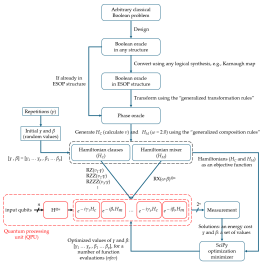
Nov 30, 2024

BHT-QAOA: Generalizing Quantum Approximate Optimization Algorithm to Solve Boolean Problems as Hamiltonians



DOI
https://dx.doi.org/10.17504/protocols.io.dm6gp9bd8vzp/v1

Ali Al-Bayaty[1], Marek Perkowski[1]

[1]Portland State University

Ali Al-Bayaty: Department of Electrical and Computer Engineering
Marek Perkowski: Department of Electrical and Computer Engineering

Ali Al-Bayaty
Portland State University

DOI: https://dx.doi.org/10.17504/protocols.io.dm6gp9bd8vzp/v1

Protocol Citation: Ali Al-Bayaty, Marek Perkowski 2024. BHT-QAOA: Generalizing Quantum Approximate Optimization Algorithm to Solve Boolean Problems as Hamiltonians. **protocols.io** https://dx.doi.org/10.17504/protocols.io.dm6gp9bd8vzp/v1

Manuscript citation:
A. Al-Bayaty and M. Perkowski, "BHT-QAOA: The generalization of quantum approximate optimization algorithm to solve arbitrary Boolean problems as Hamiltonians," *Entropy*, vol. 26, no. 10, p. 843, 2024. https://doi.org/10.3390/e26100843

Protocol status: Working
We use this protocol and it's working

Created: November 28, 2024

Last Modified: November 30, 2024

Protocol Integer ID: 113113

Keywords: Boolean oracles, Phase oracles, logical structures, Hamiltonians, Boolean problems, quantum approximate optimization algorithm, QAOA, classical-quantum algorithm, quantum boolean oracles into quantum phase oracle, design classical boolean problems into quantum boolean oracle, quantum boolean oracle, quantum approximate optimization algorithm, quantum phase oracle, generalizing quantum approximate optimization algorithm, quantum algorithm, hamiltonians transform for qaoa, arbitrary classical boolean problems as hamiltonian, boolean problems as hamiltonian, quantum gate, using qaoa, solutions for classical boolean problem, quantum phase, arbitrary classical boolean problem, classical boolean problem, many classical boolean, problems as hamiltonian, several algorithm, practical engineering applications of several algorithm, solving combinatorial optimization problem, qaoa, quantum domain, qubit, combinatorial optimization problem, generated hamiltonian, numbers of qubit, hamiltonians transform, hamiltonian, algorithm,

Disclaimer

Abstract

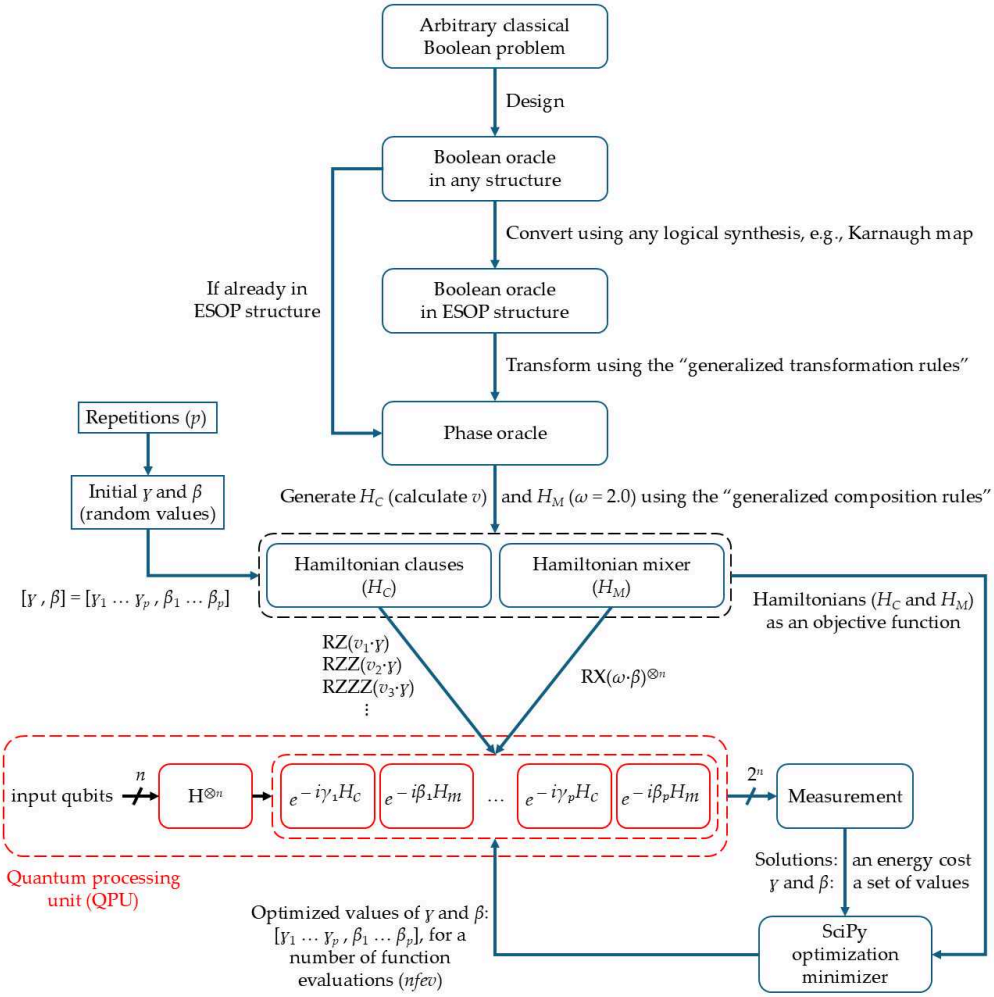
The BHT-QAOA is a hybrid classical-quantum algorithm that solves arbitrary classical Boolean problems as Hamiltonians in the quantum domain, using the quantum approximate optimization algorithm (QAOA) [1]. The BHT-QAOA stands for the "Boolean-Hamiltonians Transform for QAOA" [2].

