

# The impact of optimization approximation algorithms on the performance of the BHT-QAOA

Ali Al-Bayaty<sup>1,\*</sup>, Marek Perkowski<sup>1</sup>

Academic Editors: Misha Erementchouk, Randy Kuang

## Abstract

This article investigates the performance impact of five classical optimization approximation algorithms on our previously introduced quantum search algorithm, termed the Boolean–Hamiltonians Transform for Quantum Approximate Optimization Algorithm (BHT-QAOA), to effectively search for all best-approximated solutions for Boolean-based problems. These optimization approximation algorithms are BFGS, L-BFGS-B, SLSQP, COBYLA, and COBYQA. Their performance impact is evaluated and compared using two proposed performance metrics—(i) the final number of function evaluations (the lower numbers denote the best optimization approximation algorithms) and (ii) the final quality of qubit measurements (the higher values indicate all best-approximated solutions were found for a problem). Arbitrary classical Boolean problems in various logical structures were examined and evaluated using the BHT-QAOA, these five optimization approximation algorithms, and a simulated noisy model of an IBM quantum computer. Broadly, the BHT-QAOA, with these five algorithms, successfully finds all optimized approximated solutions for these problems. Specifically, both the BFGS and SLSQP algorithms successfully find all best-approximated solutions for these problems, in the context of fewer function evaluations and higher quality of qubit measurements.

**Keywords:** Boolean oracles, Hamiltonians, classical optimization, approximation algorithms, quantum search algorithms, QAOA

**Citation:** Al-Bayaty A, Perkowski M. The impact of optimization approximation algorithms on the performance of the BHT-QAOA. *Academia Quantum* 2025;2. <https://doi.org/10.20935/AcadQuant7996>

## 1. Introduction

Combinatorial optimization problems are important for several practical applications, such as MaxCut [1, 2], Graph Coloring [3, 4], Knapsack [3, 5], Ashenhurst–Curtis decomposition [6], and the Traveling Salesman Problem (TSP) [7, 8]. Many combinatorial optimization problems can be solved using the quantum approximate optimization algorithm (QAOA), as introduced by Farhi et al. [9, 10]. A detailed didactic presentation and explanation of the QAOA can be found in [11–15]. In general, many authors solve various Boolean-based constraint satisfaction and optimization problems using Grover’s algorithm [16–19] and not the QAOA. In contrast, our previous work [20] introduced a new generalized methodology of solving such problems using the QAOA, involving converting the Boolean oracle for a problem into a Hamiltonian-based oracle of QAOA. Our methodology was termed the “Boolean–Hamiltonians Transform for QAOA (BHT-QAOA)”. For brevity, abbreviated details for the methodology and architecture of the BHT-QAOA are discussed in Section 2 of this article. Because of the generality of the BHT-QAOA and the many possibilities it gives to extend our research, some of them are discussed here, and we believe that our approach can become a valid extension of Grover’s algorithm to solve a broad category of satisfaction and optimization problems.

In the quantum domain, the QAOA represents a combinatorial optimization problem in the form of an ansatz Hamiltonian oracle, which is the so-called “Hamiltonian clauses ( $H_C$ )”, and an ansatz Hamiltonian operator, which is the so-called “Hamiltonian mixer ( $H_M$ )”. The “ansatz” means that  $H_C$  and  $H_M$  are composed of

parameterized rotational quantum gates, such as the rotational Pauli-Z (RZ) and the rotational Pauli-X (RX), respectively [9, 10]. The  $H_C$  is a unitary operator ( $e^{-i\gamma H_C}$ ) that comprises the gates RZ( $v \cdot \gamma$ ), RZZ( $v \cdot \gamma$ ), RZZZ( $v \cdot \gamma$ ), and so on, while the  $H_M$  is a unitary operator ( $e^{-i\beta H_M}$ ) that comprises  $n$  numbers of RX( $\omega \cdot \beta$ ) gates, where  $v$  and  $\omega$  are the coefficients of time evolutions;  $\gamma$  and  $\beta$  are the parameterized angular rotations between the angles of  $[0, 2\pi]$  and  $[0, \pi]$ , respectively [9, 10]; and  $n$  is the number of input qubits for a problem initially set to the states of  $|0\rangle$ .

Note that  $H_M$  acts as the diffusion operator of the QAOA, analogous to the diffusion operator in Grover’s search algorithm [16–19], and  $H_M$  may include other variants and types of gates, not just RX gates, depending on the model of QAOA used [21–25]. To improve the quality of all approximated solutions,  $H_C$  and  $H_M$  are iterated for a number of repetitions ( $p$ ) for  $p \geq 1$ , such that every  $e^{-i\gamma_p H_C}$  consists of RZ( $v \cdot \gamma_p$ ), RZZ( $v \cdot \gamma_p$ ), etc., and every  $e^{-i\beta_p H_M}$  consists of RX( $\omega \cdot \beta_p$ ).

In the classical domain, the numerical values of coefficients ( $v$  and  $\omega$ ) are calculated during the construction of  $H_C$  and  $H_M$ , while the numerical values of angles ( $\gamma$  and  $\beta$ ) are initially randomized as  $[\gamma_1 \dots \gamma_p, \beta_1 \dots \beta_p]$  for  $H_C$  and  $H_M$ , respectively. Note that some studies initialized such angles to defined fixed initial values using machine learning and tensor techniques [21–25]. In general, the circuit of QAOA is executed with a quantum processing unit (QPU), and then classically measured for approximated solutions depending on the chosen values of  $\gamma$  and  $\beta$ . The measured solutions (as

<sup>1</sup>Department of Electrical and Computer Engineering, Portland State University, Portland, OR, USA.

\*email: albayaty@pdx.edu

the energy cost of the QAOA [9, 10]), the chosen values of  $\gamma$  and  $\beta$  (as the optimization parameters of the QAOA), and the Hamiltonians ( $H_C$  and  $H_M$  as objective functions) are fed to a classical optimization minimizer [26–28]. Note that a classical optimization minimizer may contain a set of optimization approximation algorithms. Such a minimizer then recalculates the numerical values of these optimization parameters based on the energy cost from the objective function and updates the  $H_C$  and  $H_M$  of the QAOA with new optimized numerical values of  $\gamma$  and  $\beta$ , respectively, for a number of function evaluations ( $n_{fev}$ ). Notably, in Grover’s algorithm, the freedom of a user is limited only to the design of an oracle. In contrast, such a freedom is much wider using the BHT-QAOA, as it is related to the choice of rotational values and the construction or adaptation of relevant optimization minimizers.

On the one hand, in our work [20], the BHT-QAOA successfully solves arbitrary classical Boolean problems as Hamiltonians ( $H_C$  and  $H_M$ ) of the QAOA with promising correct solutions. Such arbitrary classical Boolean problems were designed as Boolean oracles [19, 29] in various logical structures, e.g., Product-Of-Sums (POS) [30, 31], Sum-Of-Products (SOP) [30, 32], Exclusive-or Sum-Of-Products (ESOP) [33, 34], XOR-Satisfiability (CNF-XOR SAT and DNF-XOR SAT) [35, 36], Algebraic Normal Form (ANF) (or Reed-Muller expansion) [37, 38], just to name a few. In [20], the SciPy scientific package [39] is utilized as a classical optimization minimizer using the constrained optimization by the linear approximation (COBYLA) algorithm [26, 28, 39].

On the other hand, to enhance the solution correctness of the BHT-QAOA, this article investigates various optimization approximation algorithms of SciPy to find all optimized approximated solutions for arbitrary classical Boolean problems. These optimization approximation algorithms are as follows: (i) Broyden–Fletcher–Goldfarb–Shanno (BFGS) [39–41], (ii) limited-memory BFGS with bounds (L-BFGS-B) [42, 43], (iii) sequential least squares programming (SLSQP) [44, 45], (iv) COBYLA, and (v) constrained optimization by quadratic approximations (COBYQA) [46]. These algorithms are discussed in Section 2, along with the reasons why we chose to use them in this research. Our investigation compares and classifies these five algorithms based on our two proposed performance metrics: (i) the final utilization of  $n_{fev}$  and (ii) the final quality of qubit measurements.

Experimentally, arbitrary classical Boolean applications are expressed as Boolean oracles in various logical structures and then solved using the BHT-QAOA in  $p$  repetitions, the above-described five optimization approximation algorithms, and a simulated noisy model of an IBM QPU. We found that both the BFGS and SLSQP algorithms successfully find all best-approximated solutions for these applications, in the context of fewer utilizations of  $n_{fev}$  and higher quality of qubit measurements. In this article, the “best-approximated” means that the measured solutions for a problem have the highest probabilities, as compared to the same measured solutions with lower probabilities when utilizing different optimization approximation algorithms.

