Quantum Information Processing (2024) 23:360
https://doi.org/10.1007/s11128-024-04573-w

Quantum search algorithm for binary constant weight codes

Kein Yukiyoshi[1] · **Naoki Ishikawa**[1]

Received: 25 June 2024 / Accepted: 10 October 2024

Abstract
A binary constant weight code is a type of error-correcting code with a wide range of applications. The problem of finding a binary constant weight code has long been studied as a combinatorial optimization problem in coding theory. In this paper, we propose a quantum search algorithm for binary constant weight codes. Specifically, the search problem is formulated as a polynomial binary optimization problem and Grover adaptive search is used for providing the quadratic speedup. Focusing on the inherent structure of the problem, we derive an upper bound on the minimum of the objective function value and a lower bound on the exact number of solutions. By exploiting these two bounds, we successfully reduced the constant overhead of the algorithm, although the overall query complexity remains exponential due to the NP-complete nature of the problem. In our algebraic analysis, it was found that this proposed algorithm is capable of reducing the number of required qubits, thus enhancing the feasibility. Additionally, our simulations demonstrated that it reduces the average number of classical iterations by 63% as well as the average number of total Grover rotations by 31%. The proposed approach may be useful for other quantum search algorithms and optimization problems.

Keywords Coding theory · Constant weight code · Quantum computing · Grover adaptive search

1 Introduction

A binary constant weight code is an error-correcting code, defined such that all codewords have the same Hamming weight [1]. The codes have a wide range of applications, including the design of very large-scale integration systems [2], the frequency assignment problem [3], the design of optical orthogonal codes [4], error-tolerant key generation from noisy data [5], inter-cell interference mitigation for flash memory [6], and the design of Grassmannian constellations [7]. Furthermore, although not pointed

✉ Naoki Ishikawa
ishikawa-naoki-fr@ynu.ac.jp

[1] Faculty of Engineering, Yokohama National University, Kanagawa 240-8501, Japan

Published online: 23 October 2024

out in the literature, binary constant weight codes have the exact same structure as index modulation [8] used in wireless communications and data storage studies. Using binary constant weight codes for codebook design is expected to achieve near-optimal performance due to their error-robust nature [9]. This codebook design is known as the index selection problem, which has been addressed in conventional studies [10–15] with some heuristic methods, and the problem itself is equivalent to the construction problem of binary constant weight codes .

A binary constant weight code $\mathcal{C}$ is composed of M different codewords, each of which is a binary row vector $\mathbf{c} = [c_0 \ \cdots \ c_{n-1}]$ with a length n and weight w. Let $d(\mathbf{c}_m, \mathbf{c}_{m'})$ be the Hamming distance between binary vectors $\mathbf{c}_m$ and $\mathbf{c}_{m'}$. Its minimum value $d = \min_{0 \le m < m' < M} d(\mathbf{c}_m, \mathbf{c}_{m'})$ is termed the minimum distance of $\mathcal{C}$. The maximum number of possible codewords is denoted by $A(n, d, w)$, which is an upper bound on M. A concern in the coding theory [16–20] is to clarify $A(n, d, w)$ or to construct a code that achieves $A(n, d, w)$. It is widely known that this problem can be equally attributed to the maximum clique problem, which has been shown to be NP-complete [21]. Specifically, consider binary codes of length n and weight w as nodes in a graph, and connect each node with an edge only if the Hamming distance between these two adjacent nodes is greater than or equal to d. Then, the size of the maximum clique in this graph achieves $A(n, d, w)$. Based on this structure, several studies have developed heuristics to construct better binary constant weight codes [22–24]. However, depending on the parameters (n, d, w), the optimization of binary constant weight codes becomes a challenging task due to its combinatorial explosion, even with the use of heuristics. Furthermore, proving that a code achieves $A(n, d, w)$ generally requires an exhaustive search over the search space.

In 1965, Gordon Moore predicted that the density of transistors on integrated circuits would double approximately every two years, a historical trend known as Moore's Law [25]. However, Moore's Law is anticipated to be approaching its end due to physical limitations, and the computational performance of classical computers is expected to plateau. In response, quantum computing is expected to overcome current limitations and replace classical computing, at least for a selected number of tasks [26]. To offer some examples, quantum annealing (QA) [27] and quantum approximate optimization algorithm (QAOA) [28] are well-known heuristic approaches based on quantum mechanics. Although these methods have already been demonstrated to work effectively on real devices, both approaches have proven unable to outperform classical computation under realistic assumptions [29]. Consequently, there is growing interest in quantum algorithms designed for fault-tolerant quantum computers (FTQC) [30].

Quantum computation over FTQC has proven to have advantages over classical computation in terms of query complexity required for solving specific problems, such as Shor's algorithm [31] and Grover's algorithm [32]. Grover's search algorithm finds a solution in an unordered database of N elements with query complexity $O(\sqrt{N})$. Boyer *et al.* extended Grover's algorithm to the case where the number of desired solutions is initially unknown [33], which is termed the Boyer-Brassard-Høyer-Tapp (BBHT) algorithm. Then, Dürr and Høyer proposed Grover adaptive search (GAS) [34–36], an extension of the BBHT algorithm to solve optimization problems. GAS can be interpreted as a kind of quantum exhaustive search algorithm, which guarantees the optimality of the solutions obtained.

In the line of studies on Grover's algorithm, an efficient construction method for a quantum circuit corresponding to any types of objective functions has long been unknown. The breakthrough is Gilliam's method proposed in [37] that efficiently constructs a quantum circuit of GAS for arbitrary binary polynomial objective functions. This method has already demonstrated quadratic speedups of some problems, such as maximum likelihood detection [38, 39] and wireless channel assignment problem [40, 41]. In GAS, its stop policy and the Grover iterations can be appropriately tuned for a fast convergence to the optimal solution [42], demonstrating improvements in the average numbers of classical iterations and total Grover rotations.

Against this background, in this paper, we formulate the problem of finding a binary constant weight code as a polynomial binary optimization problem. This formulation enables the use of GAS and provides a quadratic speedup for the search problem. Additionally, focusing on the inherent structure of the problem, our approach further accelerates GAS, which is a unique attempt in the literature. Solving the problem with polynomial complexity is considered impossible [43], even with quantum computers, because the problem is equivalent to the NP-complete maximum clique problem. However, we believe that providing a quadratic speedup and reducing the often-ignored constant overhead of the GAS algorithm is meaningful as a first attempt. The major contributions of this paper are summarized as follows:

1. The problem of finding a binary constant weight code is formulated as a polynomial binary optimization problem, and the number of qubits required for constructing a quantum circuit is maximally reduced. In this formulation, the Hamming distance between codewords, which is usually represented by exclusive OR (XOR), is attributed to the representation using an inner product. Furthermore, a unique trick using the inner product is incorporated into the objective function to guarantee that the minimum distance of an obtained code is at least a given value d.
2. Two mathematical properties are clarified: (1) an upper bound on the minimum of the objective function value and (2) a lower bound on the exact number of solutions. These clarifications help reduce the number of qubits and query complexity, which are supported by our algebraic analysis and numerical simulations.

The remainder of this paper is organized as follows: In Sect. 2, we review conventional quantum search algorithms such as BBHT and GAS. In Sect. 3, we present our problem formulation, and our algorithm based on GAS. In Sect. 4, the proposed GAS is compared with the conventional GAS, where both rely on the proposed formulation. Finally, in Sect. 5, we conclude this paper.

Italicized symbols represent scalar values, and bold symbols represent vectors and matrices. We use zero-based indexing. Table 1 summarizes the important mathematical symbols used in this paper.

2 Conventional quantum search algorithms

In this section, we introduce the conventional quantum search and optimization algorithms upon which our proposed algorithm is based.

Table 1 List of important mathematical symbols

$\{0, 1\}$		Finite field of order 2
$\mathbb{R}$		Real numbers
$\mathbb{C}$		Complex numbers
$\mathbb{Z}$		Integers
j	$\in \mathbb{C}$	Imaginary number
$(\cdot)^{\mathrm{H}}$		Hermitian transpose
$\overline{(\cdot)}$	$\in \mathbb{R}$	Upper bound
$\underline{(\cdot)}$	$\in \mathbb{R}$	Lower bound
$\mathcal{C}$		Code
$\mathbf{c}$	$\in \{0, 1\}^{1\times n}$	Codeword
$d(\mathbf{c}_m, \mathbf{c}_{m'})$	$\in \mathbb{Z}$	Hamming distance between $\mathbf{c}_m$ and $\mathbf{c}_{m'}$
d	$\in \mathbb{Z}$	Minimum distance of a code
n	$\in \mathbb{Z}$	Length of a codeword
w	$\in \mathbb{Z}$	Hamming weight
M	$\in \mathbb{Z}$	Number of codewords
x	$\in \{0, 1\}$	Binary variable
$E(\cdot)$	$\in \mathbb{Z}$	Objective function
ρ	$\in \mathbb{Z}$	Penalty coefficient of $E(\cdot)$
l	$\in \mathbb{Z}$	Exponential coefficient of $E(\cdot)$
q_1	$\in \mathbb{Z}$	Number of binary variables
q_2	$\in \mathbb{Z}$	Number of qubits required for $E(\cdot)$
y_i	$\in \mathbb{Z}$	Threshold for $E(\cdot)$ at i-th iteration
L_i	$\in \mathbb{Z}$	i-th number of Grover rotations
$\mathbf{P}(n, w)$	$\in \{0, 1\}^{\binom{n}{w}\times n}$	Combinatorial matrix [44]
$\mathbf{p}_r$	$\in \{0, 1\}^{1\times n}$	$(r + 1)$-th row vector of $\mathbf{P}(n, w)$

2.1 Grover's search and BBHT algorithm [33, 45]

Grover's search algorithm [32] finds one of t solutions in N elements. The query complexity is $O\left(\sqrt{N/t}\right)$, which is the total number of Grover operators applied to reach an optimal solution. Specifically, the probability of observing the desired solution is amplified by applying the Grover operator $\mathbf{G}$ to the uniform superposition state $\frac{1}{\sqrt{N}}\sum_i |i\rangle$. When the Grover operator is applied L times, the success probability is expressed as [45]

$$P_{\text{success}}(L) = \sin^2\left((2L+1)\arcsin\left(\sqrt{\frac{t}{N}}\right)\right). \tag{1}$$

Algorithm 1 BBHT Algorithm [33].