Research and studies are mainly focused on solving combinatorial optimization problems using QAOA, e.g., the MaxCut problem [1]. However, the BHT-QAOA adds an additional capability to QAOA to find all optimized approximated solutions for classical Boolean problems, as expressed in the following steps and demonstrated in the figure below.

1. Design classical Boolean problems into quantum Boolean oracles in different logical structures, such as POS, SOP, ESOP, CSP-SAT, XOR-SAT, just to name a few.
2. Convert these quantum Boolean oracles into quantum Phase oracles.
3. Transform these quantum Phase oracles into the Hamiltonians (H_C and H_M) of QAOA.

Please observe that, from the aforementioned steps, the total utilized numbers of qubits and quantum gates are significantly reduced for the final generated Hamiltonians (H_C and H_M) of QAOA.

Accordingly, the BHT-QAOA will provide broad opportunities to solve many classical Boolean-based problems as Hamiltonians, for the practical engineering applications of several algorithms, digital synthesizers, robotics, machine learning, just to name a few, in the hybrid classical-quantum domain.



The architecture of BHT-QAOA in two processing domains: (i) the classical domain as denoted in blue, and (ii) the quantum domain as denoted in red [2].

The BHT-QAOA Protocol

1 Design a classical Boolean problem as a quantum Boolean oracle in a desirable logical structure, e.g., POS, SOP, ESOP, CSP-SAT, XOR-SAT, just to name a few.

> **Note**
>
> There is no need to optimize a quantum Boolean oracle, by removing the identical neighboring Pauli-X (X) gates, since all such gates are required to generate the Hamiltonians (H_C and H_M) of BHT-QAOA and calculate their coefficients (v and ω), respectively.

2 Convert the designed quantum Boolean oracle (in any logical structure) into its equivalent quantum Boolean oracle in ESOP (or DSOP) structure, using a logical synthesis method, e.g., ESOP synthesis [3], Karnaugh map synthesis [4], binary decision diagram (BDD) synthesis [5, 6], just to name a few.

Note

After converting a quantum Boolean oracle (in any logical structure) into its equivalent quantum Boolean oracle in ESOP (or DSOP) structure, all mirrored (uncomputed) gates and ancilla qubits (except for the output qubit "*fqubit*") are removed, i.e., the quantum cost and depth are dramatically minimized!

3 Transform the converted quantum Boolean oracle in ESOP (or DSOP) structure into a quantum Phase oracle, using our three proposed "generalized transformation rules" [2] as follows, where $n \geq 3$ qubits, *j* and *k* are the indices of input (control) qubits (*q*), and *fqubit* is the output (functional) qubit.

Rule 1: A Feynman (CNOT) gate is transformed into a Pauli-Z (Z) gate, if and only if the following algebraic formula is satisfied: $q_j \oplus fqubit = -(-1)^{q_j}$

Rule 2: A Toffoli gate is transformed into a controlled-Z (CZ) gate, if and only if the following algebraic formula is satisfied: $(q_j \wedge q_k) \oplus fqubit = -(-1)^{q_j \cdot q_k}$

Rule 3: An *n*-bit Toffoli gate is transformed into an (*n* – 1)-bit multi-controlled Z (MCZ) gate, if and only if the following algebraic formula is satisfied: $(\wedge_{j=1}^{n-1} q_j) \oplus fqubit = -(-1)^{\prod_{j=1}^{n-1} q_j}$

Note

Our three "generalized transformation rules" are efficiently generalized from the technique originally discussed by Figgatt *et al.* [7] for transforming 4-bit Toffoli gates into 3-bit controlled-Z (CCZ) gates.