## 2. Materials and methods

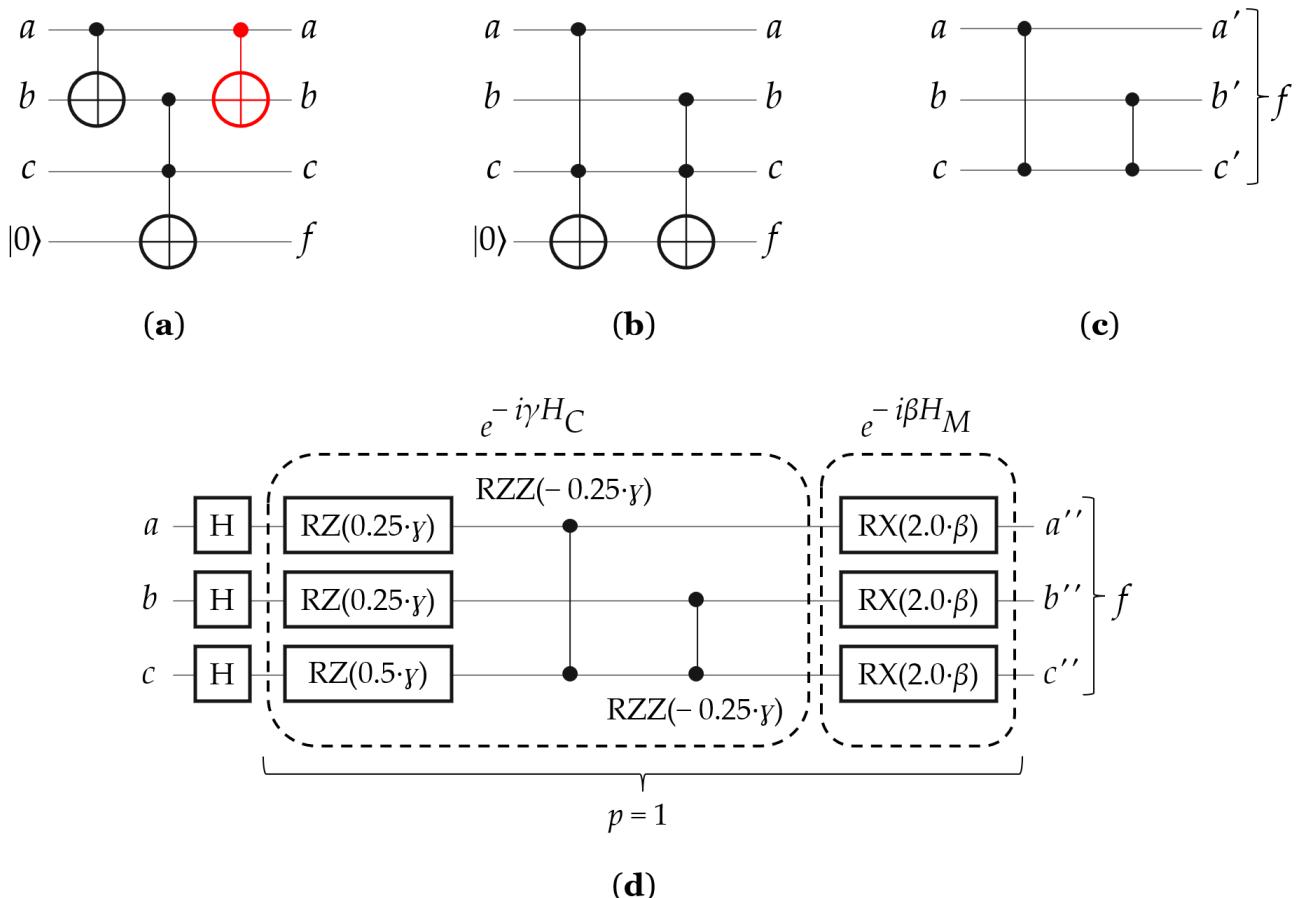
In quantum computing, a Boolean oracle is a straightforward approach for conceptually expressing an arbitrary classical Boolean problem compared to using a Phase oracle, because (i) the quantum Boolean-based gates can be easily realized using the truth tables (and De Morgan’s Laws [30]) of their equivalent classical Boolean gates, and (ii) the quantum Boolean-based gates and their qubits can be easily analyzed using classical Boolean logic, for example, the Boolean logic of ‘o’ represents a quantum state of  $|0\rangle$ , and ‘1’ represents a quantum state of  $|1\rangle$ .

In the BHT-QAOA, converting a Boolean oracle (in any logical structure) into a Phase oracle will (i) remove all ancilla qubits (including the output qubit)—i.e., the total number of utilized qubits will be dramatically reduced to the number of input qubits only—and (ii) omit the mirror (uncomputing) sub-circuits of a Boolean oracle, i.e., the total number of quantum gates will be significantly minimized for the circuit of a Phase oracle, depending on the initial construction of a Boolean oracle that expresses a classical Boolean problem. For brevity and completeness, the following two subsections discuss the abbreviated methodology and architecture of the BHT-QAOA; a complete explanation of BHT-QAOA can be found in [20].

### 2.1. The abbreviated methodology of BHT-QAOA

The essential methodology of the BHT-QAOA for solving arbitrary classical Boolean problems can be simply presented as follows. Firstly, a classical Boolean problem is conceptually designed as a Boolean oracle in an arbitrary logical structure, e.g., POS, SOP, or ESOP. Such an oracle may consist of  $n$  input qubits (as the literals of a Boolean expression or the variables of a problem),  $m$  ancilla qubits (as the auxiliary output qubits) for intermediate quantum operations, and one *fqubit* (as the functional qubit) for the final quantum output of this oracle, where  $n \geq 2$  and  $m \geq 0$ . For instance, a classical Boolean problem “ $f = (a \oplus b) \wedge c$ ” is given in CNF-XOR SAT structure [35, 36], and **Figure 1a** presents its conceptual design as a Boolean oracle, where  $a$ ,  $b$ , and  $c$  are the input qubits (as the literals of a problem), and  $f$  is the *fqubit* (as the final output for a problem).

Secondly, this Boolean oracle in an arbitrary logical structure is converted into its equivalent Boolean oracle in the ESOP structure. There are many synthesis methods to achieve such a conversion, such as ESOP synthesis [33], Karnaugh map synthesis [30], binary decision diagram (BDD) synthesis [47, 48], etc. By utilizing any synthesis method, all mirror gates and  $m$  ancillae (except *fqubit*) are removed from the converted Boolean oracle in the ESOP structure. Note that a synthesis method may not generate the minimized ESOP structure; however, the DSOP (Disjoint Sum-Of-Products) [20, 49] structure may be generated, which is an expensive structure as compared to the minimized ESOP structure, depending on the final number of  $n$ -bit Toffoli gates, where  $n \geq 3$  qubits. For instance, a Boolean oracle “ $f = (a \oplus b) \wedge c$ ” in the CNF-XOR SAT structure can be converted into the ESOP structure using the flattening technique (as the distributivity property of switching-algebra theorems [30]), such that  $f = (a \oplus b) \wedge c = (a \wedge c) \oplus (b \wedge c)$ , and **Figure 1b** shows the quantum circuit of this Boolean oracle in ESOP structure. In **Figure 1b**, notice the removal of mirror gates.



**Figure 1** • The quantum circuits for a classical Boolean problem “ $f = (a \oplus b) \wedge c$ ”: (a) the Boolean oracle in CNF-XOR SAT structure, (b) the Boolean oracle “ $f = (a \wedge c) \oplus (b \wedge c)$ ” in ESOP structure, (c) the Phase oracle, and (d) the Hamiltonians ( $H_C$  and  $H_M$ ) of BHT-QAOA, where (i)  $a, b$ , and  $c$  are the input qubits (as the literals of a problem); (ii)  $a', b'$ , and  $c'$  are the phase-modified output qubits; (iii)  $a'', b'',$  and  $c''$  are the amplitude-modified output qubits. (iv)  $f$  is the final output for a problem, (v)  $p$  is the total number of the Hamiltonians ( $H_C$  and  $H_M$ ) repetitions without Hadamard ( $H$ ) gates, and (vi) the quantum gates in red denote the mirror sub-circuits.