Input: $\lambda > 1$
Output: $\mathbf{x}$
1: Set $k_0 = 1$ and $i = 0$.
2: **while** Optimal solution not found **do**
3: Randomly select the rotation count L_i from the set $\{0, 1, ..., \lceil k_i - 1 \rceil\}$.
4: Evaluate $\mathbf{G}^{L_i}\mathbf{A}|0\rangle_{q_1}$ to obtain $\mathbf{x}$.
5: $k_{i+1} = \min\{\lambda k_i, \sqrt{2^{q_1}}\}$.
6: **end while**

According to (1), the number of Grover rotations that maximizes $P_{\text{success}}(L)$ can be expressed as [45]

$$L_{\text{opt}} = \left\lfloor \frac{\pi}{4}\sqrt{\frac{N}{t}} \right\rfloor. \tag{2}$$

Here, the number of solutions t must be known to obtain L_{opt}, but in practice, it cannot be obtained in advance. To cope with this challenge, Boyer *et al.* proposed the BBHT algorithm. As summarized in Algorithm 1, it searches for an appropriate L until the desired solution $\mathbf{x}$ is found. This repetition achieves the query complexity of $O\left(\sqrt{N/t}\right)$ even if the number of solutions is unknown. To be more specific, L is a random variable drawn from a uniform distribution $[0, k)$, where the parameter k is increased by $k_{i+1} = \min\left\{\lambda k_i, \sqrt{2^{q_1}}\right\}$ for each iteration. The parameter λ is a constant value related to the increase rate of k. In [36], Baritompa *et al.* demonstrated through numerical search that the expected value of the parameter λ that minimizes the query complexity is 1.34, while the optimal value depends on the distribution of the objective function values over the search space. They also found that when λ is set to 1.34, the expected total number of Grover operators is at most

$$1.32\sqrt{\frac{N}{t}}, \tag{3}$$

provided that the $\sqrt{N}$ ceiling of the parameter k, corresponding to line 5 in Algorithm 1, is ignored in the global optimization context, *i.e.*, $k \to \infty$.

2.2 GAS algorithm [34–37]

GAS is an algorithm that extends the BBHT algorithm to obtain the optimal solution for the polynomial binary optimization problems in the form

$$\begin{aligned} \min_{\mathbf{x}} \quad & E(\mathbf{x}) \\ \text{s.t.} \quad & x_i \in \{0, 1\} \quad (i = 0, \cdots, q_1 - 1). \end{aligned} \tag{4}$$

Here, $\mathbf{x} = (x_0, \cdots, x_{q_1-1})$ denotes binary variables and $E(\mathbf{x})$ denotes an arbitrary polynomial objective function. As with the BBHT algorithm, the query complexity

Algorithm 2 Conventional GAS [36, 37].

Input: $E : \{0,1\}^{q_1} \to \mathbb{Z}$, $\lambda = 1.34$
Output: $\mathbf{x}$
1: Uniformly sample $\mathbf{x}_0 \in \{0,1\}^{q_1}$ and set $y_0 = E(\mathbf{x}_0)$.
2: Set $k = 1$ and $i = 0$.
3: **repeat**
4: Randomly select the rotation count L_i from the set $\{0, 1, ..., \lceil k-1 \rceil\}$.
5: Evaluate $\mathbf{G}^{L_i}\mathbf{A}_{y_i}|0\rangle_{q_1+q_2}$ to obtain $\mathbf{x}$ and y.
6: **if** $y < y_i$ **then**
7: $\mathbf{x}_{i+1} = \mathbf{x}$, $y_{i+1} = y$, and $k = 1$.
8: **else**
9: $\mathbf{x}_{i+1} = \mathbf{x}_i$, $y_{i+1} = y_i$, and $k = \min\{\lambda k, \sqrt{2^{q_1}}\}$.
10: **end if**
11: $i = i + 1$.
12: **until** a termination condition is met.

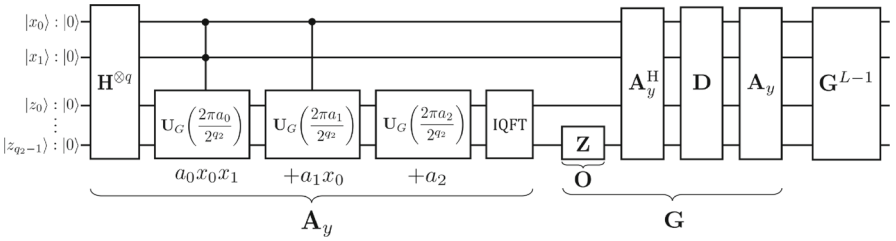


Fig. 1 Quantum circuit for GAS corresponding to an objective function $E(\mathbf{x}) - y = a_0x_0x_1 + a_1x_0 + a_2$, which is composed of $\mathbf{A}_y$, $\mathbf{D}$, $\mathbf{O}$, and $\mathbf{G} = \mathbf{A}_y\mathbf{D}\mathbf{A}_y^{\mathrm{H}}\mathbf{O}$

is $O\left(\sqrt{N/t}\right)$, where N is the search space size, *i.e.*, $N = 2^{q_1}$. As summarized in Algorithm 2, first, the initial value of the threshold $y_0 = E(\mathbf{x}_0)$ is determined by a random solution $\mathbf{x}_0$. Next, the BBHT algorithm is used to search for a better solution $\mathbf{x}$ such that the value of the objective function is less than the current threshold y_i, and the threshold is updated by $y_{i+1} = E(\mathbf{x})$. Thereby, the optimal solution is obtained through multiple trials.

Original GAS algorithm has assumed the existence of a black-box quantum oracle $\mathbf{O}$ that can identify the desired states. The breakthrough was Gilliam's method [37]. In [37], Gilliam *et al.* proposed an efficient construction method for a quantum circuit that corresponds to an objective function of binary optimization. Aided by two's complement representation of the objective function value, the quantum oracle $\mathbf{O}$ can be easily constructed with only a single quantum gate. As an explicit example, Fig. 1 shows a quantum circuit corresponding to the objective function $E(\mathbf{x}) - y = a_0x_0x_1 + a_1x_0 + a_2$, which has quantum gates that correspond to the polynomial terms $a_0x_0x_1$, a_1x_0, and a_2. Note that a_2 is a constant that includes the constant term of $E(\mathbf{x})$ and $-y$.

In the Gilliam's construction method, $q = q_1 + q_2$ qubits are required, where q_1 denotes the number of binary variables $(x_0, \cdots, x_{q_1})$, and q_2 denotes the number of qubits for encoding the objective function $E(\mathbf{x})$. Here, q_2 satisfies

$$-2^{q_2-1} \leq \min_{\mathbf{x}} E(\mathbf{x}) \leq \max_{\mathbf{x}} E(\mathbf{x}) < 2^{q_2-1} \tag{5}$$

due to the two's complement representation.

Next, we describe the procedure for constructing the quantum circuit for a binary optimization problem [37], where three types of unitary operators $\mathbf{A}_y$, $\mathbf{D}$, and $\mathbf{O}$ are constructed. The following steps 1–3 correspond to the construction of $\mathbf{A}_y$, while the step 4 corresponds to the Grover operator $\mathbf{G} = \mathbf{A}_y\mathbf{D}\mathbf{A}_y^{\mathrm{H}}\mathbf{O}$.

1. Hadamard gates $\mathbf{H}^{\otimes q}$ act on the initial state $|0\rangle_q$ to create a uniform superposition state, yielding the transition of [37]

$$\begin{aligned} |0\rangle_q \xrightarrow{\mathbf{H}^{\otimes q}} & \frac{1}{\sqrt{2^q}} \sum_{i=0}^{2^q-1} |i\rangle_q \\ = & \frac{1}{\sqrt{2^{q_1+q_2}}} \sum_{\mathbf{x}\in\{0,1\}^{q_1}} \sum_{i=0}^{2^{q_2}-1} |\mathbf{x}\rangle_{q_1} |i\rangle_{q_2}. \end{aligned} \tag{6}$$

   Using the tensor product $\otimes$, the Hadamard gates $\mathbf{H}^{\otimes q}$ are defined as

$$\mathbf{H}^{\otimes q} = \underbrace{\mathbf{H} \otimes \mathbf{H} \otimes \cdots \otimes \mathbf{H}}_{q} \tag{7}$$

   and

$$\mathbf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1. \end{bmatrix} \tag{8}$$

2. Each term of $E(\mathbf{x}) - y$ corresponds to a unitary gate $\mathbf{U}_G(\theta)$. It acts on the lower q_2 qubits to rotate the phase of quantum states. Let a be a coefficient of the term. Then, the phase shift is defined as $\theta = \frac{2\pi a}{2^{q_2}}$. That is, the phase advance represents addition, and the phase delay represents subtraction. The unitary gate is defined by

$$\mathbf{U}_G(\theta) = \underbrace{\mathbf{R}(2^{q_2-1}\theta) \otimes \cdots \otimes \mathbf{R}(2^0\theta)}_{q_2} \tag{9}$$

   and the phase gate

$$\mathbf{R}(\theta) = \begin{bmatrix} 1 & 0 \\ 0 & e^{j\theta}. \end{bmatrix} \tag{10}$$

   For a state $\mathbf{x}_0 \in \{0, 1\}^{q_1}$, $\theta' = 2\pi \left(E(\mathbf{x}_0) - y\right)/2^{q_2}$ and $\mathbf{U}_G(\theta')$ yield the transition of [37]

$$\frac{1}{\sqrt{2^{q_2}}} \sum_{i=0}^{2^{q_2}-1} |i\rangle_{q_2} \xrightarrow{\mathbf{U}_G(\theta')} \frac{1}{\sqrt{2^{q_2}}} \sum_{i=0}^{2^{q_2}-1} e^{ji\theta'} |i\rangle_{q_2}. \tag{11}$$

The interaction between binary variables is expressed by controlled $\mathbf{U}_G(\theta)$. A specific method for interacting multiple binary variables is exemplified in Fig. 1, where $\mathbf{U}_G(\theta)$ gates are controlled by the corresponding qubits.

