Grover Adaptive Search for Constrained Polynomial Binary Optimization

Austin Gilliam,[1] Stefan Woerner,[2] and Constantin Gonciulea[1]

[1] *JPMorgan Chase*
[2] *IBM Quantum, IBM Research – Zurich*

(Dated: August 11, 2020)

In this paper we discuss Grover Adaptive Search (GAS) for Constrained Polynomial Binary Optimization (CPBO) problems, and in particular, Quadratic Unconstrained Binary Optimization (QUBO) problems, as a special case. GAS can provide a quadratic speed-up for combinatorial optimization problems compared to brute force search. However, this requires the development of efficient oracles to represent problems and flag states that satisfy certain search criteria. In general, this can be achieved using quantum arithmetic, however, this is expensive in terms of Toffoli gates as well as required ancilla qubits, which can be prohibitive in the near-term. Within this work, we develop a way to construct efficient oracles to solve CPBO problems using GAS algorithms. We demonstrate this approach and the potential speed-up for the portfolio optimization problem, i.e. a QUBO, using simulation. However, our approach applies to higher-degree polynomial objective functions as well as constrained optimization problems.

I. INTRODUCTION

Using the laws of quantum mechanics, quantum computers offer novel solutions for resource-intensive problems. Quantum computers are theoretically proven to solve certain problems faster than a classical device [1–3] and are well-equipped to handle tasks such as factoring [2], linear systems of equations [4] and Monte-Carlo simulations [5–8]. In addition, quantum computers could provide practical solutions to optimization problems [9–13], such as *Quantum Unconstrained Binary Optimization* (QUBO) problems, which, for instance, find applications in resource allocation, machine learning, and partitioning.

In [1], *Grover Search* is introduced as a quantum algorithm for locating a known element in a database. It was shown to be optimal, with a quadratic speed-up over the best-known classical algorithms. Dürr and Høyer used Grover's algorithm in [14] to describe a minimization algorithm, which was later viewed as an example of the algorithmic framework known as *Grover Adaptive Search* (GAS) [15, 16]. GAS iteratively applies Grover Search to find the optimum value of an objective function, by using the best-known value as a threshold to flag all values smaller than the threshold in order to find a better solution.

GAS has been explored as a possible solution for combinatorial optimization problems [17], alongside variational algorithms such as Variational Quantum Eigensolver [10] and Quantum Approximate Optimization Algorithm [9]. One of the challenges inherent in GAS is the creation of efficient oracles, which need to be repeatedly adapted according to new parameters.

In this paper, we provide a framework for automatically generating efficient oracles for *Constrained Polynomial Binary Optimization* (CPBO). The objective function and constraints need to be efficiently encoded, for which we use a *Quantum Dictionary* [18], a pattern for representing key-value pairs as entangled quantum registers, that turns out to be efficient for polynomial functions – in particular for quadratics representing QUBO problems. The approach relies on the addition of classical numbers to a quantum register in superposition, conditioned on the state of another quantum register. It is similar to the method used in *Quantum Fourier Transform* (QFT) adders [19]. In a nutshell, given a boolean polynomial, the coefficient of each monomial is added to the value register conditioned on the qubits in the key register corresponding to the variables present in the monomial.

The remainder of this paper is organized as follows. Sec. II introduces GAS in general. Sec. III introduces QUBO problems, and shows how we can efficiently generate oracles to solve them using GAS algorithms, as well as how this approach extends to more general CPBO problems. In Sec. IV, we apply the developed technique to a concrete test case – portfolio optimization – and demonstrate it via simulation using Qiskit [20]. Sec. V concludes this paper and discusses possible directions of future research.

II. GROVER ADAPTIVE SEARCH

Optimization problems are often solved by sequential approximation methods. In many cases, such methods are the only choice, but they may be computationally more efficient even when a solution to a problem can be expressed in a closed form. GAS works in a similar way, as it repeatedly uses Grover Search to randomly sample from all solutions better than the current one.

Grover Search is often described as a search algorithm, because it was initially formulated in the context of finding a single state of interest in a superposition of n-qubit quantum states. The algorithm has been generalized to the case of multiple states of interest, in which case it is better interpreted as a sampling algorithm. It amplifies the amplitudes of the states of interest within a larger search space, thus, increasing the probability of measuring one of the target states.

