpreted as a sequence of guided transitions over hypercube vertices, where each step involves selecting a candidate move, evaluating its impact using both global and local criteria, and updating the current solution.

In this framework, THD and AHD (along with its aggregation TAHD) provide complementary guidance at different levels of the search. While THD drives the search toward global convergence, AHD captures local structural characteristics to refine neighborhood exploration and avoid misleading transitions. This combination enables each move to be evaluated not only by its global improvement but also by its local structural consistency. The pseudocode of the proposed HMD-GQTS is presented in Algorithm 1. This section also provides the detailed procedures of the proposed synthesis method.

4.2.1 Problem Formulation and Encoding Method

Fig. 4 summarizes the overall search procedure. A target reversible function is formulated as a labeling problem on a hypercube. The synthesis process is modeled as a sequential decision procedure, in which a series of discrete actions progressively transforms the target permutation into the identity.

Algorithm 1: Overall hypercube encoded HMD-GQTS synthesis

Input: target function F, number of generations G, Q-matrix update factor θ, relaxation threshold ρ.
Output: best circuit synthesis path P_{best}.

1: Compute $\text{THD}(F)$ and $\text{TAHD}(F)$, and set initial step budget $L_0 \leftarrow \lfloor \text{THD}(F)/2 \rfloor$.
2: Initialize the Q matrix of vertex and edge weights using Eqs. (10) and (11).
3: For each generation, sample two candidate trajectories according to Algorithm 2.
4: Evaluate their qualities using Eqs. (12) and (13), and identify the winner, loser, and the global best.
5: If the global best obtained a feasible circuit, directly reinforce its actions through global best locking.
6: Otherwise, update the Q matrix by global best and loser by factor θ using Eqs. (15)–(18).
7: If no feasible circuit is obtained after half of the generations, relax the step budget from L_0 to $L_0 + \rho$.
8: Return the best path found.

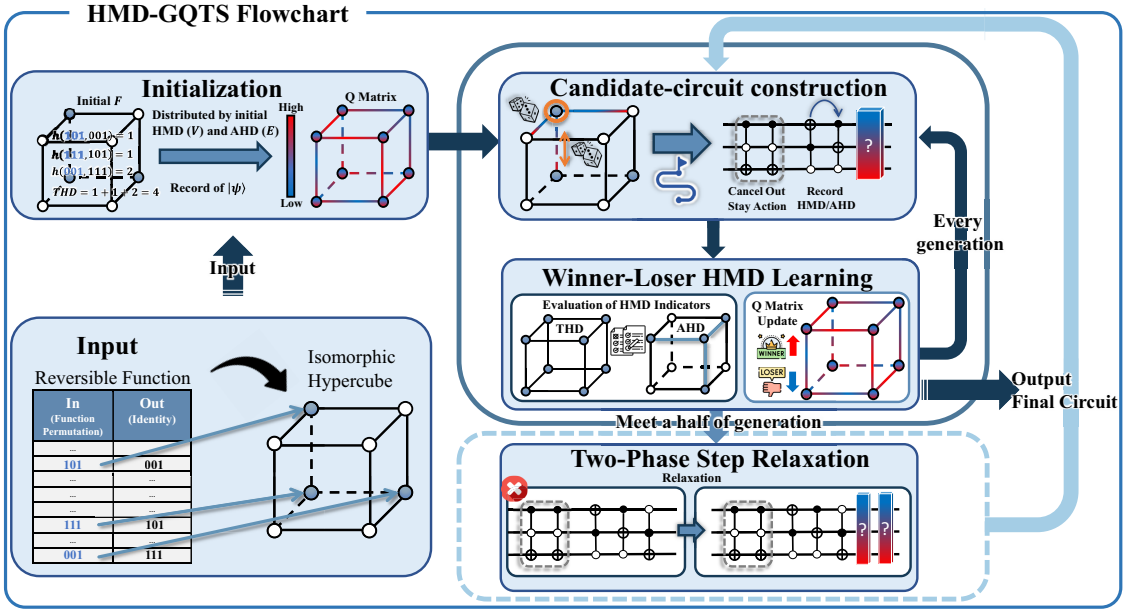

Figure 4: Conceptual architecture of the proposed HMD-GQTS synthesis framework.

At each iteration, the algorithm selects an action by first choosing a vertex and then determining one of its adjacent edges (or a stay action), which together define a reversible swap operation. The resulting candidate solution is evaluated using both global and local criteria. The selection weights of candidate actions are iteratively updated through winner-loser learning and global-best reinforcement, enabling adaptive exploration and exploitation throughout the search.

Formally, the path P including sequence of actions taken during the search is denoted as Eq. (7), where each action a_s is either a stay action (no swap) or an adjacent transposition $\tau_{u,j}$ on the hypercube. Applying these actions to the target permutation produces a sequence of intermediate reversible functions in Eq. (8). The optimization objective is to identify a sequence of actions such that $F^{(L)}$ reaches the identity permutation with as few effective gate operations as possible.

$$P = \left(a_1, a_2, \ldots, a_L\right), \forall a_s \in \{Stay,\ \tau_{u,j}\} \tag{7}$$

$$F^{(0)} = F, \qquad F^{(s+1)} = a_s\left(F^{(s)}\right), \quad s = 1, 2, \ldots, L. \tag{8}$$

To model the action selection process, GQTS maintains a superposition-inspired probability distribution over candidate actions. The superposition state is defined in Eq. (9), where the squared magnitude of the amplitude q_i associated with each state $|i\rangle$ represents the probability of selecting that state.

$$|\psi\rangle = q_0|0\rangle + q_1|1\rangle + \cdots + q_m|m\rangle, \quad \text{s.t.} \quad \sum_{i=0}^{m} ||q_i||^2 = 1 \tag{9}$$

For a circuit with L steps, a sequence of Q matrices is maintained to model the stepwise decision process. At each step s, the Q matrix is decomposed into two components: a vertex-selection model Q_s^V and an edge-selection model Q_s^E. Specifically, Q_s^V models the probability of selecting a vertex (including a stay action), while Q_s^E models the probability of selecting an adjacent edge conditioned on the chosen vertex. This two-level structure models each action as a vertex–edge pair on the hypercube.