3. Applying the inverse quantum Fourier transform (IQFT) [46] to the lower q_2 qubits yields the transition of [37]

$$\frac{1}{\sqrt{2^{q_2}}}\sum_{i=0}^{2^{q_2}-1} e^{ji\theta'}|i\rangle_{q_2} \xrightarrow{\text{IQFT}} \frac{1}{\sqrt{2^{q_2}}}|E(\mathbf{x}_0)-y\rangle_{q_2}. \tag{12}$$

In summary, the steps 1–3 are referred to as the state preparation operator $\mathbf{A}_y$. It encodes the value $E(\mathbf{x})-y$ into q_2 qubits for all the states. That is, it satisfies [37]

$$\mathbf{A}_y|0\rangle_q = \frac{1}{\sqrt{2^{q_1}}}\sum_{\mathbf{x}\in\{0,1\}^{q_1}} |\mathbf{x}\rangle_{q_1}|E(\mathbf{x})-y\rangle_{q_2}. \tag{13}$$

4. Finally, to amplify the states of interest, the Grover operator $\mathbf{G}=\mathbf{A}_y\mathbf{D}\mathbf{A}_y^{\mathrm{H}}\mathbf{O}$ acts L times, where $\mathbf{D}$ denotes the Grover diffusion operator [32]. The oracle $\mathbf{O}$ identifies the states of interest that satisfy $E(\mathbf{x})-y<0 \Leftrightarrow E(\mathbf{x})<y$. Since we use the two's complement representation, such states can be identified by a single Pauli-Z gate

$$\mathbf{Z}=\begin{bmatrix}1 & 0\\ 0 & -1,\end{bmatrix} \tag{14}$$

which is acted on the most significant qubit of q_2 qubits.

Note that an open-source implementation of the GAS algorithm is available from IBM Qiskit [47].

3 Proposed quantum search algorithm

In this section, we propose an objective function that attributes the problem of finding a binary constant weight code to a binary optimization problem. The problem is defined as finding a code such that the code length n, weight w, number of codewords M, and minimum distance d are each equal to the given values (n, w, M, d), which is hereinafter referred to as condition (n, w, M, d). Additionally, focusing on the inherent properties of the problem, we derive an upper bound on the minimum value of the objective function and a lower bound on the number of solutions. Using these bounds, we modify GAS and reduce the number of qubits and query complexity.

3.1 Proposed formulation

To obtain the optimal solution for a binary constant weight code $A(n, d, w)$, we formulate the search problem as a polynomial binary optimization problem, which is

supported by GAS. A challenging task here is that the objective function has to incorporate the Hamming distance between codewords. In general, the Hamming distance $d(\mathbf{c}_m, \mathbf{c}_{m'})$ between codewords $\mathbf{c}_m$ and $\mathbf{c}_{m'}$ is expressed using the XOR operation $\oplus$, which is not representable by a polynomial function. As long as the Hamming weight is constant, the XOR operation can be expressed by an inner product $\langle \mathbf{c}_m, \mathbf{c}_{m'} \rangle = \sum_{i=1}^{n} c_{m,i} c_{m',i}$, where $c_{m,i}$ is the i-th element of $\mathbf{c}_m$.

Theorem 1 *If a code has a constant weight w, then, for any pair of codewords $(\mathbf{c}_m, \mathbf{c}_{m'})$, we have*

$$d(\mathbf{c}_m, \mathbf{c}_{m'}) = 2\left(w - \langle \mathbf{c}_m, \mathbf{c}'_m \rangle\right). \tag{15}$$

Proof For any pair of i-th elements of codewords $(c_{m,i}, c_{m',i}) = (0, 0), (0, 1), (1, 0), (1, 1)$, the following equation holds:

$$c_{m,i} \oplus c_{m',i} = \left(c_{m,i} + c_{m',i}\right) - 2c_{m,i}c_{m',i}. \tag{16}$$

Since weights of codewords are constant, w, by summing (16) over index $i = 1, \cdots, n$, we have

$$\begin{aligned} \sum_{i=1}^{n} \left(c_{m,i} \oplus c_{m',i}\right) &= d(\mathbf{c}_m, \mathbf{c}_{m'}) \\ &= \sum_{i=1}^{n} \left(c_{m,i} + c_{m',i}\right) - 2\sum_{i=1}^{n} c_{m,i}c_{m',i} \\ &= 2\left(w - \langle \mathbf{c}_m, \mathbf{c}_{m'} \rangle\right) \end{aligned} \tag{17}$$

□

Theorem 1 allows us to transform the minimum distance of a code $d(\mathcal{C})$ into

$$\begin{aligned} d(\mathcal{C}) &= \min_{0 \le m < m' < M} d(\mathbf{c}_m, \mathbf{c}_{m'}) \\ &= \min_{0 \le m < m' < M} 2\left(w - \langle \mathbf{c}_m, \mathbf{c}_{m'} \rangle\right) \\ &= 2\left(w - \max_{0 \le m < m' < M} \langle \mathbf{c}_m, \mathbf{c}_{m'} \rangle\right). \end{aligned} \tag{18}$$

Here, the following Theorem holds.

Theorem 2 *The minimum distance of a code $\mathcal{C}^*$ is at least d, where we have*

$$\mathcal{C}^* = \arg\min_{C} s(\mathcal{C}), \tag{19}$$

$$s(\mathcal{C}) = \sum_{0 \le m < m' < M} \langle \mathbf{c}_m, \mathbf{c}_{m'} \rangle^l, \quad \text{and} \tag{20}$$

$$l = \left\lfloor \frac{\log \binom{M}{2}}{\log\left(1 + \frac{2}{2w-d}\right)} + 1 \right\rfloor, \tag{21}$$

which is valid for the $d \neq 2w$ case.[1]

Proof From (15), for any pair (m, m') of a code such that the minimum distance is at least d, the following inequality holds:

$$\langle \mathbf{c}_m, \mathbf{c}_{m'} \rangle \leq w - \frac{d}{2}. \tag{22}$$

From this relationship, the upper bound on $s(\mathcal{C})$ is given by

$$s(\mathcal{C}) \leq \overline{s}(\mathcal{C}) = \binom{M}{2} \left(w - \frac{d}{2}\right)^l, \tag{23}$$

where l is an arbitrary positive coefficient. Conversely, for a code such that the minimum distance is less than d, there is at least one pair (m, m') that satisfies

$$\langle \mathbf{c}_m, \mathbf{c}_{m'} \rangle \geq w - \frac{d-2}{2} = w - \frac{d}{2} + 1. \tag{24}$$

From this relationship, the lower bound on $s(\mathcal{C})$ is given by

$$s(\mathcal{C}) \geq \underline{s}(\mathcal{C}) = \left(w - \frac{d}{2} + 1\right)^l. \tag{25}$$

By comparing the obtained upper and lower bounds, if l is sufficiently large, the following inequality holds:

$$\binom{M}{2} \left(w - \frac{d}{2}\right)^l < \left(w - \frac{d}{2} + 1\right)^l. \tag{26}$$

Thus, if l is sufficiently large, (19) is minimized, and the minimum distance of the obtained code becomes at least d. Taking a logarithm of both sides, the condition for l is given by

$$l > \frac{\log \binom{M}{2}}{\log\left(1 + \frac{2}{2w-d}\right)}. \tag{27}$$

□

[1] If $d = 2w$, according to (18), an inner product of any pair of codewords is zero. Thus, the solution is obvious, and the maximum number of codewords is $\lfloor \frac{n}{w} \rfloor$.

Based on the above relationships, we formulate the problem of finding a binary constant weight code as a polynomial binary optimization problem, while maximally reducing the number of binary variables. We propose an objective function of

$$E(\mathbf{x}) = f(\mathbf{x}) + \rho \cdot g(\mathbf{x}), \tag{28}$$

where

$$f(\mathbf{x}) = \sum_{0 \le r < r' < \binom{n}{w}} \langle \mathbf{p}_r, \mathbf{p}_{r'} \rangle^l \cdot x_r x_{r'} \tag{29}$$

and

$$g(\mathbf{x}) = \left(\sum_{r=0}^{\binom{n}{w}-1} x_r - M \right)^2. \tag{30}$$

Here, for index $r = 0, \cdots, \binom{n}{w} - 1$, $\mathbf{p}_r$ denotes the $(r+1)$-th row vector of the combinatorial matrix $\mathbf{P}(n, w)$, which represents $\binom{n}{w}$ combinations [44]

$$\mathbf{P}(n, w) = \begin{bmatrix} \mathbf{1} & \mathbf{P}(n-1, w-1) \\ \mathbf{0} & \mathbf{P}(n-1, w) \end{bmatrix} \in \{0, 1\}^{\binom{n}{w} \times n}. \tag{31}$$

In (31), $\mathbf{1}$ denotes a one column vector of length $\binom{n-1}{w-1}$ and $\mathbf{0}$ denotes a zero column vector of length $\binom{n-1}{w}$. In (28), the binary variable x_r indicates whether the $(r+1)$-th row of the combinatorial matrix is used as a codeword or not. That is, the relationship between $\mathcal{C}$ and x_r is expressed by $\mathcal{C} = \{\mathbf{p}_r | x_r = 1\}$. The penalty function $g(\mathbf{x})$ ensures that only M from the $\binom{n}{w}$ combinations is used and has priority over the cost function $f(\mathbf{x})$. Thus, the penalty coefficient ρ can be defined as

$$\rho = \overline{f} + 1 = \binom{M}{2} (w-1)^l + 1, \tag{32}$$

where $\overline{f}$ denotes the upper bound of f.