Grover Search – the core element of GAS – needs three ingredients:

1. A state preparation operator A to construct a superposition of all states in the search space. Usually, A is implemented by Hadamard gates $H^{\otimes n}$, i.e. it constructs the equal superposition state:
$$H^{\otimes n}\left|0\right\rangle_n = \frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}\left|i\right\rangle_n. \tag{1}$$
2. An oracle operator O, that recognizes the states of interest and multiplies their amplitudes by -1. For instance, suppose $I \subset \{0,\dots,2^n-1\}$ denotes the set of target states and $A = H^{\otimes n}$, then
$$OA\left|0\right\rangle_n = \frac{1}{\sqrt{2^n}}\sum_{i\notin I}\left|i\right\rangle_n - \frac{1}{\sqrt{2^n}}\sum_{i\in I}\left|i\right\rangle_n. \tag{2}$$
3. The Grover diffusion operator D, that multiplies the amplitude of the $\left|0\right\rangle_n$ state (or, equivalently, all states except $\left|0\right\rangle_n$) by -1.

The state preparation operator A and oracle operator O recognize the states of interest as described above. The diffusion operator has the net effect of inverting all amplitudes in the quantum state about their mean. This causes all the amplitudes of the states of interest to be magnified, while the amplitudes of all other outcomes are decreased. More precisely, applying the Grover operator $G = ADA^\dagger O$ the right number of times to state $A\left|0\right\rangle_n$ – i.e. evaluating $G^r A\left|0\right\rangle_n$ – will maximally amplify the amplitudes of the states of interest. The optimal number of applications r depends on the number $N = 2^n$ of all states and the number s of states of interest, and is equal to $\lfloor\frac{\pi}{4}\sqrt{\frac{N}{s}}\rfloor$. This implies a probability of sampling a target state of at least $1/2$, which corresponds to a quadratic speed-up compared to classical search. Since s is in general unknown, we can either use Quantum Counting algorithms [21–24] to find s, or apply a randomized strategy. The latter is the essence of [25], where an algorithm for applying Grover Search for unknown s is presented.

This was then used to create a minimum-finding algorithm [14], which we refer to as GAS. In the following we outline GAS, which is formally given in Alg. 1.

Suppose a function $f : X \to \mathbb{R}$, where for ease of presentation assume $X = \{0,1\}^n$. Then, we are interested in solving $\min_{x\in X} f(x)$. The main idea of GAS is to construct A_y and O_y for a given threshold y such that they flag all states $x \in X$ satisfying $f(x) < y$, such that we can use Grover Search to find a solution $\tilde{x}$ with a function value better than y. Then we set $y = f(\tilde{x})$ and repeat until some formal termination criteria is met, e.g. based on the number of iterations, time, or progress in y.

While implementations of GAS vary around the specific use case [16, 26], the general framework still loosely follows the steps described in [14]. In the following, we will show how operator A and oracle O can be efficiently constructed for QUBO as well as CPBO problems.

Algorithm 1: Grover Adaptive Search

Input: $f : X \to \mathbb{R}$, $\lambda > 1$
Uniformly sample $x_1 \in X$ and set $y_1 = f(x_1)$;
Set $k = 1$ and $i = 1$;
repeat
  Randomly select the rotation count r_i from the set $\{0, 1, ..., \lceil k-1\rceil\}$;
  Apply Grover Search with r_i iterations, using oracles A_{y_i} and O_{y_i}. We denote the outputs x and y respectively;
  if $y < y_i$ **then**
    $x_{i+1} = x$, $y_{i+1} = y$, and $k = 1$
  else
    $x_{i+1} = x_i$, $y_{i+1} = y_i$, and $k = \lambda k$
  $i = i + 1$;
until a termination condition is met;

III. AUTOMATIC GENERATION OF QUBO AND CPBO ORACLES

A QUBO problem with n binary variables is specified by $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$, and $c \in \mathbb{R}$, and is defined as
$$\min_{x\in\{0,1\}^n}\left(\sum_{i,j=1}^{n} A_{ij}x_i x_j + \sum_{i=1}^{n} b_i x_i + c\right), \tag{3}$$
or more compactly as $\min_{x\in\{0,1\}^n}(x^T Ax + b^T x + c)$, i.e. $f(x) = x^T Ax + b^T x + c$.