4.2.2 Initialization of the Q Matrix

The probability of Q matrices should be normalized from 0 to 1. For easy understanding, the weight is utilized before being reinterpreted into the sampling space. The heuristic from the previous hypercube-based literature [10] is to improve the large HMD and AHD area first. Therefore, the initial vertex and edge weights in Q matrices are defined by Eqs. (10) and (11). The node initialization includes a Stay action. During trajectory construction, the corresponding probabilities are obtained by proportional normalization. Therefore, the first sampling distribution is already informed by local structural discrepancy.

$$Q_s^V(u) = \frac{W_s^V(u)}{\sum_{k=1}^{2^n+1} W_s^V(k)}, \qquad \text{where} \qquad W_s^V(u) = 1 + h(u, F(u)), \qquad W_s^V(\text{Stay}) = 1, \tag{10}$$

$$Q_s^E(u, j) = \frac{W_s^E(u, j)}{\sum_{k=1}^{n} W_s^E(u, k)}, \qquad \text{where} \qquad W_s^E(u, j) = 1 + \text{AHD}_F(u \oplus e_j). \tag{11}$$

4.2.3 Candidate Circuit Construction and Quality Evaluation

The construction of a candidate circuit starts with a sampled step budget L. The pseudocode is listed in Algorithm 1. A trajectory is generated by iteratively sampling actions based on Eqs. (10) and (11), applying the corresponding adjacent swaps, and recording the resulting THD and TAHD traces. If the current permutation reaches the identity (i.e., THD = 0) before the budget is exhausted, the remaining steps are assigned as Stay actions. Furthermore, consecutive identical swaps are cancelled online due to the self-inverse property of adjacent transpositions.

The circuit quality is formulated as a maximization problem. Let $L_0 = \lfloor \text{THD}(F)/2 \rfloor$ denote the practical lower-bound baseline derived from Eq. (6). The quality score Q is then evaluated based on the synthesis outcome:

If a candidate trajectory successfully reaches the identity state, its quality Q_{succ} is evaluated by the proximity of the actual gate count ℓ to the lower bound L_0 in Eq. (12), where ℓ is the number of active moves (excluding Stay actions) and k_1 is a large positive constant to ensure $Q_{\text{succ}} > 0$. This formulation rewards circuits that approach the theoretical minimum length.

$$Q_{\text{succ}} = k_1 - 100(\ell - L_0)^2, \tag{12}$$

If the trajectory fails to reach the identity state within the budget, the quality Q_{fail} is determined by the residual discrepancy and the gate usage in Eq. (13), where k_2, k_3, and k_4 are weighting constants. In this

study, we set $k_2 > k_3 > k_4$ to prioritize global alignment (THD) over local structural connectivity (TAHD) and circuit length.

$$Q_{\text{fail}} = -k_2 \cdot \text{THD}_{\text{final}} - k_3 \cdot \text{TAHD}_{\text{final}} - k_4 \cdot (\ell - L_0)^2. \tag{13}$$

Algorithm 2: Candidate-circuit construction on the hypercube

Input: reversible function F, step budget L, vertex weights W^V, edge weights W^E.
Output: sampled path P and its recorded THD/TAHD traces.
1: Set $F_{\text{curr}} \leftarrow F$.
2: For each step $s = 0, 1, \ldots, L-1$:
  2.1. If $\text{THD}(F_{\text{curr}}) = 0$, record Stay and continue.
  2.2. Sample a vertex or Stay by Eq. (10).
  2.3. If Stay is selected, record Stay.
  2.4. Otherwise, sample a direction by Eq. (11), determine the adjacent vertex, and apply the corresponding swap.
  2.5. Cancel the move if it is identical to the immediately previous move.
  2.6. Record the updated THD and TAHD values of the current permutation.
3: Evaluate the path by Eq. (12) if synthesis succeeds, or by Eq. (13) otherwise.
4: Return the sampled path and its quality.

This design ensures that exact synthesis (reaching THD = 0) has the highest priority. Among unsuccessful paths, the optimizer favors trajectories with smaller residual global mismatch and lower local discrepancy. This approach provides informative ranking signals even for failed attempts, allowing the search engine to distinguish and evolve toward more promising partial synthesis paths.

4.2.4 Winner-Loser HMD Learning and Global-Best Reinforcement

The core philosophy of the proposed quantum-inspired tabu search (QTS) is to navigate the search space by simultaneously approaching optimal solutions and avoiding inferior ones. To implement this, two candidate trajectories are sampled and compared in each generation. Let P^w denote the winner (higher-quality) trajectory and P^ℓ denote the loser (lower-quality) trajectory. Given the complexity of reversible circuit synthesis, a global-best (GB) reinforcement strategy is integrated to accelerate convergence, particularly in the early search phases. Inspired by knowledge-based updating mechanisms [9] to distinguish the improvement performance of best and worst, we propose a Winner-Loser Hamming Distance (HMD) learning strategy. This strategy evaluates the step-by-step performance improvement for each gate in the circuit sequence. Let THD_s^w and TAHD_s^w be the metrics recorded at step s of the winner path, with THD_s^ℓ and TAHD_s^ℓ defined analogously for the loser. The one-step improvement for THD is defined as in Eq. (14), with TAHD improvements (δ_{TAHD}) calculated similarly.

$$\delta_{\text{THD}}^w(s) = \text{THD}_{s-1}^w - \text{THD}_s^w, \qquad \delta_{\text{THD}}^\ell(s) = \text{THD}_{s-1}^\ell - \text{THD}_s^\ell, \tag{14}$$

A step comparison score $S(s)$ is then formulated to quantify the relative advantage of the winner's move over the loser's at each time step s:

$$S(s) = \alpha\big(\delta_{\text{THD}}^w(s) - \delta_{\text{THD}}^\ell(s)\big) + \beta\big(\delta_{\text{TAHD}}^w(s) - \delta_{\text{TAHD}}^\ell(s)\big). \tag{15}$$

In our implementation, we set $\alpha = 1.0$ and $\beta = 0.5$, prioritizing THD as it directly reflects progress toward the target identity. To bound the update magnitude, the score is normalized using a hyperbolic tangent

function in Eq. (16), where $M = 4\alpha + 4(n+1)\beta$ serves as an estimated upper bound for $S(s)$, ensuring $r(s) \in [-1, 1]$.

$$r(s) = \tanh\left(\frac{S(s)}{M}\right), \tag{16}$$

Let A_{gb} and A_{ℓ} represent the actions taken at step s in the global-best and current loser circuits, respectively. The corresponding Q-matrix weights for vertices (W^V) and edges (W^E) are updated as in Eqs. (17) and (18), where θ is the learning rate and $\varepsilon = 10^{-6}$ is a small constant to prevent numerical collapse.

$$W_s^V(A_{gb}) \leftarrow \max\left(\varepsilon, W_s^V(A_{gb})(1 + \theta \cdot r(s))\right), \quad W_s^E(A_{gb}) \leftarrow \max\left(\varepsilon, W_s^E(A_{gb})(1 + \theta \cdot r(s))\right), \tag{17}$$

$$W_s^V(A_{\ell}) \leftarrow \max\left(\varepsilon, W_s^V(A_{\ell})(1 - \theta \cdot r(s))\right), \quad W_s^E(A_{\ell}) \leftarrow \max\left(\varepsilon, W_s^E(A_{\ell})(1 - \theta \cdot r(s))\right), \tag{18}$$

This mechanism rewards actions that yield significant structural improvements while suppressing weaker competing moves. Furthermore, once a feasible circuit (THD = 0) is discovered, the focus shifts from structural guidance to gate-count optimization. We then employ a Lock-to-Global-Best strategy, where the Q-matrix is aggressively updated to favor the global-best circuit structure. During this phase, the learning factor is amplified to $\theta_{gb} = w_{gb} \cdot \theta$ (with $w_{gb} = 10$) to ensure rapid convergence toward the most efficient synthesis path.