Note

After transforming a quantum Boolean oracle in ESOP (or DSOP) structure into a quantum Phase oracle, the output qubit "*fqubit*" is omitted, i.e., the quantum circuit of a Phase oracle only consists of input qubits without any mirrored (uncomputed) gates and ancilla qubits, and its final quantum cost and depth are significantly reduced!

4 The transformed quantum Phase oracle is then utilized to generate the Hamiltonian (H_C) of BHT-QAOA, using our four proposed "generalized composition rules (H_g)" [2] as follows, where $n \geq 3$ qubits, j and k are the indices of qubits (q), Z_j is the RZ gate applied on q_j, and $Z_Q = \{Z_j, Z_k, ..., Z_jZ_k, ...\}$.

Rule 1: Construct H_g of $(-\frac{1}{2}I + \frac{1}{2}Z_j)$, for a Pauli-Z (Z) gate.

Rule 2: Construct H_g of $(-\frac{1}{4}I + \frac{1}{4}(Z_j + Z_k - Z_jZ_k))$, for a controlled-Z (CZ) gate.

Rule 3: Construct H_g of $(\frac{1}{2^n}\prod_{j=1}^{n}(-1)^{j+1}(I - Z_j))$, for an n-bit multi-controlled Z (MCZ) gate.

Rule 4: Invert the signs ($\pm$) of all jth qubits (q_j) in Z_Q, for a Pauli-X (X) gate.

Note

Our four "generalized composition rules (H_g)" are efficiently generalized from the composition rules originally discussed by Hadfield [8] for generating H_C from a set of Hamiltonians (H_f), which represent a variety of Boolean functions as simpler clauses.

5 Generate H_C and calculate its v coefficients as expressed in the following stages [2] (starting from the left side of the quantum circuit of a Phase oracle):

1. Construct one H_g for one Z, CZ, or MCZ gate, using **Rule 1**, **Rule 2**, or **Rule 3** of the generalized composition rules, respectively.
2. If there are X gates (including their mirrored "uncomputed" gates) surrounding Z, CZ, or MCZ gate in Stage 1, then apply **Rule 4** of the generalized composition rules on the H_g from Stage 1 to construct a new H_g. If not, proceed to Stage 3.
3. Repeat Stages 1 and 2 to construct another H_g, until there are no remaining Z, CZ, and MCZ gates.
4. Group all constructed sets of H_g into one Hamiltonian, which is H_C.
5. Calculate (add or subtract) all the identical terms of H_C to find the v coefficients of H_C.

Note

In QAOA, H_C is the Hamiltonian Clauses, which is the ansatz Hamiltonian oracle of a classical problem, as the unitary operator ($e^{-i\,\gamma\,H_C}$). H_C is a set of non-connected nodes as $RZ_j(v{\cdot}\gamma)$ gates, two connected nodes as $RZ_jZ_k(v{\cdot}\gamma)$ gates, and so on, where j and k are the indices of input qubits, v's are the coefficients (as the time evolutions for H_C), and γ's are the parameterized angular rotations. Note that γ's rotate between the angles of [0, 2π] [1, 2].

6 Generate H_M and calculate its ω coefficients. Since β's of H_M rotate between [0, π] [1, 2], the coefficients (ω) are set to cover all the range between [0, 2π] for possible phase values of RX gates in H_M. In other words, to find all optimized approximated solutions for an arbitrary classical Boolean problem, ω = 2.0 for all n $RX(\omega{\cdot}\beta)$ gates of all n qubits of H_M.

Note

In QAOA, H_M is the Hamiltonian Mixer, which is the ansatz Hamiltonian operator, for the sum of all n qubits as the unitary operator ($e^{-i\,\beta\,H_M}$). H_M is a set of $RX(\omega{\cdot}\beta)$ gates for all n input qubits, where ω's are the coefficients (as the time evolutions for H_M), and β's are the parameterized angular rotations. Note that H_M acts as the quantum diffusion operator of QAOA analogous to the quantum diffusion operator of Grover's algorithm [9-11].

7 For p repetitions, initially randomize the numerical values of γ as $[\gamma_1, \gamma_2, \ldots, \gamma_p]$ and β as $[\beta_1, \beta_2, \ldots, \beta_p]$, and then plug them into the quantum gates of H_C and H_M, respectively, where $p \geq 1$.