Thirdly, this Boolean oracle in the ESOP (or DSOP) structure is transformed into its equivalent Phase oracle by utilizing the technique originally discussed by Figgatt et al. [29] for transforming 4-bit Toffoli gates into 3-bit controlled-Z (CCZ) gates for Grover’s algorithm of single-solution [16–19]. We efficiently generalized their technique to include Feynman (CX) and  $n$ -bit Toffoli ( $C^{n-1}X$ ) gates as well, and we termed our generalized technique the “generalized transformation rules”, encompassing the following:

**Rule 1:** A 2-bit Feynman (CX) gate is transformed into a Pauli-Z (Z) gate, when a solution is satisfiable for the formula of  $q_j \oplus \text{fqubit} = -(-1)^{q_j}$ , where  $j$  is the index of an input qubit ( $q$ ).

**Rule 2:** A 3-bit Toffoli (CCX) gate is transformed into a controlled-Z (CZ) gate, when a solution is satisfiable for the formula of  $(q_j \wedge q_k) \oplus \text{fqubit} = -(-1)^{q_j \cdot q_k}$ , where  $j$  and  $k$  are the indices of input qubits ( $q$ ).

**Rule 3:** An  $n$ -bit Toffoli ( $C^{n-1}X$ ) gate is transformed into an  $(n-1)$ -bit multi-controlled Z (MCZ) gate, when a solution is satisfiable for the formula of  $(\bigwedge_{j=1}^{n-1} q_j) \oplus \text{fqubit} = -(-1)^{\prod_{j=1}^{n-1} q_j}$ , where  $j$  is the index of an input qubit ( $q$ ) and  $n \geq 3$  qubits.

After applying these generalized transformation rules on the Boolean oracle in ESOP (or DSOP) structure, the resultant circuit of the Phase oracle is simply constructed, with the removal of *fqubit*, i.e., the width of a circuit is reduced, and there is significant

minimization of multi-controlled quantum gates, meaning the depth of a circuit is shrunk. For instance, a Boolean oracle “ $f = (a \wedge c) \oplus (b \wedge c)$ ” in the ESOP structure can be transformed into a Phase oracle using the aforementioned generalized transformation rules, and **Figure 1c** demonstrates the quantum circuit of this Phase oracle. In **Figure 1c**, notice the removal of *fqubit* and the minimization of two 3-bit Toffoli gates into two 2-bit CZ gates.

Fourthly, the Hamiltonians ( $H_C$  and  $H_M$ ) of the BHT-QAOA are directly generated from the transformed Phase oracle using our four proposed “generalized composition rules ( $H_g$ )” stated in **Table 1**. These four rules are generalized from three of Hadfield’s Boolean-based composition rules ( $H_f$ ) [50] for the quantum Boolean-based gates of CX, CCX, and  $C^{n-1}X$ . In **Table 1**, four quantum gates are mainly utilized to generate the Hamiltonians ( $H_C$  and  $H_M$ ) from a Phase oracle, using the quantum phase-based gates of Z, CZ, MCZ, and X.

Finally, based on **Table 1**, the four generalized composition rules ( $H_g$ ) will then be directly applied to a Phase oracle to generate  $H_C$  and calculate its  $v$  coefficient, as presented in Section 1. Because  $\beta$  (as a set of rotational angles of  $H_M$ ) rotates between  $[0, \pi]$  [9, 10], we set its coefficient ( $\omega$ ) to cover all the range  $[0, 2\pi]$  for possible phase values of RX gates in  $H_M$ , to find all optimized approximated solutions for an arbitrary classical Boolean problem. Such that,  $\omega$  is initially set to ‘2.0’ for all  $n$  numbers of  $RX(\omega \cdot \beta)$  gates in  $H_M$ ,

where  $n$  is the total number of input qubits. For instance, **Figure 1d** illustrates the Hamiltonians ( $H_C$  and  $H_M$ ) of BHT-QAOA after applying the generalized composition rules ( $H_g$ ) to

the Phase oracle shown in **Figure 1c**, where  $H_C = -\frac{1}{4}I + \frac{1}{4}Z_a + \frac{1}{4}Z_b + \frac{1}{2}Z_c - \frac{1}{4}Z_aZ_b - \frac{1}{4}Z_bZ_c$  and  $H_M$  consists of three RX(2.0· $\beta$ ) gates.

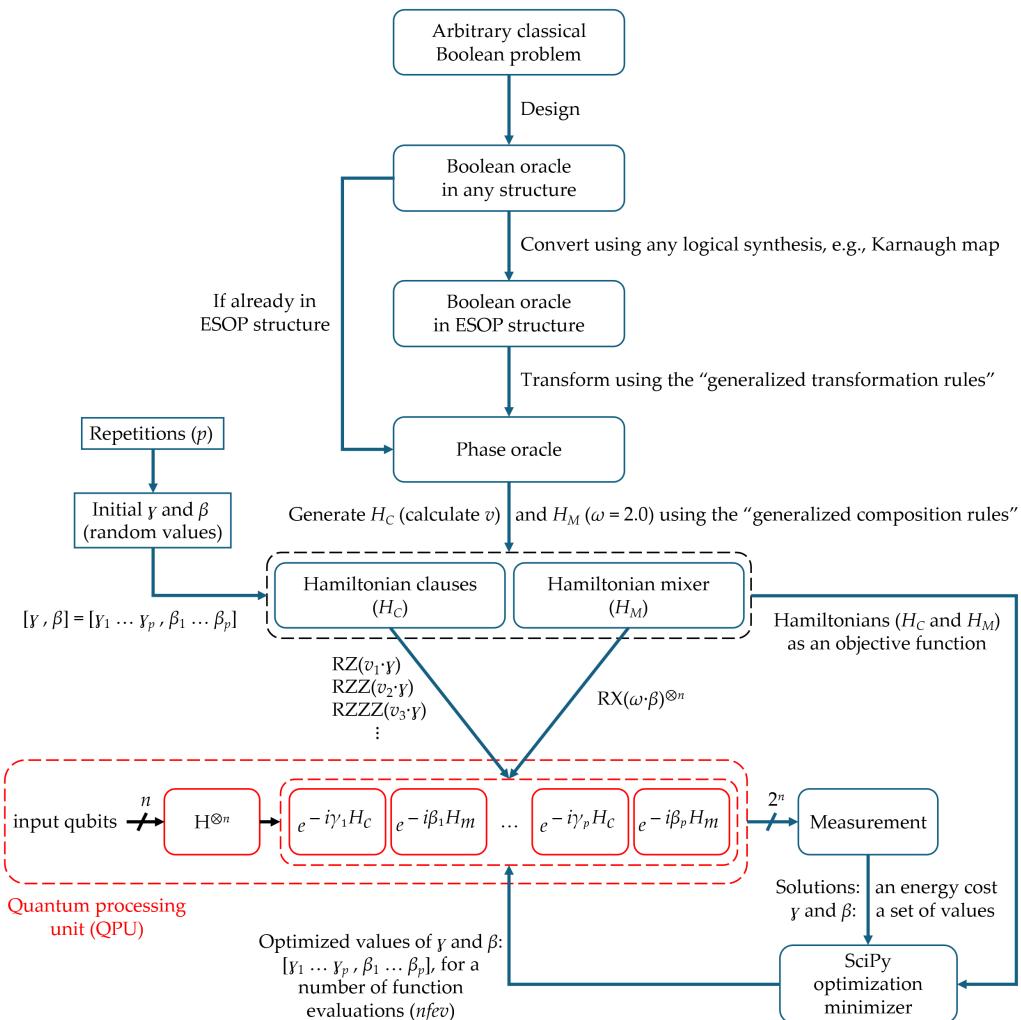
**Table 1** • Our generalized composition rules ( $H_g$ ) for Phase oracles, where  $j$  and  $k$  are the indices of input qubits ( $q$ ),  $Z_j$  is the RZ gate applied on  $q_j$ ,  $Q = \{q_j, q_k \dots q_l q_k \dots\}$ , and  $Z_Q = \{Z_j, Z_k \dots Z_l Z_k \dots\}$  [20].