3.2 Reduction of qubits

Since the number of qubits required to construct a quantum circuit determines feasibility, it should be maximally reduced. Here, we modify the objective function (28) so that a single codeword is always selected, leading to a reduction in the number of qubits.

Theorem 3 *If a code that satisfies the condition* (n, w, M, d) *exists, then for every possible codeword* $\mathbf{c}$ *in the combinatorial matrix, at least one code exists that satisfies the condition* (n, w, M, d) *such that the codeword* $\mathbf{c}$ *is included.*

Proof When we consider a code as a matrix where each row vector is a codeword, the Hamming distance between codewords does not change even if we swap any two columns of the matrix. Therefore, by swapping the columns of any code that satisfies the condition (n, w, M, d), we can construct a code that contains any specific codeword and satisfies the condition (n, w, M, d). □

We define the matrix $\mathbf{P}'(n, w)$ by removing all row vectors from $\mathbf{P}(n, w)$ whose Hamming distance from the first row vector $\mathbf{p}_0$ is less than d. Then, the objective function with reduced binary variables can be defined as

$$E'(\mathbf{x}) = f'(\mathbf{x}) + \rho' \cdot g'(\mathbf{x}), \tag{33}$$

where

$$f'(\mathbf{x}) = \sum_{0 \le r < r' < q_1} \langle \mathbf{p}'_r, \mathbf{p}'_{r'} \rangle^l \cdot x_r x_{r'}, \tag{34}$$

$$g'(\mathbf{x}) = \left(\sum_{r=0}^{q_1 - 1} x_r - (M-1) \right)^2, \tag{35}$$

and $\mathbf{p}'_r$ is the $(r+1)$-th row vector of $\mathbf{P}'(n, w)$. The number of binary variables q_1 is equal to the number of rows in $\mathbf{P}'(n, w)$ and is expressed as

$$q_1 = \sum_{i=0}^{w - \frac{d}{2}} \binom{w}{i} \binom{n-w}{w-i}. \tag{36}$$

The penalty coefficient ρ' can be defined as

$$\rho' = \overline{f}' + 1 = \binom{q_1}{2} (w-1)^l + 1 \tag{37}$$

where $\overline{f}'$ denotes the upper bound of f'. Let $E'_{\max}$ be the maximum of the objective function value, *i.e.*, $E'_{\max} = \max_{\mathbf{x}} E'(\mathbf{x})$. It is upper bounded by

$$\begin{aligned} E'_{\max} &\le \overline{E}'_{\max} = \overline{f}' + \rho' \overline{g}' \\ &= \begin{cases} \binom{q_1}{2}(w-1)^l + \rho' (q_1 - M + 1)^2 & (2(M-1) < q_1) \\ \binom{q_1}{2}(w-1)^l + \rho' (M-1)^2 & (2(M-1) \ge q_1) \end{cases} \\ &= \begin{cases} O\left(q_1^2 w^l (q_1 - M)^2\right) & (2(M-1) < q_1) \\ O\left(q_1^2 w^l M^2\right) & (2(M-1) \ge q_1). \end{cases} \end{aligned} \tag{38}$$

Then, the number of qubits required to encode the objective function value is expressed by

$$q'_2 = \lceil \log_2(\overline{E}'_{\max}) \rceil + 1$$

$$= \begin{cases} O\left(\log(q_1) + l\log(w)\right) & (2(M-1) < q_1) \\ O\left(\log(q_1) + l\log(w) + \log(M)\right) & (2(M-1) \geq q_1). \end{cases} \tag{39}$$

Assuming $w \neq 2d$, the coefficient l is upper bounded by

$$\begin{aligned} l &= \left\lfloor \frac{\log\binom{M}{2}}{\log\left(1 + \frac{2}{2w-d}\right)} + 1 \right\rfloor \leq \left\lfloor \frac{\log\binom{M}{2}}{2} + 1 \right\rfloor \\ &= O(\log(M)) \end{aligned} \tag{40}$$

Using (40), q_2' of (39) can be simplified to

$$q_2' = O\left(\log(q_1) + \log(M)\log(w)\right). \tag{41}$$

The total number of qubits is

$$q_1 + q_2' = O\left(q_1 + \log(M)\log(w)\right). \tag{42}$$

The query complexity is expressed by

$$O\left(\sqrt{\frac{2^{q_1}}{t}}\right), \tag{43}$$

where t is the exact number of solutions and is analyzed later.

3.3 Further speedup for Grover adaptive search

3.3.1 Novel initial threshold

Let $E'_{\min}$ be the minimum of the objective function value, *i.e.*, $E'_{\min} = \min_{\mathbf{x}} E'(\mathbf{x})$. Similar to (23), its upper bound can be derived as

$$E'_{\min} < \overline{E}'_{\min} + 1 = \binom{M-1}{2}\left(w - \frac{d}{2}\right)^l + 1. \tag{44}$$

The objective function value $E'(\mathbf{x})$ may become greater than $\overline{E}'_{\min}$. By contrast, if it is smaller than $\overline{E}'_{\min}$, the obtained code has at least the minimum distance d. Therefore, the desired code can be obtained without updating the threshold of GAS. That is, we set the initial value of the threshold to

$$y_0 = \overline{E}'_{\min} + 1. \tag{45}$$

Additionally, if the objective function value is greater than $\overline{E}'_{\min}$, the obtained code is not good regarding the minimum distance. This fact helps to reduce the number of

qubits q_2. Specifically, the penalty coefficient ρ' of $E'(\mathbf{x})$ given in (33) is replaced by

$$\rho'' = \overline{E}'_{\min} + 1. \tag{46}$$

and we have an updated objective function

$$E''(\mathbf{x}) = f'(\mathbf{x}) + \rho'' \cdot g'(\mathbf{x}) \tag{47}$$

where f' and g' are defined by (34) and (35), respectively. Then, the objective function is upper bounded by

$$\begin{aligned} E''_{\max} &\leq \overline{E}''_{\max} \\ &= \begin{cases} O\left(q_1^2 w^l + M^2 (w - d/2)^l (q_1 - M)^2\right) & (2(M-1) < q_1) \\ O\left(q_1^2 w^l + M^4 (w - d/2)^l\right) & (2(M-1) \geq q_1). \end{cases} \end{aligned} \tag{48}$$

As with the $E'_{\max}$ case, the number of qubits required to encode the objective function value can be simplified to

$$\begin{aligned} q''_2 &= \lceil \log_2(\overline{E}''_{\max}) \rceil + 1 \\ &= O\left(\log(q_1) + \log(M)\log(w)\right). \end{aligned} \tag{49}$$

The number of qubits that can be reduced is bounded by

$$\begin{aligned} &\lceil \log_2(\overline{E}'(\mathbf{x})) \rceil - \lceil \log_2(\overline{E}''(\mathbf{x})) \rceil \\ &> \log_2\left(\frac{\overline{E}'(\mathbf{x})}{\overline{E}''(\mathbf{x})}\right) - 2 = \log_2\left(\frac{\overline{f}' + \rho'\overline{g}' - 1}{\overline{f}' + \rho''\overline{g}' - 1}\right) - 2 \\ &> \log_2\left(\frac{\overline{f}' + \rho'\overline{g}'}{\overline{f}' + \rho''\overline{g}'}\right) - 2 = \log_2\left(\frac{1 + \overline{g}'}{1 + \frac{\rho''}{\rho'}\overline{g}'}\right) - 2 \\ &= \log_2\left(\frac{1 + \overline{g}'}{1 + \frac{(M-1)(M-2)(2w-d)^l + 4}{q_1(q_1-1)(2w-2)^l + 4}\overline{g}'}\right) - 2, \end{aligned} \tag{50}$$

which is obviously positive since $\rho' > \rho''$ when $q_1 > M$ and $d > 2$. Note that, for constructing a code that satisfies the condition (n, w, M, d), it is sufficient to obtain just one code whose objective function is smaller than $\overline{E}'_{\min}$.

3.3.2 Novel limit on the number of Grover rotations

The BBHT search and GAS algorithms try to find the appropriate number of Grover rotations, which increases the query complexity compared to the case where the number of solutions is previously known. Although, in general, the exact number

of solutions t is unknown, the possible range of t may be obtained from the inherent structure of the problem. Our proposed algorithm exploits this information.

Theorem 4 *If $w \leq d$ and $A(n-1, d, w) < M-1$, then the exact number of solutions t is lower bounded by*

$$t \geq \underline{t} = \begin{cases} w! & (w - \frac{d}{2} = 1) \\ \min\limits_{2 \leq i \leq w-d/2} \binom{w}{i} & (w - \frac{d}{2} \geq 2). \end{cases} \tag{51}$$

The proof is given in the Appendix A.

We improve the quantum search algorithm using the obtained lower bound $\underline{t}$ of (51). Traditionally, the number of Grover rotations L is uniformly drawn from the semi-open interval $[0, k)$, where k is a parameter that increases in each iteration of GAS. In the BBHT search algorithm [33], the maximum value of k is set to

$$\overline{k}_{\mathrm{BBHT}} = \sqrt{N} = \sqrt{2^{q_1}}, \tag{52}$$

which is defined on the basis of $L_{\mathrm{opt}} = \lfloor \pi/4\sqrt{N/t} \rfloor$ and the query complexity $O(\sqrt{N})$. The same value is used in the original GAS implementation of IBM Qiskit [47].