4.2.5 Two-Phase Step Relaxation Mechanism

The theoretical lower bound L_0 represents an ideal synthesis scenario; however, it is not always attainable in practice. Taking the exhaustive analysis of 3-bit reversible functions as an example, some target permutations require a gate count greater than this ideal estimation. This discrepancy between the theoretical minimum and the actual reachable gate count is precisely why reversible circuit synthesis is a highly complex combinatorial problem. To address this, this study implements a two-phase step relaxation mechanism.

In the initial phase, the search is strictly constrained by the practical baseline (L_{ini}) to prioritize the discovery of the most compact circuits. If no successful synthesis (i.e., THD = 0) is achieved after half of the prescribed generations, a relaxation trigger is employed. The step budget L is then expanded as in Eq. (19), where $\rho = 4$ in our experiments. This mechanism enables the search engine to overcome structural bottlenecks in the complex search space while still maintaining a preference for compact solutions.

$$L \leftarrow L_{ini} + \rho \tag{19}$$

5 Experimental Results

This study evaluates the proposed HMD-GQTS framework by navigating quantum-inspired optimization with structural guidance to construct effective trajectories for reversible circuit synthesis. The experimental results demonstrate that Hamming distance-based evaluation provides clear guidance, enabling the HMD-GQTS strategy to successfully synthesize all 40,320 exhaustive 3-bit functions and extend effectively to 4-bit benchmark-level experiments. This section details the implementation, parameter configurations, and performance metrics to illustrate the effectiveness and stability of the proposed approach. The comparison with the hypercube rule-based algorithm [10] is conducted to demonstrate the effectiveness.

5.1 Experimental Setup and Parameter Configurations

The performance of HMD-GQTS is validated across 3-bit and 4-bit functions to demonstrate that HMD-related reinforcement enhances both search optimization and stability.

Hyperparameter Tuning: The exhaustive testing of all 40,320 3-bit functions serves as a basis for fine-tuning the hyperparameters, such as the maximum number of generations and the relaxation increment ρ. These global parameters define the search budget and the macro-level strategy for exploring the permutation space. To avoid case-specific over-optimization, the tuning protocol applies one uniform parameter setting to all functions within each bit-level experiment rather than selecting parameters individually for each target function. In the 3-bit experiments, each candidate setting is evaluated over the complete set of 40,320 functions. For the 4-bit experiments, the 20 benchmark-level functions in Table 1 are used to conduct a dimension-level parameter sensitivity analysis, with the goal of identifying a stable operational interval rather than optimizing parameters for individual benchmark cases. All reported experiments use a fixed base random seed of 114.

Table 1: The 20 benchmark-level 4-bit reversible functions used in this study.

Benchmark	Best	Function (Permutation)	Benchmark	Best	Function (Permutation)
4_49	16	15, 0, 10, 4, 3, 11, 1, 7, 8, 5, 6, 2, 12, 9, 14, 13	decode42	11	1, 2, 4, 8, 0, 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15
hwb4	22	0, 8, 1, 12, 2, 5, 9, 14, 4, 6, 10, 7, 3, 11, 13, 15	dmasl	12	0, 1, 14, 3, 4, 5, 7, 8, 15, 13, 10, 6, 9, 12, 11, 2
4b15g_1	19	1, 5, 0, 8, 9, 11, 2, 15, 3, 12, 4, 6, 10, 14, 13, 7	gyang	7	2, 5, 3, 15, 4, 13, 6, 7, 8, 9, 10, 11, 12, 1, 14, 0
4b15g_2	20	1, 9, 0, 4, 10, 8, 2, 11, 3, 15, 5, 12, 7, 14, 13, 6	mini-alu	8	0, 1, 2, 3, 4, 5, 14, 11, 8, 6, 10, 9, 12, 15, 13, 7
4b15g_3	24	3, 1, 7, 13, 11, 0, 8, 15, 2, 5, 10, 6, 9, 14, 12, 4	mod10_171	9	1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 10, 11, 12, 13, 14, 15
4b15g_4	21	3, 1, 11, 7, 8, 0, 9, 5, 2, 6, 15, 13, 14, 4, 10, 12	msaee	15	11, 3, 9, 2, 7, 13, 15, 14, 8, 1, 4, 10, 0, 12, 6, 5
4b15g_5	20	3, 5, 11, 1, 8, 0, 9, 7, 2, 6, 14, 13, 10, 4, 12, 15	oc5	17	6, 0, 12, 15, 7, 1, 5, 2, 4, 10, 13, 3, 11, 8, 14, 9
nth_prime4_inc	14	0, 2, 3, 5, 7, 11, 13, 1, 4, 6, 8, 9, 10, 12, 14, 15	oc6	19	9, 0, 2, 15, 11, 6, 7, 8, 14, 3, 4, 13, 5, 1, 12, 10
aj-e11_81	15	8, 14, 0, 3, 2, 5, 10, 7, 4, 9, 12, 11, 1, 13, 6, 15	oc7	21	6, 15, 9, 5, 13, 12, 3, 7, 2, 10, 1, 11, 0, 14, 4, 8
App2.2	23	7, 14, 9, 6, 11, 0, 13, 2, 5, 15, 10, 12, 1, 4, 3, 8	oc8	15	11, 3, 9, 2, 7, 13, 15, 14, 8, 1, 4, 10, 0, 12, 6, 5

Optimization Parameters: The delicate optimization parameter θ for update Q matrix is fine-tuned separately. Since a one-bit increment significantly increases the complexity of the reversible function space, the update dynamics must be adjusted to maintain convergence. Because this study adopts a winner-loser competitive strategy, two candidate solutions are sampled per generation; thus, the total number of evaluations is calculated as 2 × Generations.

Hardware Environment: The experiments were conducted on a desktop computer equipped with an Intel Core i9-14900K processor, 32 GB RAM, and Windows 11 Pro.