Rules	Gate	$g(x)$	$H_g$
Rule 4	Z	$(-1)^{q_j}$	$-\frac{1}{2}I + \frac{1}{2}Z_j$
Rule 5	CZ	$(-1)^{q_j \cdot q_k}$	$-\frac{1}{4}I + \frac{1}{4}(Z_j + Z_k - Z_jZ_k)$
Rule 6	MCZ	$(-1)^{\prod_{j=1}^n q_j}$	$\frac{1}{2^n} \prod_{j=1}^n (-1)^{j+1} (I - Z_j)$
Rule 7	X	$(-1)^{\forall j \in Q}$	Invert signs ( $\pm$ ) of all $j$ th qubits in $Z_Q$

## 2.2. The abbreviated architecture of BHT-QAOA

After transforming an arbitrary classical Boolean problem to the Hamiltonians ( $H_C$  and  $H_M$ ) and calculating their coefficients ( $v$  and  $\omega$ ), respectively, the numerical values of  $\gamma$  and  $\beta$  are initially randomized and then plugged into the architecture of the BHT-QAOA for the first execution, as illustrated in **Figure 2**.

Subsequently, the SciPy minimizer optimizes these numerical values ( $\gamma$  and  $\beta$ ) to find all optimized approximated solutions for  $nfev$  using (i) both  $H_C$  and  $H_M$  (in a number of  $p$ ), as the “objective function”, which needs to be minimized; (ii) previously measured solutions, as the “energy cost” of the objective function; and (iii) the previously calculated  $\gamma$  and  $\beta$ , as their “numerical values” need to be optimized.



**Figure 2** • The architecture of our Boolean–Hamiltonians Transform for QAOA (BHT-QAOA) to solve arbitrary classical Boolean problems with Hamiltonians ( $H_C$  and  $H_M$ ). The architecture of BHT-QAOA is mainly grouped into two processing domains: (i) the classical processing domain (denoted in blue) and (ii) the quantum processing domain (denoted in red) [20].

### 2.3. The five optimization approximation algorithms

In our research on the BHT-QAOA [20], we utilized the SciPy minimizer and COBYLA algorithm, which resulted in promising optimized approximated solutions for arbitrary classical Boolean problems. In the present article, we investigate various optimization approximation algorithms for the BHT-QAOA. These algorithms are BFGS, L-BFGS-B, SLSQP, COBYLA, and COBYQA. We compare the outcomes of these five algorithms using our two proposed performance metrics:

- (1) The final utilization of  $nfev$ : the total number of function evaluations (calls) between a QPU and a classical optimization minimizer for a specific approximation algorithm.
- (2) The final quality of qubit measurements: the percentage of the highest probabilities for measured solutions ( $S$ ) over all probabilities ( $P$ ) of  $S$  and non-solutions (with lower probabilities) for a specific approximation algorithm, as expressed in **Equation (1)**, where  $N = 2^n$ ,  $n$  is the total number of input qubits, and  $P$  is always equal to ‘1.0’.

$$\begin{aligned} \text{Quality of qubit measurement}_{\text{algorithm}} &= \frac{\sum_{i=1}^N S_i}{\sum_{j=1}^N P_j} \times 100 \\ &= \sum_{i=1}^N S_i \times 100 \end{aligned} \quad (1)$$

Note that, after measurement, qubits are converted into bits, and the quality of qubit measurements indicates that the solutions become more distinguishable than the non-solutions.

In our research, the reasons that we chose these five algorithms are because they (i) do not require excessive computations in finding the Hessian matrix and gradients [45, 46]; (ii) require fewer approximation iterations, i.e.,  $nfev$ ; and (iii) can handle unconstrained and linearly and nonlinearly constrained problems.

The Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm [39–41] is an iterative quasi-Newton method [45] for solving unconstrained nonlinear optimization problems. The BFGS iteratively approximates the Hessian matrix to find the minimum of a function, often requiring fewer iterations than the gradient descent (GD) algorithm [45]. The BFGS is widely used in machine learning (ML) [51] for training models, such as neural networks (NN) [52] and support vector machines (SVM) [53].

The limited-memory BFGS with bounds (L-BFGS-B) algorithm [42, 43] is a limited-memory quasi-Newton method to solve large-scale bound-constrained nonlinear optimization problems. The L-BFGS-B is mainly utilized for large dense problems and when there is difficulty in computing the Hessian matrix. The L-BFGS-B extends the standard L-BFGS algorithm [54] by incorporating simple bounds (lower and upper limits) on the variables for a function.

The sequential least squares programming (SLSQP) algorithm [44, 45] is a method for solving nonlinear optimization problems with constraints. The SLSQP iteratively solves a sequence of quadratic programming subproblems to find the optimal solutions. The SLSQP is particularly useful for problems with both equality and inequality constraints.

The constrained optimization by linear approximation (COBYLA) algorithm [26, 28, 39] is a derivative-free optimization algorithm

for solving problems with nonlinear inequality and equality constraints. COBYLA works by approximating the objective functions and their constraints using linear models, to find solutions without requiring the derivatives of these functions, where their gradients are difficult to compute.

Constrained optimization by quadratic approximations (COBYQA) [46] is a derivative-free optimization algorithm for solving general nonlinear optimization problems. COBYQA replaces COBYLA as a general derivative-free optimization solver, since COBYQA can handle unconstrained, bound-constrained, linearly constrained, and nonlinearly constrained problems.

## 3. Results and discussion

Arbitrary classical Boolean problems (as applications) are designed as Boolean oracles in various logical structures. These Boolean oracles are then solved using the BHT-QAOA for  $p$  repetitions, where  $p \geq 1$ . As shown in **Figure 2**, our experiments have utilized the ibm\_brisbane [55] QPU to perform the quantum processing domain of the BHT-QAOA, i.e., to execute the quantum circuit of an application in  $p$  repetitions. While the SciPy minimizer performs the classical processing domain of the BHT-QAOA, i.e., to optimize the numerical values of  $\gamma$  and  $\beta$  with the energy cost of their Hamiltonians ( $H_C$  and  $H_M$ ) in  $nfev$ . These applications are investigated using five optimization approximation algorithms of the SciPy minimizer, which are BFGS, L-BFGS-B, SLSQP, COBYLA, and COBYQA. In our research, we utilized the default settings [56] of the SciPy minimizer for all five algorithms, including tolerance ( $\text{tol} = 0.001 \times p$ ), constraints = (), bound = None, and maximum iterations ( $\text{maxiter} = 1000$ ).

Due to the limitations of our IBM Quantum Platform account, the complete architecture of the BHT-QAOA (shown in **Figure 2**) is completely simulated in the classical domain using IBM quantum libraries (Qiskit, Aer-EstimatorV2, and AerSimulator [55, 57, 58]), for 1024 resampling times as “shots” [59], and the simulated noisy model of the ibm\_brisbane QPU.

Note that, because of the limited physical connectivity of four neighboring qubits for the recent quantum layouts (architectures) of IBM QPUs [60, 61], the designed Boolean oracles (in various logical structures) must have  $n$  input qubits and  $m$  ancilla qubits (including  $f\text{qubit}$ ), where  $2 \leq n \leq 4$  and  $m \geq 0$ . To continue from our previous work in [20], we utilize the same five applications listed below. **Figure 3** demonstrates the quantum circuits of their designed Boolean oracles.

- (1) An arbitrary Boolean problem in the POS structure is stated in **Equation (2)** and shown in **Figure 3a**. Notice that this is a generalization of a practical problem to find a Minimum Set of Support Problem [3], and POS is equivalent to the AND Node in tree searching [62, 63].

$$(a \vee b \vee \neg c) \wedge (\neg a \vee c) \wedge (\neg b \vee c) \quad (2)$$

- (2) An arbitrary Boolean problem in the SOP structure is stated in **Equation (3)** and illustrated in **Figure 3b**. Notice that this is a generalized structure for the product of unification for all constraints for a problem, and SOP is equivalent to the OR Node in tree searching [62, 63].

$$(a \wedge b \wedge \neg c) \vee (\neg a \wedge c) \vee (\neg b \wedge c) \quad (3)$$

- (3) An arbitrary Boolean problem in the ESOP structure is stated in **Equation (4)** and shown in **Figure 3c**. Note that ESOP is equivalent to the Even–Odd Covering Problem for the product of constraints [64, 65].

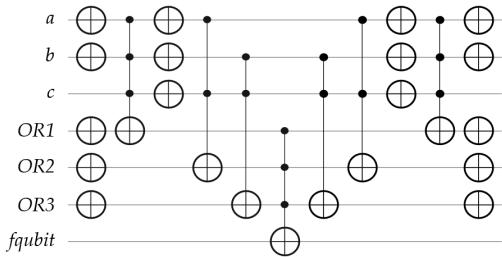
$$(a \wedge b \wedge \neg c) \oplus (\neg a \wedge c) \oplus (\neg b \wedge c) \quad (4)$$

- (4) A  $2 \times 2$  Sudoku game, which is the constraints satisfaction problem—satisfiability (CSP-SAT) [66, 67], is stated in **Equation (5)** and demonstrated in **Figure 3d**. Note that Sudoku is a special case of graph coloring problems.