Although $\overline{k}_{\mathrm{BBHT}} = \sqrt{N}$ is commonly used, its exact mathematical justification has not been provided, and the true optimal value is still unknown. In the following, we clarify the optimal maximum value of k, denoted by $\overline{k}_{\mathrm{opt}}$, that minimizes the query complexity. The probability of success $P_k(t)$ is known as [33]

$$P_k(t) = \frac{1}{2} - \frac{\sin(4k\theta)}{4k\sin(2\theta)} \tag{53}$$

where

$$\theta = \arcsin\left(\sqrt{\frac{t}{2^{q_1}}}\right). \tag{54}$$

Then, the optimal value k_{opt} that minimize query complexity is

$$\begin{aligned} k_{\mathrm{opt}} &= \arg\min_k \frac{k}{P_k(t)} \\ &= \arg\min_k \frac{4k^2\sin(2\theta)}{2k\sin(2\theta) - \sin(4k\theta)} \end{aligned} \tag{55}$$

and is upper bounded by

$$k_{\mathrm{opt}} \leq \overline{k}_{\mathrm{opt}} = \arg\min_k \frac{k}{P_k(\underline{t})}. \tag{56}$$

Theorem 5 *If the ratio $t/2^{q_1}$ is sufficiently small, then*

$$\overline{k}_{\mathrm{opt}} \in \left[1, \lceil \frac{1+\sqrt{2}}{2} \sqrt{\frac{2^{q_1}}{\underline{t}}} \rceil\right]. \tag{57}$$

The proof is given in the Appendix B. In general, $t/2^{q_1}$ is sufficiently small that it can be approximated as $(t/2^{q_1})^2 \sim 0$. The search range of $\overline{k}_{\mathrm{opt}}$ can be limited within the closed interval using Theorem 5. Then, we can search for $\overline{k}_{\mathrm{opt}}$ with negligible complexity; however, $\overline{k}_{\mathrm{opt}}$ itself is not expressed in a closed form. For example, within the closed interval, root-finding algorithm such as the bisection method [48] or Newton–Raphson method [49] is used for the first-order derivative of $k/P_k(\underline{t})$, all the k that take the extreme values are listed, and $\overline{k}_{\mathrm{opt}}$ is one of the listed values or one of both ends of the closed interval. Note that, the lower bound obtained in Theorem 4 requires assumption $w \leq d$ and $A(n-1, d, w) < M - 1$. Thus, this extra technique cannot be applied to all possible parameters of the problem. Since there has not been much research on the number of solutions for constant weight codes, more general and stronger bounds may be found in the future. Moreover, Theorem 4 also holds for k_{opt} by replacing $\underline{t}$ with t. This result suggests that the value of k need not be larger than $(1+\sqrt{2})/2 \cdot \sqrt{2^{q_1}} \sim 1.207 \cdot \sqrt{2^{q_1}}$, even if the number of solutions is completely unknown.

The query complexity can be reduced by setting $\overline{k}_{\mathrm{opt}}$ as the upper bound on k. Note that, although this paper only focuses on the lower bound, an upper bound on t may be obtained for some problems, and a similar approach could further accelerate the quantum search algorithm more.

Overall, the proposed GAS is summarized in Algorithm 3. The conventional GAS is initiated with a random solution $\mathbf{x}_0$ and a random threshold $y_0 = E''(\mathbf{x}_0)$. In the proposed GAS, a strict threshold is set from the beginning, using the upper bound on the minimum of the objective function value, $\overline{E}'_{\mathrm{min}} + 1$ of (45). Additionally, instead of the conventional upper bound $k \leq \sqrt{2^{q_1}}$, we use the stricter bound, $\overline{k}_{\mathrm{opt}}$ of (56). These modifications further accelerate the efficient GAS algorithm. Note that, in [36], the probability of a success is maximized with $\lambda = 1.44$ if the number of solutions is sufficiently smaller than the search space size (about 0.2% in [36]). In general, if the initial threshold is as strict as $\overline{E}'_{\mathrm{min}}$, the number of solutions becomes much smaller than the search space size. Thus, we set $\lambda = 1.44$ in Algorithm 3 following [36].

3.4 Number of quantum gates

We analyze the number of quantum gates that determines the size of a quantum circuit, which closely relates to the feasibility of the proposed algorithm. In the GAS circuit, the number of gates required for $\mathbf{A}_{y_i}$ has a dominant impact, which corresponds to the state preparation operator. Thus, we only focus on $\mathbf{A}_{y_i}$ later.

Algorithm 3 Proposed GAS.

Input: $E'' : \{0, 1\}^n \to \mathbb{Z}$, $\overline{E}_{\min}, \overline{k}_{\text{opt}}, \lambda = 1.44$
Output: **x**
1: Set $y_0 = \overline{E}_{\min}$, $k = 1$, and $i = 0$.
2: **repeat**
3: Randomly select the rotation count L_i from the set $\{0, ..., \lceil k - 1 \rceil\}$.
4: Evaluate $\mathbf{G}^{L_i}\mathbf{A}_{y_i}|0\rangle_{n+m}$ to obtain **x** and y.
5: **if** $y < y_i$ **then**
6: $\mathbf{x}_{i+1} = \mathbf{x}$, $y_{i+1} = y$, and $k = 1$.
7: **else**
8: $\mathbf{x}_{i+1} = \mathbf{x}_i$, $y_{i+1} = y_i$, and $k = \min\{\lambda k, \overline{k}_{\text{opt}}\}$.
9: **end if**
10: $i = i + 1$.
11: **until** a termination condition is met.

As described in the first step of Sect. 2.2, the Hadamard gate **H** acts once on every qubit. Thus, using (42), the number of **H** gates involved in (6) is

$$G_{\text{H}} = q_1 + q_2 = O\left(q_1 + \log(M)\log(w)\right). \tag{58}$$

In the second step, the phase gate **R** is used to represent the coefficients in the objective function. Using (39), the number of **R** gates involved in (11) is

$$G_{\text{R}} = q_2 = O\left(\log(q_1) + \log(M)\log(w)\right). \tag{59}$$

The number of 1-controlled phase (1-CR) gates is the same as that of first-order terms in the objective function and is expressed as

$$G_{\text{1-CR}} = q_1 \cdot q_2 = O\left(q_1\left(\log(q_1) + \log(M)\log(w)\right)\right). \tag{60}$$

Similarly, the number of 2-controlled phase (2-CR) gates is the same as that of second-order terms and is expressed by

$$G_{\text{2-CR}} = \frac{q_1(q_1 - 1)}{2} \cdot q_2 = O\left(q_1^2\left(\log(q_1) + \log(M)\log(w)\right)\right). \tag{61}$$

Based on the above analysis, the number of gates required to implement the state preparation operator $\mathbf{A}_{y_i}$ is summarized in Table 2.

3.5 Specific example

Let us consider a specific example $(n, w, d, M) = (7, 3, 4, 7)$ of the proposed GAS using the objective function $E''(\mathbf{x})$ given in (47). In this case, the number of qubits is calculated as $q = q_1 + q_2'' = 22 + 15 = 37$ from (36) and (49). The objective function is $E''(\mathbf{x}) = \mathbf{x}^{\text{T}}\mathbf{Q}\mathbf{x} + 576$, where the matrix $\mathbf{Q} \in \mathbb{Z}^{q_1 \times q_1}$ is given in (65), and Fig. 2 shows the corresponding quantum circuit. Based on the combinatorial matrix $\mathbf{P}(7, 3)$,

Table 2 Number of quantum gates required by $\mathbf{A}_{y_i}$

Gate	Number of gates	
H	$q_1 + q_2$	$= O(q_1 + \log M \log w)$
R	q_2	$= O(\log q_1 + \log M \log w)$
1-CR	$q_1 \cdot q_2$	$= O\left(q_1 \left(\log q_1 + \log M \log w\right)\right)$
2-CR	$q_1(q_1 - 1) \cdot q_2/2$	$= O\left(q_1^2 \left(\log q_1 + \log M \log w\right)\right)$
IQFT	1	$= O(1)$

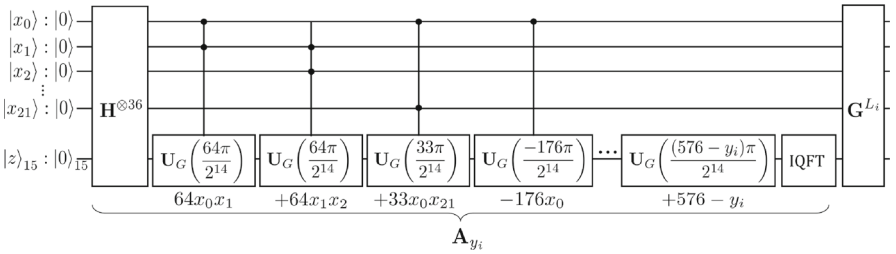


Fig. 2 Quantum circuit corresponding to $E''(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + 576 - y_i$

the reduced matrix $\mathbf{P}'(7, 3) \in \{0, 1\}^{q_1 \times n}$ is generated by removing row vectors whose Hamming distance to $\mathbf{p}_0 = [1\ 1\ 1\ 0\ 0\ 0\ 0]$ is less than $d = 4$ as follows:

$$\mathbf{P}'(7,3) = \begin{bmatrix} 1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,0\,0\,0 \\ 1\,1\,1\,0\,0\,0\,1\,1\,1\,0\,0\,0\,1\,1\,1\,0\,0\,0\,1\,1\,1\,0 \\ 1\,0\,0\,1\,1\,0\,1\,0\,0\,1\,1\,0\,1\,0\,0\,1\,1\,0\,1\,1\,0\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,1\,0\,1\,1 \\ 0\,0\,1\,0\,1\,1\,0\,0\,1\,0\,1\,1\,0\,0\,1\,0\,1\,1\,0\,1\,1\,1 \end{bmatrix}^{\mathrm{T}}. \tag{62}$$