Reward and Punishment Constant Settings The constant settings for reward and penalty are defined as follows: $k_1 = 10^6$, $k_2 = 10^4$, $k_3 = 10^2$, $k_4 = 10$, $\alpha = 1$, $\beta = 0.5$, $w_{gb} = 10$, and $\varepsilon = 10^{-6}$. These values are employed to maintain a clear numerical hierarchy between different circuit qualities.

5.2 Exhaustive 3-Bit Experimental Result

5.2.1 Parameter-Tuning Protocol

In this study, the optimization process is guided by HMD-related indicators to navigate the vast search space efficiently. While existing heuristic and metaheuristic studies on reversible circuit synthesis typically evaluate performance on selected benchmark functions, the present work extends the validation protocol to the entire 3-bit permutation space, covering all 40,320 possible functions. This exhaustive testing ensures that the algorithm's effectiveness is not limited to specific cases but is robust across the complete functional landscape. To identify the optimal configuration for this extensive search, we conducted a comprehensive sweep of the hyperparameters (generations G and relaxation increment ρ) and the optimization parameter (updating factor θ). The search space for this tuning included: Generation budget: $G \in \{50,000, 100,000\}$, Relaxation increment: $\rho \in \{2, 4\}$, and Updating factor: $\theta \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. The initial step budget L_{ini} is set to theoretical minimum L_0.

Each parameter combination was tested via an exhaustive pass over all 40,320 functions. The primary selection criterion was the number of exact-optimal circuits recovered in a single pass. As shown in Fig. 5a, the combination of ρ and G exhibited a similar sensitivity to θ, with 10^{-4} demonstrating the most robust performance.

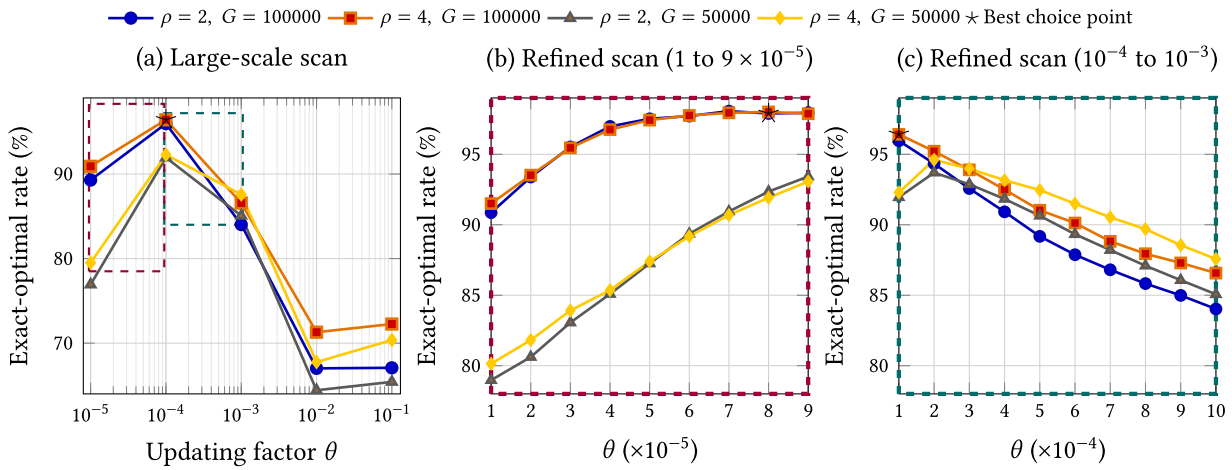

Figure 5: Combined 3-bit parameter-tuning visualization into three panels. (**a**) The left panel shows the large-scale scan to find the θ level tendency. (**b**) The center panel zooms $\theta \in \{1, 2, \ldots, 9\} \times 10^{-5}$, and (**c**) the right panel shows a finer scan over$10^{-4} \leq \theta \leq 10^{-3}$. Star markers denote the best exact-optimal selection point in each panel.

A finer-scale scan of θ was subsequently performed within the ranges of $\{10^{-5}, 10^{-4}\}$ and $\{10^{-4}, 10^{-3}\}$ (comprising 19 ticks in total). Detailed results in Fig. 5b,c indicate that the generation setting significantly influences the optimal θ selection. Specifically, for the delicate range of 1×10^{-5} to 9×10^{-5}, G = 100,000 demonstrates a clear performance advantage over G = 50,000. As shown in Fig. 5c, when the budget is limited to G = 50,000, a slightly higher updating factor $\theta = 2 \times 10^{-4}$ yields better solutions. Ultimately, the relaxation increment ρ = 4 proved to be robust across all experiments. One single set of uniform parameters was applied universally to evaluate all 40,320 distinct functions, preventing the case-specific optimization. Consequently, the optimal configuration was identified as G = 100,000, ρ = 4, and $\theta = 8 \times 10^{-5}$, which achieved 39,516 (98.01%) exact-optimal solutions in a single exhaustive sweep. The remaining 804 functions were successfully synthesized to their exact-optimal gate counts within at most four additional retry rounds (RR) (with $10 \cdot RR$ independent runs executed per remaining failed case in each round, where the number of failed cases progressively decreased).

5.2.2 Single-Sweep Ablation Study

To isolate the contribution of the main design components, we conducted a single-sweep ablation study over the full 40,320-function 3-bit space under the same exhaustive protocol. The full-feature configuration (G = 100,000, $\theta = 8 \times 10^{-5}$, ρ = 4) yields the smallest missed-optimal set, with 804 non-optimal cases out of 40,320. Table 2 shows the ablation validation that removing relaxation, using THD-only guidance, using AHD-only guidance, or disabling global-best locking would all increase the number of missed-optimal cases.

5.2.3 Comparison of Gate-Count Distribution over the Full Permutation Space

A distribution-level comparison over the full 3-bit permutation space provides a more robust measure of algorithm efficacy. Fig. 6 and Table 3 compare the gate-count distributions under the same Generalized

Toffoli (GT) gate library between the prior hypercube rule-based algorithm [10], a hybrid method [32], and the theoretical optimal distribution. Notably, the gate-count distribution produced by the proposed HMD-GQTS perfectly matches the optimal distribution, resulting in completely overlapping curves for "This work" and "Optimal."

Table 2: Single-sweep 3-bit ablation comparison over the full 40,320-function space. Missed-optimal cases denote functions that did not reach the exact-optimal gate count in the first exhaustive sweep.