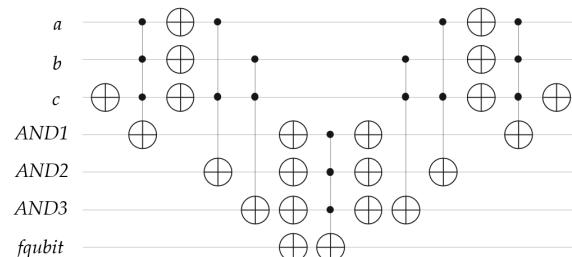
$$\begin{aligned} & (cell_1 \oplus cell_2) \wedge (cell_1 \oplus cell_3) \wedge \\ & (cell_2 \oplus cell_4) \wedge (cell_3 \oplus cell_4) \end{aligned} \quad (5)$$

- (5) A 4-bit conditioned half-adder (HA) digital circuit (as an arithmetic block), which is ORing two 1-bit sums and then ANDing them with one 1-bit carry-out, is stated in **Equation (6)** and illustrated in **Figure 3e,f**. Note that arithmetic blocks are used in many cryptarithmic problems, e.g., the problem of SEND+MORE=MONEY [68].

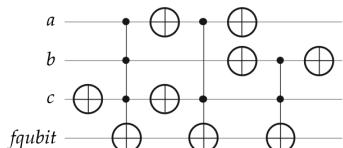
$$\begin{aligned} & [(a_0 \oplus b_0) \vee ((a_0 \wedge b_0) \oplus (a_1 \oplus b_1))] \wedge \\ & [(a_1 \wedge b_1) \vee ((a_0 \wedge b_0) \oplus (a_1 \oplus b_1))] \end{aligned} \quad (6)$$



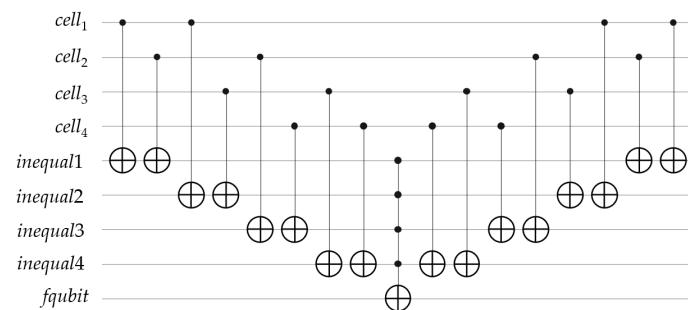
(a)



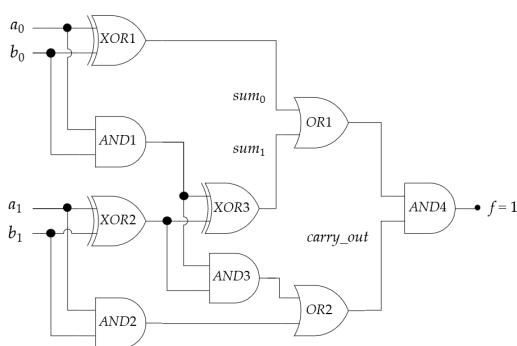
(b)



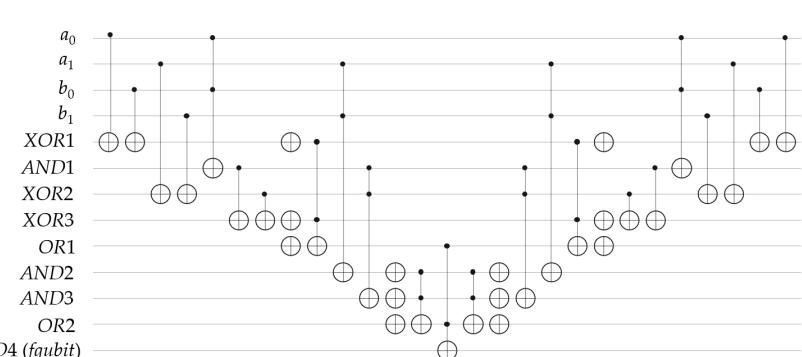
(c)



(d)



(e)



(f)

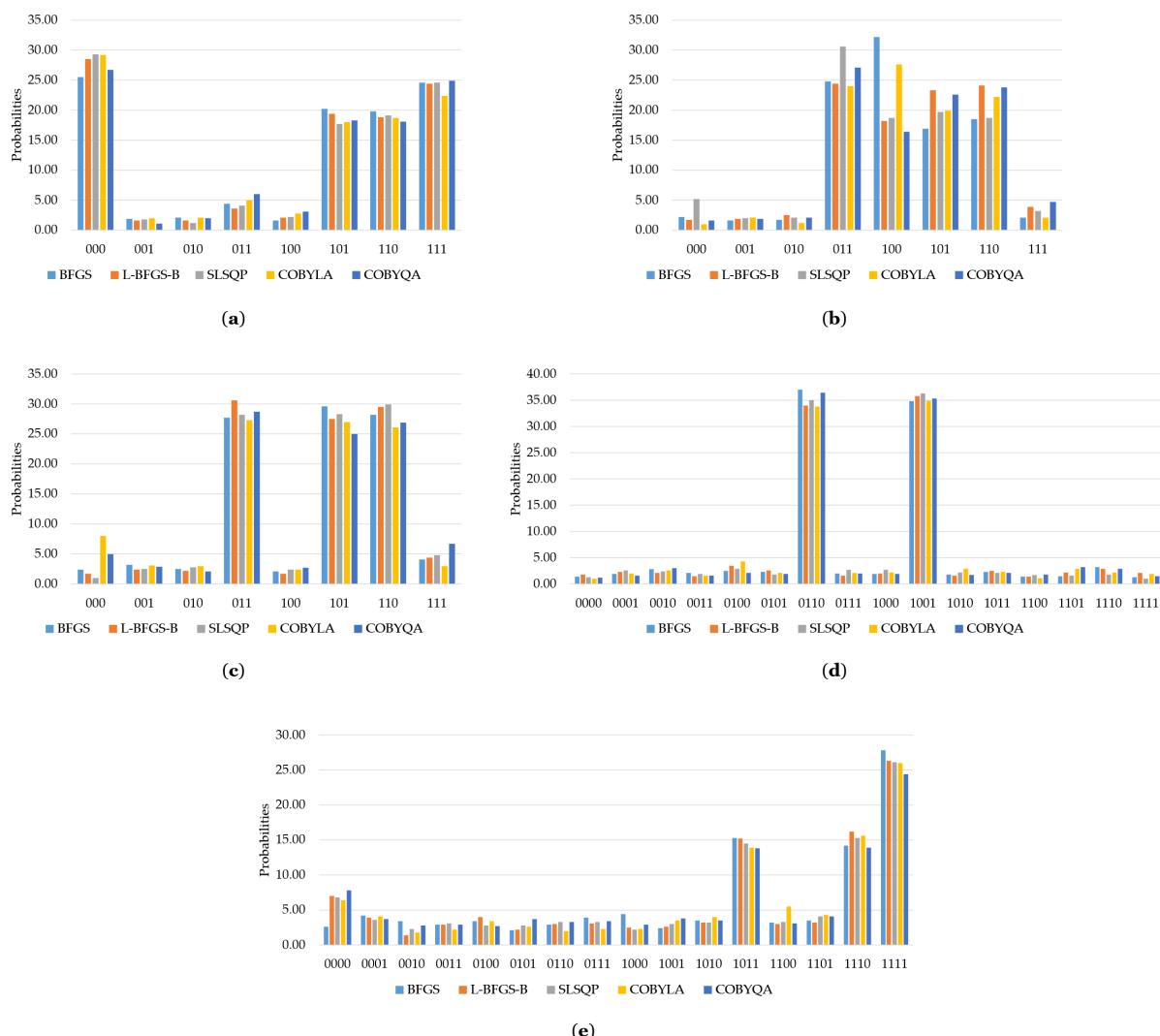
**Figure 3** • Schematics of the circuits for arbitrary Boolean problems: (a) the Boolean oracle in a POS structure representing **Equation (2)**, (b) the Boolean oracle in an SOP structure representing **Equation (3)**, (c) the Boolean oracle in an ESOP structure representing **Equation (4)**, (d) the Boolean oracle in a CNF-XOR SAT structure of  $2 \times 2$  Sudoku representing **Equation (5)**, (e) the classical 4-bit half-adder (HA) digital circuit for the two 2-bit numbers ( $A = a_1a_0$  and  $B = b_1b_0$ ), and (f) the Boolean oracle in a mixed structure representing **Equation (6)**. All qubits are initially set to the states of  $|0\rangle$ .