The proposed GAS outputs one of the optimal solutions

$$\mathbf{x}_{\mathrm{opt}} = \left[1\,0\,0\,0\,0\,1\,0\,1\,0\,0\,1\,0\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\right]^{\mathrm{T}}, \tag{63}$$

which indicates that the optimal code consists of the first, sixth, eighth, 11th, 15th, and 16th rows of $\mathbf{P}'(7, 3)$ should be used, in addition to $\mathbf{p}_0 = [1\ 1\ 1\ 0\ 0\ 0\ 0]$. Finally,

the optimal code corresponding to (63) in a matrix format is

$$\mathcal{C}_{\text{opt}} = \begin{bmatrix} 1\,1\,1\,0\,0\,0\,0 \\ 1\,0\,0\,1\,1\,0\,0 \\ 1\,0\,0\,0\,0\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0 \\ 0\,1\,0\,0\,1\,0\,1 \\ 0\,0\,1\,1\,0\,0\,1 \\ 0\,0\,1\,0\,1\,1\,0 \end{bmatrix}. \tag{64}$$

$$\mathbf{Q} = \begin{bmatrix}
-176 & 64 & 64 & 64 & 64 & 33 & 64 & 33 & 33 & 33 & 33 & 32 & 64 & 33 & 33 & 33 & 33 & 32 & 64 & 64 & 33 & 33 \\
0 & -176 & 64 & 64 & 33 & 64 & 33 & 64 & 33 & 33 & 32 & 33 & 33 & 64 & 33 & 33 & 32 & 33 & 64 & 33 & 64 & 33 \\
0 & 0 & -176 & 33 & 64 & 64 & 33 & 33 & 64 & 32 & 33 & 33 & 33 & 33 & 64 & 32 & 33 & 33 & 33 & 64 & 64 & 33 \\
0 & 0 & 0 & -176 & 64 & 64 & 33 & 33 & 32 & 64 & 33 & 33 & 33 & 33 & 32 & 64 & 33 & 33 & 64 & 33 & 33 & 64 \\
0 & 0 & 0 & 0 & -176 & 64 & 33 & 32 & 33 & 33 & 64 & 33 & 33 & 32 & 33 & 33 & 64 & 33 & 33 & 64 & 33 & 64 \\
0 & 0 & 0 & 0 & 0 & -176 & 32 & 33 & 33 & 33 & 33 & 64 & 32 & 33 & 33 & 33 & 33 & 64 & 33 & 33 & 64 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 64 & 64 & 33 & 64 & 33 & 33 & 33 & 33 & 33 & 32 & 64 & 64 & 33 & 33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 64 & 33 & 64 & 33 & 64 & 33 & 33 & 32 & 33 & 64 & 33 & 64 & 33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 33 & 64 & 64 & 33 & 33 & 64 & 32 & 33 & 33 & 33 & 64 & 64 & 33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 64 & 33 & 33 & 32 & 64 & 33 & 33 & 64 & 33 & 33 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 33 & 32 & 33 & 33 & 64 & 33 & 33 & 64 & 33 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 32 & 33 & 33 & 33 & 33 & 64 & 33 & 33 & 64 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 64 & 64 & 64 & 33 & 64 & 64 & 33 & 33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 64 & 33 & 64 & 64 & 33 & 64 & 33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 33 & 64 & 64 & 33 & 64 & 64 & 33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 64 & 64 & 33 & 33 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 33 & 64 & 33 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 33 & 33 & 64 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 64 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176 & 64 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -176
\end{bmatrix} \tag{65}$$

4 Performance comparisons

In this section, we compare the proposed GAS with the conventional GAS in terms of query complexity and demonstrate that it converges to the optimal solutions faster.

Evaluation metrics In the following, we use two metrics: (1) the total number of classical iterations required to obtain the optimal solution, which is equal to the number of measurements required, and (2) query complexity, which is equal to the total number of Grover rotations $\sum_i L_i$. Both metrics are the same as those used in [42]. In [50], the former is called 'query complexity in the classical domain' and the latter is called 'query complexity in the quantum domain.' Typically, the query complexity in the quantum domain is considered as an important metric for evaluating quantum algorithms, corresponding to the time complexity in classical algorithms.

Simulation parameters All the parameters are the same as those used in Sect. 3.5, where we had $(n, w, d, M) = (7, 3, 4, 7)$. The objective function values were averaged over 10^6 trials. The termination condition of GAS can be defined by, for example, the number of iterations, time, or threshold value. Since the simulation was performed as a benchmark in this paper, we defined the termination condition that the threshold

value y_i reaches the minimum objective function value. We assume the realization of FTQC.

Considered schemes The classic exhaustive search is considered as a reference scheme since the concepts of computational complexity in classical computing and query complexity in quantum computing are completely different metrics. The original search space size is calculated as $\binom{\binom{n}{w}}{M}$. Using Theorem 3 and $\mathbf{P}'(n, w)$, the search space size can be readily reduced to $\binom{q_1}{M-1}$. Here, q_1, defined in (36), is always less than $\binom{n}{w}$. The first codeword is fixed to $\mathbf{p}_0$, and the remaining $M-1$ codewords are selected from q_1 candidates. The conventional GAS (Algorithm 2) is considered as a performance baseline. The proposed GAS (Algorithm 3) relies on the derived bounds, which contribute to increasing the speedup. Both classical exhaustive search and conventional GAS used the proposed objective function $E'(\mathbf{x})$ of (33), while the proposed GAS used the objective function $E''(\mathbf{x})$ of (47).

For example, in the $(n, w, d, M) = (7, 3, 4, 7)$ case, the original search space size is

$$\binom{\binom{n}{w}}{M} = \binom{\binom{7}{3}}{7} = \binom{35}{7} = 6724520. \tag{66}$$

Here, we use Theorem 3 and $\mathbf{P}'(n, w)$. Since we have $q_1 = 22$ from (36), the search space size is significantly reduced to

$$\binom{q_1}{M-1} = \binom{22}{7-1} = 74613. \tag{67}$$

The proposed formulation for GAS yields a search space size of

$$2^{q_1} = 2^{22} = 4194304, \tag{68}$$

and the expected minimum query complexity is

$$\sqrt{\frac{2^{q_1}}{\underline{t}}} = \sqrt{\frac{4194304}{6}} \sim 836. \tag{69}$$

Figure 3 shows the expected query complexity $k/P_k(\underline{t})$ in (56), over the interval obtained from Theorem 5. As shown in Fig. 3, when $\overline{k}_{opt} \sim 656$, the expected query complexity is minimized, which is utilized in the proposed GAS. Note that all the following results, including conventional GAS, proposed GAS, and the classical exhaustive search, are simulated over the reduced search space of (67).

First, Fig. 4 shows the relationship between each evaluation metric and the average of the objective function values, where the total numbers of classical iterations and Grover rotations were evaluated. As shown in Fig. 4, the objective function values for the conventional and proposed GAS differed significantly. By suppressing the objective function value, the number of qubits was reduced from $q_2' = 22$ to $q_2'' = 15$.

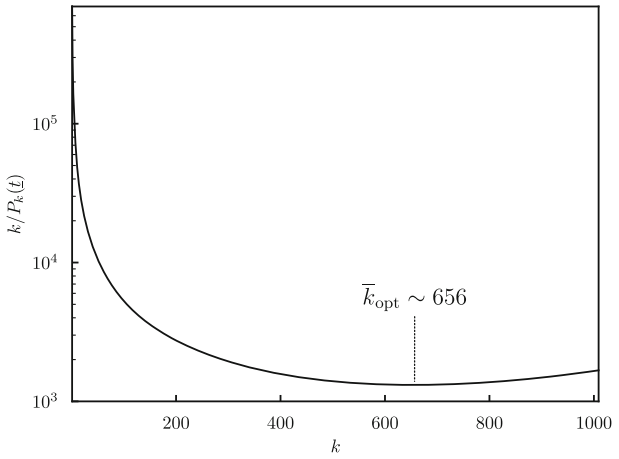
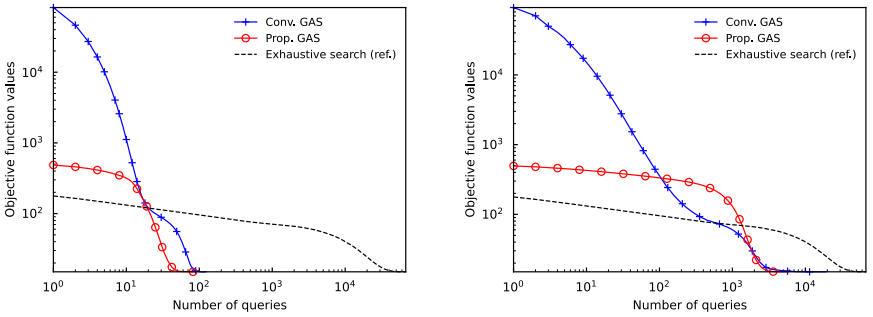


Fig. 3 Plot of the expected query complexity $k/P_k(\underline{t})$ in (56), over the interval obtained from Theorem 5



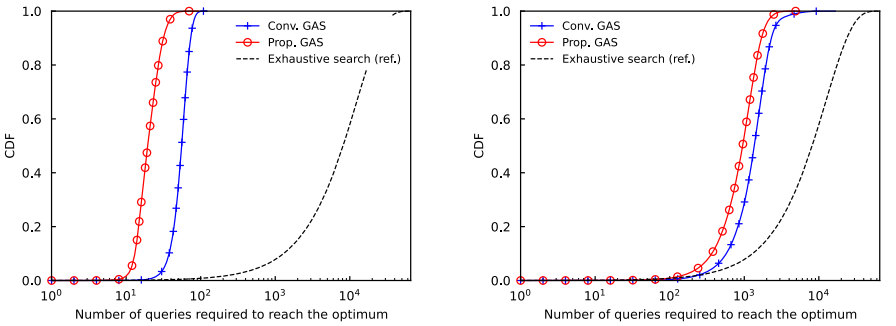
(a) The total number of classical iterations. (b) The total number of Grover rotations.