Setting	Missed-Optimal Cases	Extra Misses vs. Full Features
Full features	804	–
No relaxation	2464	1660
AHD-only guidance	1485	681
THD-only guidance	1405	601
No global-best locking	1349	545

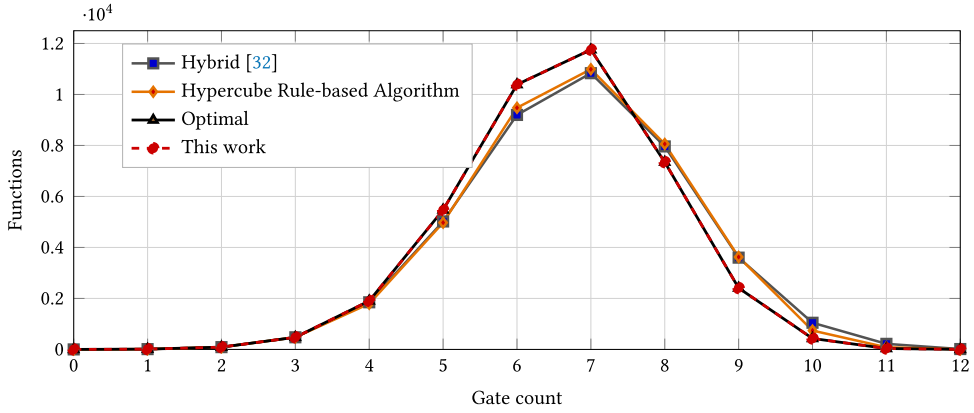

Figure 6: Gate-count distribution over all 40,320 3-bit reversible functions. The present method coincides with the optimal distribution, so the red dashed "This work" curve overlaps the black "Optimal" curve throughout the plot.

Table 3: Gate-count distribution over all 40,320 3-bit reversible functions. The "Hybrid" and " Hypercube Rule-based Algorithm" columns are reproduced from the hypercube-based study [10], while the present method matches the optimal distribution exactly.

Gates	Optimal	Hybrid [32]	Hypercube Rule-Based Algorithm [10]	This Work
0	**1**	1	1	**1**
1	12	12	12	12
2	90	90	90	90
3	476	476	476	476
4	1903	1849	1793	1903
5	5472	5019	4973	5472
6	10,388	9197	9477	10,388
7	11,756	10,834	10,990	11,756
8	7347	7962	8058	7347
9	2408	3600	3625	2408

(Continued)

Table 3 (continued)

Gates	Optimal	Hybrid [32]	Hypercube Rule-Based Algorithm [10]	This Work
10	430	1046	740	430
11	36	219	84	36
12	1	15	1	1
Avg. Gates	6.606349	6.802480	6.767708	6.606349

Compared to the baselines, our method successfully shifts the distribution toward the lower gate-count regions and eliminates the heavy tail previously observed above 9 gates. These results highlight the potential of metaheuristic optimization to match or even exceed the performance of rule-based algorithms through a more flexible and computationally efficient search mechanism.

5.3 Benchmark-Level 4-Bit Experimental Result

The 4-bit experiments complement the exhaustive 3-bit study by evaluating the scalability of HMD-GQTS. Given that the full 4-bit space encompasses $2^4! \approx 2.09 \times 10^{13}$ functions, exhaustive optimality validation is computationally prohibitive. Instead, these experiments serve as evidence that the proposed HMD-GQTS remains practical and effective in producing high-quality circuits for more complex benchmarks. This study utilizes the 20-case 4-bit benchmark set archived in prior studies [10,26]. For each benchmark, the theoretical minimum L_0 is not guaranteed to be a reachable value. To prevent the algorithm from wasting computational effort searching within an infeasible solution space, the initial step budget L_{ini} is set to the current best-known solution. Crucially, adopting this tightest known bound serves as a highly challenging step limit that strictly tests the framework's convergence capability. If the target gate count is not achieved in the first pass, HMD-GQTS is permitted up to four retry attempts per function.

5.3.1 Parameter-Tuning Protocol

The tuning evaluation was conducted with a fixed generation budget $G = 100{,}000$, $\rho = 4$, one relaxation phase, and 100 runs per benchmark case, while scanning the updating factor parameter over $\theta \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$. To verify the parameter sensitivity and robustness, the success rate and the optimal rate are reported across the 20 benchmark cases. The tuning results are summarized in Fig. 7. The initial large-scale exploration identified localized performance peaks around the scales of $\theta = 10^{-1}$ and 10^{-3}. To capture the explicit sensitivity behaviors, three subsequent localized windows were evaluated at a higher resolution: spanning from 0.001 to 0.009 (base scale of 10^{-3}), 0.01 to 0.09 (base scale of 10^{-2}), and 0.1 to 0.9 (base scale of 10^{-1}).

The empirical trajectories demonstrate that both the success rate and optimal rate consistently maintain a remarkably high performance baseline (>85%) across distinct parameters, except at the extreme boundary condition of $\theta = 10^{-4}$, verifying the stable search resilience of the proposed algorithm. Crucially, the final top-performing configurations achieve 100% for both metrics within a continuous range of $\theta = 0.6$ to 0.9, directly confirming the robustness of the framework to parameter variations.

5.3.2 Comparison of Gate-Count Performance

This subsection evaluates the performance of the proposed HMD-GQTS against two representative baselines: an ant-colony optimization (ACO)-based metaheuristic [26] and the hypercube rule-based

algorithm [10]. Table 4 presents the final gate counts, absolute reductions in circuit cost, and corresponding improvement ratios. While ACO-based methods are traditionally recognized as suitable for combinatorial problems with sequence-dependent constraints, our results indicate that hypercube-based frameworks—both the rule-based algorithm and the present study—significantly outperform ACO in reducing gate counts in these 20 benchmark-level functions.

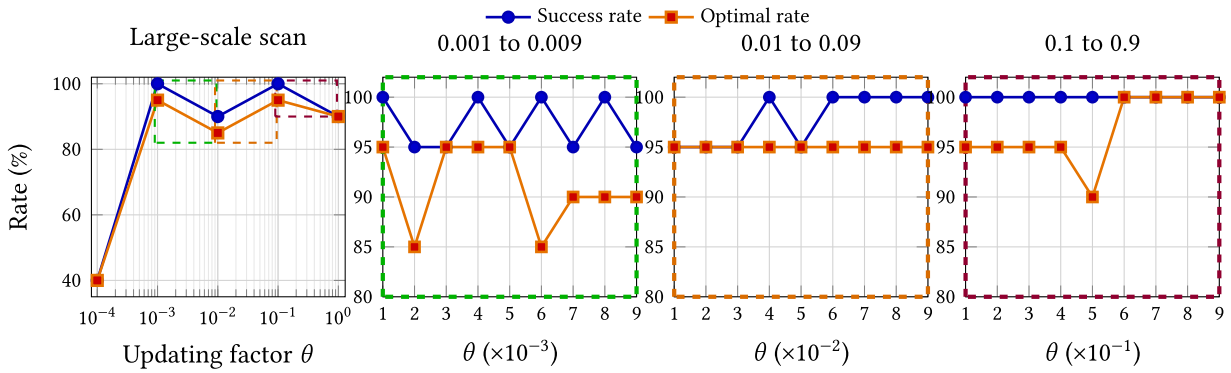

Figure 7: Parameter visualization for the 4-bit tuning. The large-scale panel shows the global trend over $\theta \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$, while the three zoom panels refine the ranges 0.001–0.009, 0.01–0.09, and 0.1–0.9.