For the two performance metrics (the final utilization of  $nfev$  and the final quality of qubit measurements), **Table 2** presents the comparison of the final utilization of  $nfev$  for the five optimization approximation algorithms for all applications. Notice that, in **Table 2**, the results for an application are the average (mean) of

three repetitions for each optimization approximation algorithm. In addition, **Figure 4** illustrates the final measured solutions (as the final quality of qubit measurements) for all applications using these five algorithms.

**Table 2** • Comparison of the final utilization of  $nfev$  performance metric for the five optimization approximation algorithms. The desirable lowest (best) values of  $nfev$  are denoted in bold.

Applications	BFGS	L-BFGS-B	SLSQP	COBYLA	COBYQA
POS	<b>27</b>	<b>21</b>	<b>30</b>	49	38
SOP	<b>90</b>	185	181	296	165
ESOP	<b>75</b>	100	108	379	123
$2 \times 2$ Sudoku	<b>80</b>	115	<b>61</b>	515	124
4-bit HA circuit	<b>70</b>	85	<b>71</b>	99	93



**Figure 4** • The final measured solutions (as the final quality of qubit measurement performance metric) for all applications executed using five optimization approximation algorithms with simulated noisy model of the ibm\_brisbane QPU (1024 shots): (a) the four solutions for the Boolean oracle in POS structure of **Equation (2)** in  $p = 1$  for the measured bits sequence = {c, b, a}, (b) the four solutions for the Boolean oracle in SOP structure of **Equation (3)** in  $p = 2$  for the measured bits sequence = {c, b, a}, (c) the three solutions for the Boolean oracle in ESOP structure of **Equation (4)** in  $p = 2$  for the measured bits sequence = {c, b, a}, (d) the two solutions for the  $2 \times 2$  Sudoku game of **Equation (5)** in  $p = 2$  for the measured bits sequence = {cell<sub>4</sub>, cell<sub>3</sub>, cell<sub>2</sub>, cell<sub>1</sub>}, and (e) the three solutions for the 4-bit conditioned HA circuit of **Equation (6)** in  $p = 2$  for the measured bits sequence = {b<sub>1</sub>, b<sub>0</sub>, a<sub>1</sub>, a<sub>0</sub>}, where  $p$  is the number of repetitions for the quantum circuits of the Hamiltonians ( $H_C$  and  $H_M$ ) for an application.

As illustrated in **Figure 4a–e**, the highest bars in the histograms denote the final solutions for all applications using the five optimization approximation algorithms, and each histogram of an application denotes the results of (a) four solutions  $\{\bar{a}\bar{b}\bar{c}, a\bar{b}c, \bar{a}b\bar{c}, a b c\}$  for the Boolean oracle in the POS structure of **Equation (2)**; (b) four solutions  $\{a b \bar{c}, \bar{a}b\bar{c}, \bar{a}\bar{b}c, \bar{a}b c\}$  for the Boolean oracle in the SOP structure of **Equation (3)**; (c) three solutions  $\{a b \bar{c}, a \bar{b}c, \bar{a}b c\}$  for the Boolean oracle in the ESOP structure of **Equation (4)**; (d) two permutative solutions {solution 1:  $cell_1 = cell_4 = 0$  and  $cell_2 = cell_3 = 1$ ; solution 2:  $cell_1 = cell_4 = 1$  and  $cell_2 = cell_3 = 0$ } for the  $2 \times 2$  Sudoku game of **Equation (5)**; and (e) three solutions  $\{a_0a_1\bar{b}_0b_1, \bar{a}_0a_1b_0b_1, a_0a_1b_0b_1\}$  for the 4-bit conditioned half-adder (HA) circuit of **Equation (6)**.

Notice that, in **Figure 4**, (i) the first measured bit (the most-right) of a solution is equivalent to the first input qubit, (ii) the last measured bit (the most-left) of a solution is equivalent to the last input qubit, (iii) the higher probability of qubits indicates a solution, and (iv) the lower probability of qubits indicates a

non-solution. **Table 3** summarizes the final quality of qubit measurements for all five algorithms, with the BHT-QAOA calculated using **Equation (1)**.

Based on the evaluation of the two performance metrics in **Tables 2** and **3**, on the one hand, the BHT-QAOA successfully optimizes the numerical values of  $\gamma$  and  $\beta$  and finds all approximated solutions for all applications, as a proof of concept for utilizing any optimization approximation algorithm with the BHT-QAOA to solve arbitrary classical Boolean problems. On the other hand, both BFGS and SLSQP are the fastest optimization approximation algorithms for the BHT-QAOA, in the context of lower utilization of  $nfev$ . While the BFGS, L-BFGS-B, and SLSQP are the most accurate optimization approximation algorithms for the BHT-QAOA, in the context of a higher quality of qubit measurements. Concluding, the BFGS and SLSQP algorithms successfully demonstrate their capabilities to optimize the numerical values of  $\gamma$  and  $\beta$  and find all best-approximated solutions for all applications, as compared to other optimization approximation algorithms of SciPy minimizer.

**Table 3** • Summary of the final quality of qubit measurement performance metric for the five optimization approximation algorithms. The desirable highest (best) qualities are denoted in bold.

Applications	BFGS	L-BFGS-B	SLSQP	COBYLA	COBYQA
POS	<b>90.1</b>	<b>91.1</b>	<b>90.7</b>	88.3	88.0
SOP	<b>92.4</b>	90.0	87.7	<b>93.7</b>	89.9
ESOP	<b>85.5</b>	<b>87.6</b>	<b>86.4</b>	80.4	80.6
$2 \times 2$ Sudoku	<b>71.8</b>	69.8	<b>71.3</b>	68.7	<b>71.7</b>
4-bit HA circuit	<b>57.3</b>	<b>57.7</b>	55.9	55.5	<b>52.1</b>

Accordingly, the BHT-QAOA with both BFGS and SLSQP approximation optimization algorithms will provide broad opportunities in investigating many classical Boolean problems in the hybrid classical–quantum domain, which are neither designed nor solved using the standard QAOA [9, 10]. Thus, various classical Boolean problems for the applications of digital circuits and synthesizers can be directly realized as Hamiltonians ( $H_C$  and  $H_M$ ) and successfully solved using the BHT-QAOA with these two BFGS and SLSQP algorithms for highly approximated solutions correctness.

## 4. Conclusions

Our previous research work demonstrated that Grover’s algorithm has two drawbacks in solving arbitrary Boolean oracles in various logical structures. Firstly, Grover’s algorithm is not asymptotically optimal in finding all correct solutions for such oracles. Secondly, many Boolean oracles require additional ancilla qubits and mirror (uncomputing) sub-circuits of an oracle, which substantially increase the cost of quantum Boolean-based circuit design. For this reason, our BHT-QAOA (Boolean–Hamiltonians Transform for Quantum Approximate Optimization Algorithm) was introduced to solve arbitrary classical Boolean problems as Hamiltonians in the hybrid classical–quantum domain, to find all correct optimized approximated solutions for such problems without requiring any additional ancilla qubits and mirror sub-circuits.

For finding the best-approximated solutions for arbitrary Boolean problems, the goal of this article is to re-investigate the BHT-QAOA using five classical optimization approximation algorithms and two proposed performance metrics. These algorithms are BFGS, L-BFGS-B, SLSQP, COBYLA, and COBYQA. While the two proposed performance metrics are (i) the final utilization of  $nfev$  and (ii) the final quality of qubit measurements for the best-approximated solutions, where  $nfev$  is the number of function evaluations for a problem between a quantum computer and a classical optimization approximation algorithm.

In this study, arbitrary Boolean applications were constructed as Boolean oracles in various logical structures, and then BHT-QAOA successfully searched for all optimized approximated solutions for these applications using all five optimization approximation algorithms. Experimentally, it has been demonstrated that the BFGS and SLSQP are the fastest approximation algorithms, requiring fewer utilizations of  $nfev$ . BFGS, L-BFGS-B, and SLSQP are the most accurate approximation algorithms, achieving higher-quality qubit measurements. Both the BFGS and SLSQP optimization approximation algorithms successfully demonstrated their capabilities in finding all best-approximated solutions for arbitrary Boolean applications using BHT-QAOA.