Fig. 4 Relationship between each metric and the average of the objective function values, where we had $(n, w, d, M) = (7, 3, 4, 7)$

Additionally, the proposed GAS exhibited the fastest convergence performance among the considered schemes in both metrics.

A question arises here 'Why does the proposed GAS converge so fast?' One major reason is the quadratic speedup of GAS with the aid of quantum superposition, entanglement, and amplitude amplification. Another reason is that the proposed GAS starts with the strict initial threshold, $y_0 = \overline{E}'_{\rm min} + 1$ of (45), and the number of Grover rotations L_i is chosen from a limited range, $[0, \bar{k}_{\rm opt})$. These proposals lead to fewer states of interest and higher probability of success, resulting in faster convergence.

In Fig. 4, we simply averaged 10^6 curves for each scheme. After converging to the minimum objective function value, which was 15, the value of each curve remained the same. That is, in Fig. 4, the point where each scheme reached the minimum was the maximum of 10^6 simulations. For example, the classical exhaustive search reached the minimum objective function value at the query complexity of approximately $5.0 \cdot 10^4$, and this complexity was the maximum of all the trials. The query complexity required



(a) The total number of classical iterations. (b) The total number of Grover rotations.

Fig. 5 CDF of evaluation metrics required to reach the optimal solution, where we had $(n, w, d, M) = (7, 3, 4, 7)$

to reach the minimum is an important metric and is a stochastic variable. Its statistical property is investigated in Fig. 5.

In Fig. 5, the cumulative distribution function (CDF) of the query complexity required to reach the optimum was investigated. In Fig. 4, the difference between the conventional GAS and the proposed GAS appears to be small, but when compared with CDF in Fig. 5, the difference is clear. This result indicates that on average the proposed GAS reaches the optimal solution faster than the conventional one in both the total numbers of classical iterations and Grover rotations. That is, our approach is capable of reducing the constant overhead involved in the quadratic speedup of the conventional GAS.

5 Conclusions

Although much work has been done on binary constant weight codes, its parameter-dependent combinatorial explosion is still a problem. In this paper, we proposed a quantum search algorithm for binary constant weight codes. The search problem was formulated as a polynomial binary optimization problem, and bounds were derived for the minimum of the objective function value and the exact number of solutions. We proposed a modified algorithm for GAS using the derived bounds and showed that it can further accelerate the quantum algorithm while reducing the number of qubits. The overall query complexity remains exponential because the problem is considered NP-complete. Similar approaches that utilize problem-specific bounds, such as those in this study, have the potential for wide use in applications of quantum search algorithms to other problems, thereby improving the feasibility of quantum speedup.

Appendix A: Proof of Theorem 4

We regard a code as a matrix $\{0, 1\}^{M\times n}$ where each row vector is a codeword containing w 1 s. As mentioned in the proof of Theorem 3, if we have a code that satisfies a given

condition (n, w, M, d), a new code generated by swapping any two columns also satisfies the same condition (n, w, M, d). Using this property, we prove that new codes can be generated by reordering the first w columns of an existing code. In the following, we state the lemmas necessary to prove Theorem 4. Note that $\mathbf{p}_0$ is the first row of the combinatorial matrix $\mathbf{P}(n, w)$ of (31), and, hereinafter, it is assumed to be included in the code as a codeword without loss of generality.

Lemma 1 *If $A(n-1, d, w) < M$, then each column of a code that satisfies the condition (n, w, M, d) contains at least one 1.*

Proof Suppose that a code that satisfies the condition (n, w, M, d) exists, and a column of that code contains only 0 s. Then, a code generated by removing that column satisfies the condition $(n-1, w, M, d)$. The necessary and sufficient condition for such a code to exist is $A(n-1, d, w) \geq M$, which violates the assumption $A(n-1, d, w) < M$. □

Lemma 2 *If $A(n-1, d, w) < M-1$, then each column of a code that satisfies the condition (n, w, M, d) contains at least two 1 s.*

Proof According to Lemma 1, a column that contains only 0 s in a code does not exist under the condition $(n-1, w, M, d)$. Then, we prove that no column contains only one 1. Suppose that there exists a code that contains only one 1 in a column. A code generated by removing that column and that row, corresponding to the position of the contained 1, satisfies the condition $(n-1, w, M-1, d)$. The necessary and sufficient condition for such a code is $A(n-1, d, w) \geq M-1$, which violates the assumption $A(n-1, d, w) < M-1$. □

Lemma 3 *The first w columns of any codeword except for $\mathbf{p}_0$ contain at most $w - \frac{d}{2}$ 1 s.*

Proof Except for $\mathbf{p}_0$, suppose that a codeword exists such that the first w columns contain more 1 s than $w - \frac{d}{2}$. The inner product between that codeword and $\mathbf{p}_0$ is greater than $w - \frac{d}{2}$. From (22), this violates the condition that the minimum distance of the code is at least d. □

Corollary 1 *The last $n-w$ columns of any codeword except for $\mathbf{p}_0$ contain at least $\frac{d}{2}$ 1 s.*

Proof Each codeword contains w 1 s. Thus, it is clear from Lemma 3. □

Lemma 4 *If $w \leq d$, then there is no pair of codewords such that the last $n-w$ columns are identical.*

Proof According to Corollary 1, the last $n-w$ columns of any codeword except for $\mathbf{p}_0$ contain at least $\frac{d}{2}$ 1 s. If a pair of codewords exist, the last $n-w$ columns of which are identical, then $w - \frac{d}{2} > \frac{d}{2}$ holds, which violates the assumption $w \leq d$. □

Lemma 5 *If $A(n-1, d, w) < M-1$ and $w - \frac{d}{2} = 1$, then $t \geq w!$ holds.*

Proof According to Lemma 2, each column contains at least two 1 s. Also, according to Lemma 3, the number of 1 s in the first w columns is at most one except for $\mathbf{p}_0$. Lemmas 2 and 3 indicate that a code that satisfies the condition contains each column of $[\mathbf{I}_w \ \mathbf{0}_{w,M-w}]^{\mathrm{T}} \in \{0,1\}^{M\times w}$ in the first w columns. By reordering the first w columns, $w!$ number of new distinguishable codes can be generated, and accordingly, the minimum number of solutions is calculated as $w!$. □

Based on the above lemmas, we prove Theorem 4.

Theorem 6 *If $w \le d$ and $A(n-1,d,w) < M-1$, then the exact number of solutions t is lower bounded by*

$$t \ge \underline{t} = \begin{cases} w! & (w - \frac{d}{2} = 1) \\ \min\limits_{2\le i\le w-d/2} \binom{w}{i} & (w - \frac{d}{2} \ge 2). \end{cases} \tag{A1}$$

Proof According to Lemma 5, the $w - \frac{d}{2} = 1$ case is clear. In the $w - \frac{d}{2} \ge 2$ case, each column of a code contains at least two 1 s, according to Lemma 2. Here, if the first w columns of any codeword except for $\mathbf{p}_0$ contain at most one 1, non-overlapping $w!$ number of codes can be generated by reordering these columns as in Lemma 5. Similarly, if the first w columns contain $2 \le i \le w - \frac{d}{2}$ 1 s, $\binom{w}{i}$ codes can be generated by reordering these columns. Also, according to Lemma 4, since the last $n-w$ columns are all different, any generated code does not overlap with others. That is, if $w - \frac{d}{2} \ge 2$, the following inequality holds:

$$\begin{aligned} t \ge \underline{t} &= \min\left\{ w!, \binom{w}{2}, \binom{w}{3}, \ \cdots, \binom{w}{w-\frac{d}{2}} \right\} \\ &= \min_{2\le i\le w-d/2} \binom{w}{i}. \end{aligned} \tag{A2}$$

□

Appendix B: Proof of Theorem 5

Proof Let the argument of (55) be

$$h(k) = \frac{4k^2\alpha}{2k\alpha - \sin(4k\theta)}, \tag{B3}$$

where we have $\alpha = \sin(2\theta) > 0$. The domain of $h(k)$ is $k \in [1,\infty)$. Then, to eliminate $\sin(4k\theta)$, we consider the following inequality

$$\frac{4k^2\alpha}{2k\alpha+1} \le h(k) \le \frac{4k^2\alpha}{2k\alpha-1} \tag{B4}$$

that holds for $k > 1/(2\alpha)$. Next, for real numbers k_1 and k_2 that satisfies $1/(2\alpha) < k_1 < k_2$, we have an inequality

$$\frac{4k_1^2\alpha}{2k_1\alpha - 1} \leq \frac{4k_2^2\alpha}{2k_2\alpha + 1} \tag{B5}$$

and we obtain

$$k_2 \geq \underline{k}_2 = \frac{k_1^2\alpha + k_1\sqrt{k_1^2\alpha^2 + 2k_1\alpha - 1}}{2k_1\alpha - 1}, \tag{B6}$$

which indicates that $k_2 \in [\underline{k}_2, \infty) \Rightarrow h(k_1) \leq h(k_2)$ holds for a given k_1. Here, k_1 that minimizes $\underline{k}_2$ is

$$\arg\min_{k_1} \underline{k}_2 = \frac{1}{\alpha}. \tag{B7}$$

Then, the minimum value of $\underline{k}_2$ is expressed by

$$\begin{aligned} \min_{k_1}(\underline{k}_2) = \frac{1+\sqrt{2}}{\alpha} &= \frac{1+\sqrt{2}}{2\sqrt{\frac{t}{2^{q_1}} - (\frac{t}{2^{q_1}})^2}} \\ &\sim \frac{1+\sqrt{2}}{2}\sqrt{\frac{2^{q_1}}{t}} < \frac{1+\sqrt{2}}{2}\sqrt{\frac{2^{q_1}}{\underline{t}}}, \end{aligned} \tag{B8}$$

when $(t/2^{q_1})^2$ is sufficiently small. From (B8), $\overline{k}_{\mathrm{opt}}$ exists within $\left[1, \lceil \frac{1+\sqrt{2}}{2}\sqrt{\frac{2^{q_1}}{\underline{t}}} \rceil\right]$. □

Acknowledgements IBM and Qiskit are trademarks of International Business Machines Corporation.