Table 4: Representative 4-bit gate-count comparison against the ACO-based metaheuristic [26] method and prior hypercube rule-based baseline [10]. Negative diff means fewer gates than the previous best-known solution.

Benchmark	ACO	Hyp.	Ours	Diff	Imp.	Benchmark	ACO	Hyp.	Ours	Diff	Imp.
4_49	35	16	**14**	−2	12.5%	decode42	25	11	**9***	−2	18.2%
hwb4	20	22	**18**	−4	18.2%	dmas1	24	**12**	**12**	−0	–
4b15g_1	27	19	**17**	−2	10.5%	gyang	30	**7**	**7**	−0	–
4b15g_2	32	**20**	**20**	−0	–	mini-alu	13	8	**6***	−2	25.0%
4b15g_3	25	24	**18**	−6	25.0%	mod10_171	22	**9***	**9***	−0	–
4b15g_4	26	21	**19**	−2	9.5%	msaee	26	**15**	**15**	−0	–
4b15g_5	35	20	**18**	−2	10.0%	oc5	31	**17**	**17**	−0	–
aj-e11_81	25	15	**9***	−6	40.0%	oc6	30	19	**17**	−2	10.5%
nth_prime4_inc	18	**14**	**14**	−0	–	oc7	20	21	**19**	−2	9.5%
App2.2	22	23	**21***	−2	8.7%	oc8	24	**15**	**15**	−0	–

Note: *denotes that the corresponding result achieves the theoretical lower bound L_0.

By integrating the hypercube-based HMD evaluation into a quantum-inspired optimization framework, this study further pushes the boundaries of synthesis efficiency. In summary, the proposed method successfully matched or improved the best-known results for all 20 benchmark functions, achieving an average reduction of 1.70 gates across the entire set. Notably, our approach reduced the gate count for 12 functions, with a maximum reduction of 6 gates, representing a maximum improvement of 40.00% over previous studies. Among these results, five starred values in Table 4 represent cases that reach the theoretical optima, while bold values indicate the best performance among the compared methods. These 4-bit results underscore the scalability and high potential of the HMD-guided strategy for extension to higher-bit reversible functions. The circuits generated by our method are listed in the Appendix A.

6 Conclusion

This study presents the first hypercube-centered reversible synthesis framework that reformulates the rule-based constructive paradigm into a search-based heuristic optimization approach. By embedding

explicit Hamming-distance guidance (THD and AHD) into a quantum-inspired engine, the proposed HMD-GQTS framework ensures that the optimization is driven by domain knowledge rather than blind, black-box exploration. To enhance the search efficiency within the overall structure, the framework incorporates a domain-informed initialization and couples a streamlined two-particle configuration with a global-best mechanism, enabling the algorithm to escape local optima and rapidly converge once a successfully synthesized superior path is locked.

Experimental results rigorously validate this formulation at both exhaustive and benchmark levels. For the complete 3-bit function space, HMD-GQTS achieves over 98% exact-optimal circuits in a single fine-tuned sweep and attains 100% coverage following selective retries. For typical 4-bit benchmark instances, the framework consistently produces competitive gate counts, matching or improving upon prior hypercube rule-based and search-based outcomes.

This superior benchmark performance directly stems from the structural advantages of HMD-GQTS over existing approaches. Compared to rule-based methods that rely on rigid constructive lines, this search-based reformulation operates within a well-defined budget limit while providing flexibility to dynamically adapt paths through the proposed online step relaxation mechanism. Furthermore, compared to search-based methods, the framework delivers explicit hypercube structure-informed HMD guidance that avoids blind wandering. Supported by the efficient Q-matrix updating mechanism and the powerful ability to lock and accelerate toward global optima, the proposed framework successfully integrates domain knowledge with robust search optimization, offering an effective and highly scalable avenue for reversible circuit synthesis.

Acknowledgement: The author expresses sincere gratitude to the previous collaborators whose foundational development and insightful support inspired the conceptual framework and experimental methodology of this research.

Funding Statement: The author received no specific funding for this study.

Availability of Data and Materials: The datasets supporting this article are not currently available due to ongoing analyses. Access requests may be directed to the corresponding author.

Ethics Approval: Not applicable.

Conflicts of Interest: The author declares no conflicts of interest.

Appendix A The Generated Circuit in 4-Bit Benchmark

In this appendix, we present the verified, top-performing gate sequences generated by the proposed framework for the 20 4-bit benchmark instances in Table A1. The bits are ordered from top to bottom as a, b, c, d, where a denotes the most significant bit (MSB). Each gate is represented as `(target, control1, control2, ...)`, with negative controls explicitly designated by an apostrophe (').

Table A1: The generated circuit results optimized by HMD-GQTS within the Generalized Toffoli gate library.

Benchmark	Gate Sequence
4_49	(d,a',b',c') (c,a',b,d') (b,a,c',d) (b,a',c',d) (a,b',c,d') (c,a',b',d) (a,b,c',d) (c,a,b,d) (d,a',b,c') (a,b',c,d) (b,a',c',d) (b,a',c,d') (d,a',b,c') (d,a',b',c)
hwb4	(b,a,c',d) (d,a',b',c) (c,a',b,d') (b,a',c,d') (a,b,c',d') (c,a',b,d') (d,a',b,c) (d,a,b,c') (c,a,b,d') (a,b',c,d) (b,a',c,d) (d,a,b,c') (a,b,c,d') (b,a,c',d') (c,a,b',d) (a,b',c',d) (a,b,c',d') (d,a,b',c')
4b15g_1	(c,a,b,d') (b,a',c',d) (a,b,c,d) (b,a,c,d') (d,a,b',c') (a,b',c,d) (a,b',c',d) (d,a,b,c') (c,a',b',d') (d,a',b',c) (c,a',b',d) (c,a,b',d) (c,a,b,d') (a,b,c',d') (b,a,c',d') (b,a',c,d') (d,a,b',c')
4b15g_2	(c,a,b',d) (a,b,c',d') (a,b,c,d) (d,a',b,c') (c,a,b,d') (b,a,c,d) (b,a,c,d') (d,a,b,c') (d,a,b',c') (c,a,b,d') (a,b,c',d') (d,a',b,c) (b,a',c,d) (b,a,c',d') (c,a',b,d) (d,a',b,c') (a,b',c',d) (d,a',b',c) (c,a',b',d) (d,a',b',c')