On the one hand, we believe that comparing and evaluating different optimization approximation algorithms with the BHT-QAOA can help potential users to effectively solve decision and optimization

problems using customizable logical structures, since the BHT-QAOA can be considered a promising successor to Grover's algorithm in solving various practical problems. On the other hand, the main contribution of this paper for selecting an appropriate optimization approximation algorithm for BHT-QAOA matters, since our second observation based on several logical structures indicates that the BFGS algorithm is the first winner and the SLSQP algorithm is the second winner. Our future work will deeply investigate these two algorithms for bigger applications by studying different hybrid variational algorithms.

In conclusion, further classical Boolean problems can be investigated for arbitrary logical structures for practical engineering applications in the fields of digital logic synthesis, robotics, machine learning, etc. The BHT-QAOA, with the BFGS and SLSQP optimization approximation algorithms, will successfully solve such practical applications effectively in the hybrid classical–quantum domain.

## Funding

This research received no external funding.

## Author contributions

Conceptualization, A.A.; methodology, A.A.; formal analysis, A.A.; writing—original draft preparation, A.A.; writing—review and editing, A.A. and M.P.; visualization, A.A.; supervision, A.A. and M.P. All authors have read and agreed to the published version of the manuscript.

## Conflict of interest

The authors declare that they have no competing interests.

## Data availability statement

All data supporting the findings of this publication are available within this article.

## Additional information

Received: 2025-07-29

Accepted: 2025-10-27

Published: 2025-11-18

*Academia Quantum* papers should be cited as *Academia Quantum* 2025, ISSN 3064-979X, <https://doi.org/10.20935/AcadQuant7996>. The journal's official abbreviation is *Acad. Quant.*

## Publisher's note

Academia.edu Journals stays neutral with regard to jurisdictional claims in published maps and institutional affiliations. All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

## Copyright

© 2025 copyright by the authors. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## References

1. Goemans MX, Williamson DP. .879-approximation algorithms for max cut and max 2sat. Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing; 1994 May 23; Montreal, QC, Canada. 1994. p. 422–31.
2. Rendl F, Rinaldi G, Wiegele A. Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. *Math Program.* 2010;121(2):307–35. doi: 10.1007/s10107-008-0235-8
3. Perkowski M. Inverse problems, constraint satisfaction, reversible logic, invertible logic and Grover quantum oracles for practical problems. *Sci Comput Program.* 2022; 218:102775. doi: 10.1016/j.scico.2022.102775
4. Bravyi S, Kliuchnikov A, Koenig R, Tang E. Hybrid quantum-classical algorithms for approximate graph coloring. *Quantum.* 2022;6:678. doi: 10.22331/q-2022-03-30-678
5. Hou W, Perkowski M. Quantum-based algorithm and circuit design for bounded Knapsack optimization problem. *Quantum Inf Comput.* 2020;20(9):766–86. doi: 10.26421/QIC20.9-10-4
6. Li Y, Tsai E, Perkowski M, Song X. Grover-based Ashenhurst-Curtis decomposition using quantum language quipper. *Quantum Inf Comput.* 2019;19(1–2):35–66. doi: 10.26421/QIC19.1-2-4
7. Ruan Y, Marsh S, Xue X, Liu Z, Wang J. The quantum approximate algorithm for solving traveling salesman problem. *Comput Mater Continua.* 2020;63(3):1237–47. doi: 10.32604/cmc.2020.010001
8. Qian W, Basili RA, Eshaghian-Wilner MM, Khokhar A, Luecke G, Vary JP. Comparative study of variations in quantum approximate optimization algorithms for the traveling salesman problem. *Entropy.* 2023;25(8):1238. doi: 10.3390/e25081238
9. Farhi E, Goldstone J, Gutmann S. A quantum approximate optimization algorithm. *arXiv.* 2014. arXiv:1411.4028. doi: 10.48550/arXiv.1411.4028
10. Farhi E, Goldstone J, Gutmann S, Neven H. Quantum algorithms for fixed qubit architectures. *arXiv.* 2017. arXiv: 1703.06199. doi: 10.48550/arXiv.1703.06199
11. Symons BC, Galvin D, Sahin E, Alexandrov V, Mensa S. A practitioner's guide to quantum algorithms for optimisation problems. *J Phys A Math Theor.* 2023;56(45):453001. doi: 10.1088/1751-8121/adoof0
12. Brandhofer S, Braun D, Dehn V, Hellstern G, Hüls M, Ji Y, et al. Benchmarking the performance of portfolio optimization with QAOA. *Quantum Inf Process.* 2022;22(1):25. doi: 10.1007/s11128-022-03766-5

13. Grange C, Poss M, Bourreau E. An introduction to variational quantum algorithms for combinatorial optimization problems. *4OR*. 2023;21(3):363–403. doi: 10.1007/s10288-023-00549-1
14. Borle A, Elfving V, Lomonaco SJ. Quantum approximate optimization for hard problems in linear algebra. *SciPost Phys Core*. 2021;4(4):031. doi: 10.21468/SciPostPhysCore.4.4.031
15. Choi J, Kim J. A tutorial on quantum approximate optimization algorithm (QAOA): fundamentals and applications. Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC); 2019 Oct 16; Jeju Island, Republic of Korea. New York (NY): IEEE; 2019. p. 138–42.
16. Grover LK. A fast quantum mechanical algorithm for database search. Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing; 1996 Jul 1; Philadelphia, PA, USA. 1996. p. 212–19.
17. Grover LK. Quantum mechanics helps in searching for a needle in a haystack. *Phys Rev Lett*. 1997;79(2):325. doi: 10.1103/PhysRevLett.79.325
18. Grover LK. A framework for fast quantum mechanical algorithms. Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing; 1998 May 23; Dallas, TX, USA. 1998. p. 53–62.
19. Al-Bayaty A, Perkowski M. A concept of controlling Grover diffusion operator: a new approach to solve arbitrary boolean-based problems. *Sci Rep*. 2024;14(1):23570. doi: 10.1038/s41598-024-74587-y
20. Al-Bayaty A, Perkowski M. BHT-QAOA: the generalization of quantum approximate optimization algorithm to solve arbitrary boolean problems as hamiltonians. *Entropy*. 2024; 26(10):843. doi: 10.3390/e26100843
21. Moussa C, Wang H, Bäck T, Dunjko V. Unsupervised strategies for identifying optimal parameters in quantum approximate optimization algorithm. *EPJ Quantum Technol*. 2022;9(1):11. doi: 10.1140/epjqt/s40507-022-00131-4
22. Amosy O, Danzig T, Lev O, Porat E, Chechik G, Makmal A. Iteration-free quantum approximate optimization algorithm using neural networks. *Quantum Mach Intell*. 2024; 6(2):38. doi: 10.1007/s42484-024-00159-y
23. Herrman R, Lotshaw PC, Ostrowski J, Humble TS, Siopsis G. Multi-angle quantum approximate optimization algorithm. *Sci Rep*. 2022;12(1):6781. doi: 10.1038/s41598-022-10555-8
24. Wurtz J, Lykov D. Fixed-angle conjectures for the quantum approximate optimization algorithm on regular Max-Cut graphs. *Phys Rev A*. 2021;104(5):052419. doi: 10.1103/PhysRevA.104.052419
25. Crooks GE. Performance of the quantum approximate optimization algorithm on the maximum cut problem. *arXiv*. 2018. arXiv:1811.08419. doi: 10.48550/arXiv.1811.08419
26. Fernández-Pendás M, Combarro EF, Vallecorsa S, Ranilla J, Rúa IF. A study of the performance of classical minimizers in the quantum approximate optimization algorithm. *J Comput Appl Math*. 2022;404:113388. doi: 10.1016/j.cam.2021.113388
27. Powell MJ, Gomez S. Advances in optimization and numerical analysis. In: Mathematics and its applications. Berlin and Heidelberg: Springer; 1994. p. 51–67.
28. Larson J, Menickelly M, Wild SM. Derivative-free optimization methods. *Acta Num*. 2019;28:287–404. doi: 10.1017/S0962492919000060
29. Figgatt C, Maslov D, Landsman KA, Linke NM, Debnath S, Monroe C. Complete 3-qubit Grover search on a programmable quantum computer. *Nat Commun*. 2017;8(1):1918. doi: 10.1038/s41467-017-01904-7
30. Wakerly JF. Digital design: principles and practices. 4th ed. Tamil Nadu: Pearson Education India; 2008.
31. Zhang LX, Huang W. A note on the invariance principle of the product of sums of random variables. *Electron Commun Probab*. 2007;12:59–64. doi: 10.48550/arXiv.math/0610515
32. Zimmermann R, Tran DQ. Optimized synthesis of sum-of-products. Proceedings of the Thirty-Seventh Asilomar Conference on Signals, Systems & Computers; 2003 Nov 9; Pacific Grove, CA, USA. Vol. 1. New York (NY): IEEE; 2003. p. 867–72.
33. Mishchenko A, Perkowski M. Fast heuristic minimization of exclusive sums-of-products. Proceedings of the RM'2001 Workshop; 2001 Aug 9; Portland, OR, USA. p. 242–50.
34. Sasao T. EXMIN2: A simplification algorithm for exclusive-OR-sum-of-products expressions for multiple-valued-input two-valued-output functions. *IEEE Trans Comput-Aided Des Integr Circuits Syst*. 2002;12(5):621–32. doi: 10.1109/43.277608
35. Ibrahimi M, Kanoria Y, Kraning M, Montanari A. The set of solutions of random XORSAT formulae. Proceedings of the Twenty-Third Annual ACM-SIAM symposium on Discrete Algorithms; 2012 Jan 17; Kyoto, Japan. Philadelphia (PA): Society for Industrial and Applied Mathematics; 2012. p. 760–79.
36. Soos M, Meel KS. BIRD: engineering an efficient CNF-XOR SAT solver and its applications to approximate model counting. Proceedings of the AAAI Conference on Artificial Intelligence; 2019 Jul 17; Honolulu, HI, USA. 2019. Vol. 33, p. 1592–9.
37. Stankovic RS, Sasao T. A discussion on the history of research in arithmetic and Reed-Muller expressions. *IEEE Trans Comput-Aided Des Integr Circuits Syst*. 2001; 20(9):1177–9. doi: 10.1109/43.945313
38. Kurgalin S, Borzunov S. Concise guide to quantum computing. Cham: Springer; 2021.