Funding Open Access funding provided by Yokohama National University. This work was supported in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI (Grant Number 22H01484).

Declarations

Conflict of interest The authors declare no conflict of interest in any other work or publication.

References

1. Brouwer, A.E., Shearer, J.B., Sloane, N.J.A., Smith, W.D.: A new table of constant weight codes. IEEE Trans. Inf. Theory **36**(6), 1334–1380 (1990)
2. Tallini, L.G., Bose, B.: Design of balanced and constant weight codes for VLSI systems. IEEE Trans. Comput. **47**(5), 556–572 (1998)
3. Moon, J.N.J., Hughes, L.A., Smith, D.H.: Assignment of frequency lists in frequency hopping networks. IEEE Trans. Veh. Technol. **54**(3), 1147–1159 (2005)
4. Chung, F.R.K., Salehi, J.A., Wei, V.K.: Optical orthogonal codes: design, analysis and applications. IEEE Trans. Inf. Theory **35**(3), 595–604 (1989)
5. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. SIAM J. Comput. **38**(1), 97–139 (2008)
6. Yassine, H., Coon, J.P., Simmons, D.E.: Index programming for flash memory. IEEE Trans. Commun. **65**(5), 1886–1898 (2017)
7. Dela Cruz, B.P., Lampos, J.M., Palines, H., Sison, V.: A new construction of anticode-optimal Grassmannian codes. J. Algebra Comb. Discrete Struct. Appl. **8**(1) (2021)
8. Ishikawa, N., Sugiura, S., Hanzo, L.: 50 years of permutation, spatial and index modulation: From classic RF to visible light communications and data storage. IEEE Commun. Surv. Tutor. **20**(3), 1905–1938 (2018)
9. Fazeli, A., Nguyen, H.H.: Code design for non-coherent index modulation. IEEE Commun. Lett. **24**(3), 477–481 (2020)
10. Başar, E., Aygölü, Ü., Panayırcı, E., Poor, H.V.: Orthogonal frequency division multiplexing with index modulation. IEEE Trans. Signal Process. **61**(22), 5536–5549 (2013)
11. Wen, M., Zhang, Y., Li, J., Basar, E., Chen, F.: Equiprobable subcarrier activation method for OFDM with index modulation. IEEE Commun. Lett. **20**(12), 2386–2389 (2016)
12. Dang, S., Chen, G., Coon, J.P.: Lexicographic codebook design for OFDM with index modulation. IEEE Trans. Wirel. Commun. **17**(12), 8373–8387 (2018)
13. Ishikawa, N.: IMToolkit: an open-source index modulation toolkit for reproducible research based on massively parallel algorithms. IEEE Access **7**, 93830–93846 (2019)
14. Ishikawa, N.: Quantum speedup for index modulation. IEEE Access **9**, 111114–111124 (2021)
15. Yukiyoshi, K., Ishikawa, N.: On the capacity of generalized quadrature spatial modulation. IEEE Wirel. Commun. Lett. **12**(12), 2158–2162 (2023)
16. Johnson, S.: A new upper bound for error-correcting codes. IRE Trans. Inf. Theory **8**(3), 203–207 (1962)
17. Brouwer, A.: A few new constant weight codes. IEEE Trans. Inf. Theory **26**(3), 366–366 (1980)
18. Nurmela, K.J., Kaikkonen, M.K., Ostergard, P.R.J.: New constant weight codes from linear permutation groups. IEEE Trans. Inf. Theory **43**(5), 1623–1630 (1997)
19. Smith, D.H., Hughes, L.A., Perkins, S.: A new table of constant weight codes of length greater than 28. Electron. J. Combin. **13**(1) (2006)
20. Etzion, T., Vardy, A.: A new construction for constant weight codes. In: 2014 International Symposium on Information Theory and Its Applications, pp. 338–342 (2014)
21. Karp, R.M.: Reducibility among combinatorial problems. Complexity of Computer Computations, 85–103 (1972)
22. Gamal, A.E., Hemachandra, L., Shperling, I., Wei, V.: Using simulated annealing to design good codes. IEEE Trans. Inf. Theory **33**(1), 116–123 (1987)
23. Bland, J.A., Baylis, D.J.: Modelling constant weight codes using tabu search. Appl. Math. Model. **21**(11), 667–672 (1997)
24. Montemanni, R., Smith, D.H.: Heuristic algorithms for constructing binary constant weight codes. IEEE Trans. Inf. Theory **55**(10), 4651–4656 (2009)
25. Moore, G.E.: Cramming more components onto integrated circuits. Electronics **38**(8) (1965)
26. Choi, S., Moses, W.S., Thompson, N.: The quantum tortoise and the classical hare: a simple framework for understanding which problems quantum computing will accelerate (and which it will not). arXiv:2310.15505 (2023)
27. Kadowaki, T., Nishimori, H.: Quantum annealing in the transverse ising model. Phys. Rev. E **58**(5), 5355–5363 (1998)
28. Farhi, E., Goldstone, J., Gutmann, S.: A quantum approximate optimization algorithm. arXiv:1411.4028 (2014)

29. Stilck França, D., García-Patrón, R.: Limitations of optimization algorithms on noisy quantum devices. Nat. Phys. **17**(11), 1221–1227 (2021)
30. Suzuki, Y., Endo, S., Fujii, K., Tokunaga, Y.: Quantum error mitigation as a universal error reduction technique: applications from the NISQ to the fault-tolerant quantum computing eras. PRX Quantum **3**(1), 010345 (2022)
31. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th Annual Symposium on Foundations of Computer Science, pp. 124–134 (1994)
32. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of ACM Symposium on Theory of Computing, pp. 212–219 (1996)
33. Boyer, M., Brassard, G., Høyer, P., Tapp, A.: Tight bounds on quantum searching. Fortschritte der Physik **46**(4–5), 493–505 (1998)
34. Durr, C., Hoyer, P.: A quantum algorithm for finding the minimum. arXiv:quant-ph/9607014 (1999)
35. Bulger, D., Baritompa, W.P., Wood, G.R.: Implementing pure adaptive search with Grover's quantum algorithm. J. Optim. Theory Appl. **116**(3), 517–529 (2003)
36. Baritompa, W.P., Bulger, D.W., Wood, G.R.: Grover's quantum algorithm applied to global optimization. SIAM J. Optim. **15**(4), 1170–1184 (2005)
37. Gilliam, A., Woerner, S., Gonciulea, C.: Grover adaptive search for constrained polynomial binary optimization. Quantum **5**, 428 (2021)
38. Norimoto, M., Mori, R., Ishikawa, N.: Quantum algorithm for higher-order unconstrained binary optimization and MIMO maximum likelihood detection. IEEE Trans. Commun. **71**(4), 1926–1939 (2023)
39. Norimoto, M., Mikuriya, T., Ishikawa, N.: Quantum speedup for multiuser detection with optimized parameters in Grover adaptive search. IEEE Access **12**, 83810–83821 (2024)
40. Sano, Y., Norimoto, M., Ishikawa, N.: Qubit reduction and quantum speedup for wireless channel assignment problem. IEEE Trans. Quantum Eng. **4**, 1–12 (2023)
41. Sano, Y., Mitarai, K., Yamamoto, N., Ishikawa, N.: Accelerating Grover adaptive search: qubit and gate count reduction strategies with higher order formulations. IEEE Trans. Quantum Eng. **5**, 1–12 (2024)
42. Giuffrida, L., Volpe, D., Cirillo, G.A., Zamboni, M., Turvani, G.: Engineering grover adaptive search: Exploring the degrees of freedom for efficient QUBO solving. IEEE J. Emerg. Sel. Top. Circuits Syst. **12**(3), 614–623 (2022)
43. Bennett, C.H., Bernstein, E., Brassard, G., Vazirani, U.: Strengths and weaknesses of quantum computing. SIAM J. Comput. **26**(5), 1510–1523 (1997)
44. Frenger, P.K., Svensson, N.A.B.: Parallel combinatory OFDM signaling. IEEE Trans. Commun. **47**(4), 558–567 (1999)
45. Brassard, G., Hoyer, P., Mosca, M., Tapp, A.: Quantum amplitude amplification and estimation. AMS Contemporary Math. **305**, 53–74 (2002)
46. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997)
47. Anis, M.S., Abby-Mitchell, Abraham, H., et al.: Qiskit: an open-source framework for quantum computing (2021)
48. Burden, R.L., Faires, J.D.: Numerical Analysis, 4th edn (1989)
49. Galántai, A.: The theory of Newton's method. J. Comput. Appl. Math. **124**(1), 25–44 (2000)
50. Botsinis, P., Ng, S.X., Hanzo, L.: Fixed-complexity quantum-assisted multi-user detection for CDMA and SDMA. IEEE Trans. Commun. **62**(3), 990–1000 (2014)