In the following, we show how to efficiently construct the GAS oracles for QUBO problems. We will construct A_y such that it prepares a n-qubit register to represent the equal superposition of all $\left|x\right\rangle_n$ and a m-qubit register to (approximately) represent the corresponding $\left|Q(x)-y\right\rangle_m$. Then, all states with $(Q(x)-y)$ negative should be flagged by O_y. Note that in the implementation discussed, the oracle operator is actually independent of y, but this is not a requirement. For clarity, we will refer to the oracle as O when the oracle is independent of y. More formally, we show how to construct the oracles such that
$$A_y\left|0\right\rangle_n\left|0\right\rangle_m = \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}\left|x\right\rangle_n\left|Q(x)-y\right\rangle_m, \text{ and} \tag{4}$$
$$O\left|x\right\rangle_n\left|z\right\rangle_m = \text{sign}(z)\left|x\right\rangle_n\left|z\right\rangle_m \tag{5}$$
where $\left|x\right\rangle$ is the binary encoding of the integer x. Furthermore, we will show how the developed technique can be used to extend GAS to higher-degree polynomials of binary variables, as well as to constrained optimization.

A. Construction of operator A

To construct A, we will use a Quantum Dictionary, as introduced in [18], and summarize the construction in the following.

Given an m-qubit register and an angle $\theta \in [-\pi, \pi)$, we wish to prepare a quantum state whose state vector represents a "periodic signal" equivalent to a geometric sequence of length 2^m:

$$G(\theta) = (1, e^{i\theta}, \ldots, e^{i(2^m-1)\theta}). \tag{6}$$

In other words, after normalizing, we need a unitary operator defined by:

$$U_{\mathrm{G}}(\theta) H^{\otimes m} |0\rangle_m = \frac{1}{\sqrt{2^m}} \sum_{k=0}^{2^m-1} e^{ik\theta} |k\rangle_m . \tag{7}$$

The simplest implementation of $U_{\mathrm{G}}(\theta)$ uses the phase gate $R(\theta)$ that, when applied to a qubit, rotates the phase of the amplitudes of the states having $|1\rangle$ in that qubit. In Qiskit, this gate is the $U_1(\theta)$ operator [20]. The circuit for $U_{\mathrm{G}}(\theta)$ is shown in Fig. 1, and consists of applying the gate $R(2^i\theta)$ to the qubit $m-1-i$ in the m-qubit register prepared in the state of equal superposition in equation (1).

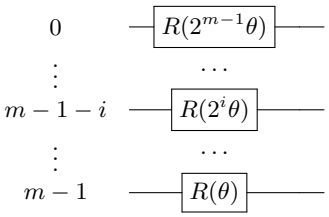

FIG. 1. The $U_{\mathrm{G}}(\theta)$ circuit.

In [18] an alternative way to implement U_{G} was introduced by applying controlled R_y rotations to an ancilla register containing the encoding of an eigenstate of R_y, as explained in Appendix A.

Note that QFT applied to a register containing the binary encoding of an non-negative integer also creates a geometric sequence of amplitudes in that register. U_{G} can be seen as a shortcut for the QFT when the encoded numbers are known classically, as we avoid multi-qubit interactions.

Given an integer $-2^{m-1} \leq k < 2^{m-1}$, if we apply $U_{\mathrm{G}}(\frac{2\pi}{2^m}k)$, followed by the inverse QFT to an m-qubit register prepared in the state of equal superposition in equation (1), we end up with $k \pmod{2^m}$ being encoded in the register, as shown in Fig. 2.

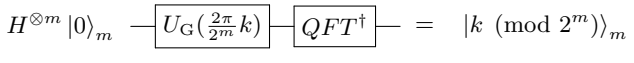

FIG. 2. Geometric sequence encoding of an integer k.