(Continued)

Table A1 (continued)

Benchmark	Gate Sequence
4b15g_3	(b,a',c,d') (d,a',b,c') (d,a,b,c') (c,a',b',d') (a,b,c,d) (c,a,b,d') (a,b',c',d') (b,a,c',d) (a,b',c,d) (d,a',b',c) (c,a',b',d') (a,b,c',d) (d,a',b,c) (b,a',c',d') (b,a,c,d) (c,a',b,d') (c,a,b,d) (b,a,c,d)
4b15g_4	(c,a,b',d) (b,a',c',d') (c,a',b,d') (a,b',c',d') (b,a,c,d') (d,a',b,c') (a,b,c',d) (d,a,b,c) (d,a,b',c) (a,b',c,d') (c,a',b,d) (b,a,c',d) (d,a',b,c) (b,a',c,d') (b,a',c,d) (c,a',b',d') (c,a,b,d') (b,a',c',d') (d,a,b',c)
4b15g_5	(c,a,b',d) (c,a,b,d') (b,a,c,d') (b,a',c',d) (b,a',c',d') (c,a',b,d') (d,a',b,c') (a,b',c',d') (d,a',b',c) (a,b',c,d) (c,a',b',d') (d,a',b',c') (b,a',c,d) (a,b,c',d) (d,a',b,c') (d,a',b,c) (b,a',c,d) (b,a,c',d)
nth_prime4_inc	(c,a',b,d') (c,a',b',d) (b,a,c',d') (c,a',b,d) (a,b,c',d') (b,a',c',d) (c,a,b',d) (d,a',b',c) (a,b',c,d) (c,a,b',d') (b,a',c,d) (a,b',c,d) (d,a,b,c') (d,a',b,c)
aj-e11_81	(b,a,c',d') (a,b',c',d') (a,b,c,d') (d,a',b',c') (b,a',c',d') (a,b,c',d') (b,a,c,d') (c,a',b',d') (c,a,b,d')
App2.2	(b,a',c',d) (d,a',b,c') (b,a,c,d) (b,a',c,d) (d,a,b,c) (d,a',b',c') (a,b,c',d') (d,a',b,c) (c,a',b,d) (d,a',b',c) (c,a,b,d') (c,a,b',d) (c,a',b',d) (d,a,b',c') (b,a,c,d) (b,a',c,d) (a,b',c',d) (a,b,c',d) (d,a',b,c') (b,a',c',d) (a,b',c,d)
decode42	(c,a',b',d) (d,a',b',c') (a,b',c',d') (c,a',b',d') (d,a',b,c) (d,a',b',c) (c,a',b,d) (b,a',c,d) (b,a',c',d')
dmasl	(d,a,b,c) (d,a',b,c) (b,a,c,d) (b,a,c',d') (a,b,c,d') (c,a,b,d') (b,a',c,d') (d,a,b,c') (d,a,b,c) (b,a,c',d) (d,a,b',c') (a,b,c,d')
Gyang	(a,b,c',d) (b,a',c',d) (b,a',c,d) (a,b,c,d) (b,a',c,d) (d,a',b',c) (c,a',b',d')
mini-alu	(c,a,b',d) (a,b,c,d) (b,a,c,d) (d,a,b,c) (c,a,b,d) (a,b,c,d')
mod10_171	(c,a',b,d) (c,a',b',d) (b,a',c',d) (d,a,b',c') (d,a',b',c') (a,b',c',d') (d,a',b,c) (d,a',b,c') (d,a',b',c)
msaee	(d,a',b',c) (d,a',b,c) (a,b',c,d') (c,a',b',d') (d,a',b,c') (a,b,c,d) (c,a',b',d) (a,b',c,d') (a,b,c',d') (a,b,c',d') (d,a,b',c) (d,a,b,c') (b,a',c',d') (c,a',b,d) (a,b',c',d) (a,b,c,d')
oc5	(c,a',b',d') (d,a,b,c') (a,b',c',d') (c,a,b,d) (b,a,c',d') (c,a',b,d') (d,a',b,c) (a,b',c,d) (d,a,b',c') (d,a',b,c') (b,a,c,d) (b,a',c',d') (c,a,b',d') (b,a',c,d') (d,a,b',c') (b,a,c',d) (d,a',b,c')
oc6	(a,b,c',d') (c,a,b,d') (c,a,b',d) (c,a',b,d) (d,a',b,c') (c,a,b',d') (d,a',b',c') (b,a,c',d') (d,a',b,c) (d,a,b,c) (a,b',c,d) (b,a,c',d) (b,a,c,d) (a,b',c',d) (b,a,c,d') (a,b,c',d') (b,a,c',d')
oc7	(c,a',b',d') (c,a',b',d) (a,b,c',d') (c,a,b',d') (d,a,b',c') (d,a,b,c') (c,a,b,d') (b,a',c',d') (d,a',b,c') (a,b,c',d') (b,a,c,d) (a,b',c,d) (b,a',c,d') (d,a',b',c) (b,a,c,d) (b,a',c,d) (a,b',c,d) (a,b',c,d') (c,a,b',d')
oc8	(d,a',b',c) (d,a',b,c) (a,b',c,d') (c,a',b',d') (d,a',b,c') (a,b,c,d) (c,a',b',d) (a,b',c,d') (a,b,c',d') (d,a,b',c) (d,a,b,c') (b,a',c',d') (c,a',b,d) (a,b',c',d) (a,b,c,d')

References

1. Nielsen MA, Chuang IL. Quantum computation and quantum information. Cambridge, UK: Cambridge University Press; 2000.
2. Shende VV, Bullock SS, Markov IL. Synthesis of quantum logic circuits. IEEE Trans Comput Aided Des Integr Circ Syst. 2006;25(6):1000–10. doi:10.1109/TCAD.2005.855930.
3. Devi SS, Bhanumathi V. Reversible logic based MOS current mode logic implementation in digital circuits. Comput Mater Contin. 2022;70(2):3609–24. doi:10.32604/cmc.2022.020426.
4. Alharbi M, Edwards G, Stocker R. Novel ultra-energy-efficient reversible designs of sequential logic quantum-dot cellular automata flip-flop circuits. J Supercomput. 2023;79:11530–57. doi:10.1007/s11227-023-05134-1.
5. Chou YH, Hua CY, Tseng RW, Kuo SY. Quantum-inspired optimization algorithm for 3D multi-objective base-station deployment in next-generation 5G/6G wireless network. Comput Mater Contin. 2026;87(2):1. doi:10.32604/cmc.2025.075705.
6. Xu G, Wang L, Huang Y, Lu Y, Liu X, Tan W, et al. A quantum-inspired algorithm for clustering and intrusion detection. Comput Mater Contin. 2026;87(1):48. doi:10.32604/cmc.2025.074256.
7. Chiang HP, Chou YH, Chiu CH, Kuo SY, Huang YM. A quantum-inspired Tabu search algorithm for solving combinatorial optimization problems. Soft Comput. 2014;18(9):1771–81. doi:10.1007/s00500-013-1203-7.
8. Wang WH, Chiu CH, Kuo SY, Huang SF, Chou YH. Quantum-inspired tabu search algorithm for reversible logic circuit synthesis. In: Proc IEEE Int Conf Syst Man Cybern (SMC); 2012 Oct 14–17; Seoul, Republic of Korea. p. 709–14. doi:10.1109/ICSMC.2012.6377742.