39. Lavrijsen W, Tudor A, Müller J, Iancu C, De Jong W. Classical optimizers for noisy intermediate-scale quantum devices. Proceedings of the 2020 IEEE International Conference on Quantum Computing and Engineering (QCE); 2020 Oct 12; Denver, CO, USA. New York (NY): IEEE; 2020. p. 267–77.
40. Yuan YX. A modified BFGS algorithm for unconstrained optimization. *IMA J Numer Anal.* 1991;11(3):325–32. doi: 10.1093/imanum/11.3.325
41. Dai YH. Convergence properties of the BFGS algorithm. *SIAM J Optim.* 2002;13(3):693–701. doi: 10.1137/S1052623401383455
42. Zhu C, Byrd RH, Lu P, Nocedal J. Algorithm 778: L-BFGS-B: fortran subroutines for large-scale bound-constrained optimization. *ACM Trans Math Softw.* 1997;23(4):550–60. doi: 10.1145/279232.279236
43. Liu DC, Nocedal J. On the limited memory BFGS method for large scale optimization. *Math Program.* 1989;45(1): 503–28. doi: 10.1007/BF01589116
44. Ma Y, Gao X, Liu C, Li J. Improved SQP and SLSQP algorithms for feasible path-based process optimisation. *Comput Chem Eng.* 2024;188:108751. doi: 10.1016/j.compchemeng.2024.108751
45. Bonnans JF, Gilbert JC, Lemaréchal C, Sagastizábal CA. Numerical optimization: theoretical and practical aspects. Berlin and Heidelberg: Springer; 2006.
46. Ragonneau TM. Model-based derivative-free optimization methods and software. arXiv. 2022. arXiv:2210.12018. doi: 10.48550/arXiv.2210.12018
47. Ebendt R, Fey G, Drechsler R. Advanced BDD optimization. Boston (MA): Springer; 2005.
48. Wille R, Drechsler R. Effect of BDD optimization on synthesis of reversible and quantum logic. *Electron Notes Theor Comput Sci.* 2010;253(6):57–70. doi: 10.1016/j.entcs.2010.02.006
49. Balasubramanian P, Arisaka R, Arabnia HR. RB\_DSOP: a rule based disjoint sum of products synthesis method. Proceedings of the International Conference on Computer Design (CDES); 2012 Nov 5–8; New York, NY, USA. 2012. p. 1.
50. Hadfield S. On the representation of Boolean and real functions as Hamiltonians for quantum computing. *ACM Trans Quantum Comput.* 2021;2(4):1–21. doi: 10.1145/3478519
51. Al-Bayaty A, Perkowski M. COVID-19 features detection using machine learning models and classifiers. In: The Science Behind the COVID pandemic and healthcare technology solutions. Cham: Springer International Publishing; 2022. p. 379–403.
52. Zhang Y, Mu B, Zheng H. Link between and comparison and combination of Zhang neural network and quasi-Newton BFGS method for time-varying quadratic minimization. *IEEE Trans Cybern.* 2013;43(2):490–503. doi: 10.1109/TSMCB.2012.2210038
53. Li S, Tan M. Tuning SVM parameters by using a hybrid CLPSO–BFGS algorithm. *Neurocomputing.* 2010;73(10–12):2089–96. doi: 10.1016/j.neucom.2010.02.013
54. Xiao Y, Wei Z, Wang Z. A limited memory BFGS-type method for large-scale unconstrained optimization. *Comput Math Appl.* 2008;56(4):1001–9. doi: 10.1016/j.camwa.2008.01.028
55. Karimi N, Elyasi SN, Yahyavi M. Implementation and measurement of quantum entanglement using IBM quantum platforms. *Phys Scr.* 2024;99(4):045121. doi: 10.1088/1402-4896/ad3518
56. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat Methods.* 2020; 17(3):261–72. doi: 10.1038/s41592-019-0686-2
57. Wille R, Van Meter R, Naveh Y. IBM’s Qiskit tool chain: working with and developing for real quantum computers. Proceedings of the 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE); 2019 Mar 25; Florence, Italy. New York (NY): IEEE; 2019. p. 1234–40.
58. Georgopoulos K, Emary C, Zuliani P. Modeling and simulating the noisy behavior of near-term quantum computers. *Phys Rev A.* 2021;104(6):062432. doi: 10.1103/PhysRevA.104.062432
59. Rao P, Yu K, Lim H, Jin D, Choi D. Quantum amplitude estimation algorithms on IBM quantum devices. Proceedings of the Quantum Communications and Quantum Imaging XVIII; 2020 Aug 22; San Diego, CA, USA. Vol. 11507. Bellingham (WA): SPIE; 2020. p. 49–60.
60. Al-Bayaty A, Perkowski M. GALA-n: generic architecture of layout-aware n-bit quantum operators for cost-effective realization on IBM quantum computers. arXiv. 2023. arXiv:2311.06760. doi: 10.48550/arXiv.2311.06760
61. Al-Bayaty A, Song X, Perkowski M. CALA-n: a quantum library for realizing cost-effective 2-, 3-, 4-, and 5-bit gates on IBM quantum computers using bloch sphere approach, clifford+ t gates, and layouts. arXiv. 2024. arXiv:2408.01025. doi: 10.48550/arXiv.2408.01025
62. Martelli A, Montanari U. Optimizing decision trees through heuristically guided search. *Commun ACM.* 1978;21(12): 1025–39. doi: 10.1145/359657.359664
63. Nilsson NJ. Problem solving methods in artificial intelligence. New York (NY): McGraw-Hill; 1971.
64. Perkowski MA, Dysko P, Falkowski BJ. Two learning methods for a tree-search combinatorial optimizer. Proceedings of the IEEE Int. Conference on Computing and Communications; 1990 Mar 21; Scottsdale, AZ, USA. 1990. p. 606–13.
65. Zhang Q, Wang P, Hu J, Zhang H. Cube-based synthesis of ESOPs for large functions. *Chin J Electron.* 2018;27(3):527–34. doi: 10.1049/cje.2018.01.006
66. Simonis H. Sudoku as a constraint problem. Proceedings of the CP Workshop on Modeling and Reformulating Constraint Satisfaction Problems; 2005 Oct 1; Sitges, Spain. Vol. 12. San Jose (CA): Citeseer; 2005. p. 13–27.

67. Lynce I, Ouaknine J. Sudoku as a SAT Problem. Proceedings of the 9th Symposium on Artificial Intelligence & Mathematics (AI&M); 2006 Jan 4; Fort Lauderdale, FL, USA. 2006.
68. Epstein D. On the NP-completeness of cryptarithms. ACM SIGACT News. 1987;18(3):38–40. doi: 10.1145/24658.24662