This representation is called the binary Two's Complement of k, which just adds 2^m to negative values k, similar to the way we can represent negative angles with their complement, e.g. equating $-\pi/4$ with $7\pi/4$. The reason this representation occurs naturally in this context is due to the fact that rotation composition behaves like modular arithmetic.

FIG. 3. $U_{\mathrm{G}}(\theta)$ followed by inverse QFT.

Fig. 3 shows the general case of an angle $\theta \in [-\pi, \pi)$. The application of the same sequence of gates results in a quantum state whose state vector consists of the inner product between $G(\theta)$ and the Fourier bases $G(\frac{2\pi}{2^m}j)$, representing a similarity measure between θ and $\frac{2\pi}{2^m}j$, for $0 \leq j \leq 2^m - 1$. We will call the operator preparing this state $U_{\mathrm{Fejér}}$ because the outcome probability distribution is the Fejér distribution [27]:

$$U_{\mathrm{Fejér}}(\theta) |0\rangle_m = \frac{1}{\sqrt{2^m}} \sum_{j=0}^{2^m-1} \langle G(\theta), G(\frac{2\pi}{2^m}j)\rangle |j\rangle . \tag{8}$$

If $\theta = \frac{2\pi}{2^m}a$, for a real number $-2^{m-1} \leq a < 2^{m-1}$, then $U_{\mathrm{Fejér}}(\theta)|0\rangle_m$ prepares a state whose two most likely measurement outcomes are the closest two integers to a. The probability of measuring one of them is at least 81% [28]. Fig. 4 shows the probability distribution of the outcomes for $a = -4.76$ and $m = 4$.

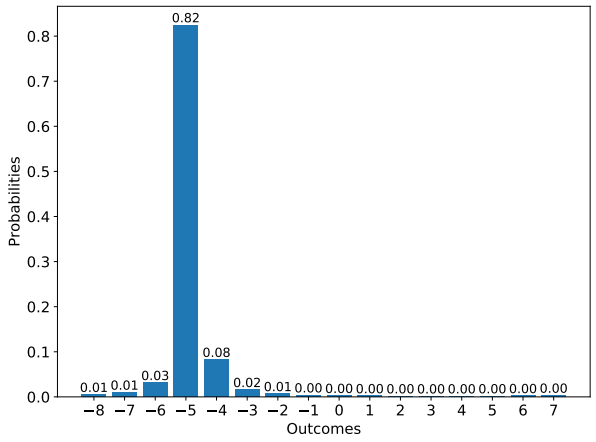

FIG. 4. The encoding of a non-integer ($a = -4.76$ shown here) leads to a superposition of approximations.

When encoding non-integers, we have two choices. The first is to approximate them with fractions with a common denominator and encode the numerator into quantum registers before performing computations, cf. Appendix B. The second is to phase-encode real numbers and let the inverse QFT convert the result of the computation into a superposition of approximations as shown above. The second choice may require multiple measurements. While encoding non-integers this way has a lot of potential, in this paper we will focus on integers only.

Next we will discuss the application of $U_{\mathrm{G}}(\theta)$ to a register $|z\rangle_m$ representing the values of a function, controlled on a register $|x\rangle_n$ representing the inputs of the same

function. In general, the application of a unitary operator U to $|z\rangle_m$ controlled by a set $J \subseteq \{0,\ldots,n-1\}$ of qubits of $|x\rangle_n$ can be expressed as

$$C^J(U)\,|x\rangle\,|z\rangle = |x\rangle\,U^{\prod_{j\in J} x_j}\,|z\rangle\,, \tag{9}$$

and an example is shown in Fig. 5.

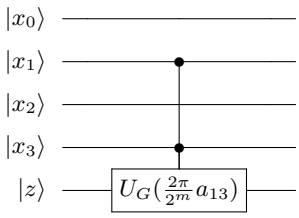

FIG. 5. Example of $C^{\{1,3\}}(U)$ when $n = 4$.

With this notation we can define the operator A for a boolean polynomial of n variables. The general form of such a polynomial is:

$$P(x) = \sum_{J\subseteq\{0,\ldots,n-1\}} a_J \prod_{j\in J} x_j. \tag{10}$$

Each subset $J \subseteq \{0,\ldots,n-1\}$ has a corresponding monomial $\prod_{j\in J} x_j$. The free term is represented by $a_\emptyset$.