9. Kuo SY, Hua CY, Hsu ET, Chen HP, Shen JY, Liu CL, et al. Controlled-swap-based and knowledge navigated quantum-inspired computational intelligence for quantum circuit optimization. In: Proc IEEE Int Conf Fuzzy Syst (FUZZ-IEEE); 2024 Jun 6–Jul 5; Yokohama, Japan. p. 1–9. doi:10.1109/FUZZ-IEEE60900.2024.10612170.
10. Jiang YC, Tseng KC, Hua CY, Kuo SY, Chou YH, Kuo SY. A novel hypercube-based heuristic for quantum boolean circuit synthesis. IEEE J Emerg Sel Top Circ Syst. 2022;12(3):648–61. doi:10.1109/JETCAS.2022.3202840.
11. Yang M, Yue F, Wang W, Meng X, Wang L, Han P, et al. LIRB-based quantum circuit fidelity assessment and gate fault diagnosis. Comput Mater Contin. 2025;82(2):2215–33. doi:10.32604/cmc.2024.058163.
12. Han Z, Li H, Lu K, Liu S, Ju M. Research on optimization of hierarchical quantum circuit scheduling strategy. Comput Mater Contin. 2025;82(3):5097–113. doi:10.32604/cmc.2025.059577.
13. Li Z, Zhang W, Zhang G, Dai J, Hu J, Perkowski M, et al. An extended approach for generating unitary matrices for quantum circuits. Comput Mater Contin. 2020;62(3):1413–21. doi:10.32604/cmc.2020.07483.
14. Landauer R. Irreversibility and heat generation in the computing process. IBM J Res Dev. 1961;5(3):183–91. doi:10.1147/rd.53.0183.
15. Bennett CH. Logical reversibility of computation. IBM J Res Dev. 1973;17(6):525–32. doi:10.1147/rd.176.0525.
16. Shende VV, Prasad AK, Hayes JP. Synthesis of reversible logic circuits. IEEE Trans Comput Aided Des Integr Circ Syst. 2003;22(6):710–22. doi:10.1109/TCAD.2003.811448.
17. Miller DM, Maslov D, Dueck GW. A transformation based algorithm for reversible logic synthesis. In: Proc 40th Des Autom Conf (DAC); 2003 Jun 2–6; Anaheim, CA, USA. p. 318–23. doi:10.1145/775832.775915.
18. Maslov D, Dueck GW, Miller DM. Techniques for the synthesis of reversible Toffoli networks. ACM Trans Des Autom Electron Syst. 2007;12(4):42. doi:10.1145/1278349.1278355.
19. Saeedi M, Markov IL. Synthesis and optimization of reversible circuits—a survey. ACM Comput Surv. 2013;45(2):21. doi:10.1145/2431211.2431220.
20. Jung J, Choi IC. A multi-commodity network model for optimal quantum reversible circuit synthesis. PLoS One. 2021;16(6):e0253140. doi:10.1371/journal.pone.0253140.
21. Chen YC, Chao FJ. Optimization of reversible logic networks with gate sharing. In: Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC); 2023 Jul 16–19; Tokyo, Japan. p. 166–71.
22. Wu X, Li L. Asymptotically optimal synthesis of reversible circuits. Inf Comput. 2024;305:105235. doi:10.1016/j.ic.2024.105235.
23. Sasamal TN, Singh AK, Mohan A. Reversible logic circuit synthesis and optimization using adaptive genetic algorithm. Procedia Comput Sci. 2015;70:407–13. doi:10.1016/j.procs.2015.10.054.
24. Abdalhaq B, Awad A, Hawash A. A fast Binary Decision Diagram (BDD)-based reversible logic optimization engine driven by recent meta-heuristic reordering algorithms. Microelectron Reliab. 2021;122:114168. doi:10.1016/j.microrel.2021.114168.
25. Palahuta M, Deibuk V. Improved synthesis of reversible circuits based on the ant colony optimization algorithm. Inf Technol Comput Model. 2023:202–4. doi:10.31861/ITCM.2023.57.
26. Podlaski K. Ant colony optimization implementation for reversible synthesis in Walsh-Hadamard domain. In: Comput Sci (ICCS). Lect Notes Comput Sci. 2020;12141:230–43. doi:10.1007/978-3-030-50426-7_18.
27. Kuo SY, Jiang YC, Chou YH, Kuo SY, Kung SY. Quantum computer-aided design automation: optimization algorithms and visualization interfaces. IEEE Nanotechnol Mag. 2023;17(2):15–25. doi:10.1109/MNANO.2023.3249500.
28. Oneto L, Navarin N, Micheli A, Pasa L, Gallicchio C, Bacciu D, et al. Informed machine learning for complex data. Neurocomputing. 2026;669:132505. doi:10.1016/j.neucom.2025.132505.
29. Li Y, Zhan RH, Rao J, Liu M, Sang P, Zeng X, et al. Structure-informed machine learning for drug discovery: a task-centric perspective. Brief Bioinform. 2026;27(1):bbag081. doi:10.1093/bib/bbag081.
30. Toffoli T. Reversible computing. In: Automata Lang Program (ICALP). Lect Notes Comput Sci. 1980;85:632–44. doi:10.1007/3-540-10003-2_104.
31. Zeng GJ, Chiang HH, Kuo SY, Chou YH. A novel classifying algorithm for reversible circuit synthesis. In: Proc IEEE Int Conf Syst Man Cybern (SMC); 2015 Oct 9–12; Hong Kong, China. p. 62–7. doi:10.1109/SMC.2015.24.
32. Zhu W, Li Z, Zhang G, Pan S, Zhang W. A reversible logical circuit synthesis algorithm based on decomposition of cycle representations of permutations. Int J Theor Phys. 2018;57(8):2466–74. doi:10.1007/s10773-018-3768-5.