8 Construct the quantum circuit of BHT-QAOA that consists of two segments:

1. ***n* Hadamard (H) gates:** applied to all n input qubits. Note that these n input qubits represent the literals (variables) of a classical Boolean problem, without any utilized ancilla qubits and *fqubit*, where $n \geq 2$.
2. ***p* repetitions of H_C and H_M:** in the sequence of $\{e^{-i\,\gamma_1\,H_C}\ e^{-i\,\beta_1\,H_M}, e^{-i\,\gamma_2\,H_C}\ e^{-i\,\beta_2\,H_M}, \ldots, e^{-i\,\gamma_p\,H_C}\ e^{-i\,\beta_p\,H_M}\}$, where $p > 1$.

9 Execute the constructed quantum circuit of BHT-QAOA in the quantum domain (or simulate it in the classical domain), and then classically measure all n input qubits into n bits.

10 After measuring all n input qubits of BHT-QAOA as solutions, employ a classical optimization minimizer, e.g., SciPy optimization minimizer [2, 12], to optimize the numerical values of γ and β for better-approximated solutions, in a number of function evaluations (*nfev*).

In general, a classical optimization minimizer utilizes the following three cofactors for better-approximated solutions:

1. H_C and H_M (in p repetitions), as the "objective function" needs to be minimized.
2. Measured solutions of BHT-QAOA, as the "energy cost" of the objective function.
3. Previously calculated γ and β, as their "numerical values" need to be optimized.

Note

For the SciPy optimization minimizer, we employed the constrained optimization by linear approximation (COBYLA) algorithm [2, 12]. COBYLA is a parametric iterative method for derivative-free constrained optimization, which updates and minimizes the approximated numerical values (as γ and β) of an objective function (as H_C and H_M) for a lower energy cost (as the measured solutions).

11 Finally, after updating the numerical values of γ and β, **Step 9** and **Step 10** are concurrently repeated between the quantum and classical domains (or between the simulated quantum and classical domains) for *nfev*, until all optimized approximated solutions for a classical Boolean problem are found or stopping based on a pre-defined "halt condition".

Protocol references

[1] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," 2014, *arXiv:1411.4028*.

[2] A. Al-Bayaty and M. Perkowski, "BHT-QAOA: The generalization of quantum approximate optimization algorithm to solve arbitrary Boolean problems as Hamiltonians," *Entropy*, vol. 26, no. 10, p. 843, 2024.

[3] A. Mishchenko and M. Perkowski, "Fast heuristic minimization of exclusive sums-of-products," in *Proc. RM'2001 Workshop*, 2001, pp. 242-250.

[4] J.F. Wakerly, *Digital Design: Principles and Practices*, 4th ed. New Delhi, India: Pearson Education, 2014. ISBN 978-0131863897.

[5] R. Ebendt, G. Fey, and R. Drechsler, *Advanced BDD Optimization*. Dordrecht, The Netherlands: Springer, 2005. ISBN 978-0387254531.

[6] R. Wille and R. Drechsler, "Effect of BDD optimization on synthesis of reversible and quantum logic," *Electronic Notes in Theoretical Computer Science*, vol. 253, pp. 57-70, 2010.

[7] C. Figgatt, D. Maslov, K.A. Landsman, N.M. Linke, S. Debnath, and C. Monroe, "Complete 3-qubit Grover search on a programmable quantum computer," *Nature Communications*, vol. 8, p. 1918, 2017.

[8] S. Hadfield, "On the representation of Boolean and real functions as Hamiltonians for quantum computing," *ACM Trans. on Quantum Computing* (*TQC*), vol. 2, pp. 1-21, 2021.

[9] L.K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. of the 28th Ann. ACM Symp. on Theory of Computing*, 1996, pp. 212-219.

[10] A. Al-Bayaty and M. Perkowski, "A concept of controlling Grover diffusion operator: A new approach to solve arbitrary Boolean-based problems," *Scientific Reports*, vol. 14, pp. 1-16, 2024.

[11] A. Al-Bayaty and M. Perkowski, "Grover controlled-diffuser (CUs) for quantum Boolean oracles of Grover's algorithm," 2024, *protocols.io*. https://dx.doi.org/10.17504/protocols.io.e6nvw1wj9lmk/v2

[12] W. Lavrijsen, A. Tudor, J. Müller, C. Iancu, and W. De Jong, "Classical optimizers for noisy intermediate-scale quantum devices," in *2020 IEEE Int. Conf. on Quantum Computing and Engineering* (*QCE*), 2020, pp. 267-277.