Now we are ready to define the operator A, as shown in Fig. 6. It consists of applying a controlled geometric sequence transformation $C^J(U_G(\frac{2\pi}{2^m}a_J))$ for each subset $J \subseteq \{0,\ldots,n-1\}$ with a non-zero coefficient a_J, followed by a single application of the inverse QFT.

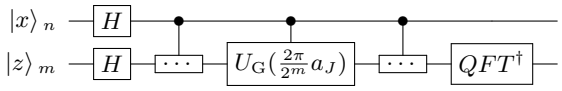

FIG. 6. Circuit for operator A.

Note that QUBO polynomials only have single and pairs of qubits, i.e. $|J| \leq 2$. An example of encoding a single monomial is shown in Fig. 7.

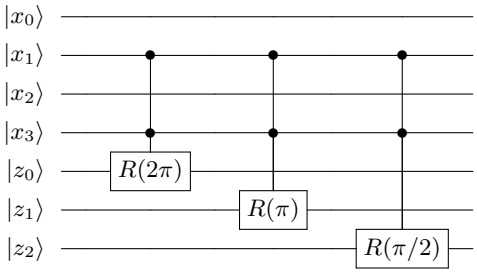

FIG. 7. Example of encoding the monomial $2x_1x_3$ when $n = 4$ and $m = 3$.

The operator A prepares a state where the $|x\rangle_n$ register holds all 2^n inputs in superposition, entangled with the corresponding values $P(x)$ encoded in the $|z\rangle_m$ register:

$$A\,|0\rangle_n\,|0\rangle_m = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle_n\,|P(x)\rangle_m\,, \tag{11}$$

where we are assuming that an appropriate size m for the value register is known.

Note that operator A can be regarded as controlled addition, which for integers can be performed with straight quantum arithmetic, which requires explicit encoding of negative integers, and a lot more gates – especially multi-controlled ones [29–31].

The desired A_y operator is obtained by adding the $-y$ constant to the free term of the original polynomial. This construction works for polynomials of arbitrary degree. However, the circuit complexity scales with the same polynomial degree. For every monomial, we require a controlled U_G that has the name number of controls as the number of variables in the monomial, i.e. the grade of the monomial.

B. Construction of oracle O

At each step of the algorithm, we are adding a constant to the polynomial, and search for remaining negative values. This means the oracle just needs to recognize negative integers. Since values are represented in Two's Complement, the most significant qubit in the value register can be used to recognize negative integers. The typical oracle that multiplies target amplitudes by -1 can be applied with this qubit as its control.

Note that the oracle stays unchanged between iterations, because we add a constant to the polynomial, which may lead to overflow in the value register. In order to avoid that, we may need to increase the number of qubits in the value register by 1.

Alternatively, threshold-based oracles (which are potentially more expensive) can be used, that will reduce the search space by filtering the numbers above the given threshold.

C. Constrained Optimization

It is common for optimization problems to impose additional constraints, for example the total number or cost of assets in a portfolio may be subjected to an upper bound. Such constraints translate into further reductions of the search space based on the key register. For example, the number of assets in a portfolio corresponds to the number of 1s in the binary representation of the input of the objective function, called the Hamming weight.

It is straightforward to use the GAS oracles introduced in Sec. III A and III B to take into account polynomial equality and inequality constraints on the key register. Therefore, we can add additional registers to evaluate other polynomials. Whether an inequality constraint is satisfied or not can again be mapped to the sign qubit by applying an appropriate shift to the polynomial. Equality constraints are a bit more evolved, as they require the detection of a particular state, which essentially has the same complexity as the Grover diffusion operator D.

This leads to a set of qubits flagging feasible states: one qubit identifying the states that correspond to objective values below the current threshold, and one qubit for each constraint flagging feasible states. Applying a logical AND-operation to all of them essentially acts as an intersection of the individually flagged states and allows to construct oracles for CPBO.

IV. TEST CASES

In the remainder of this paper we demonstrate the introduced technique on the portfolio optimization problem, and then show a simple example on quantum hardware.

A. Portfolio Optimization

Suppose an investment universe consisting of n assets, denoted by $i = 1, \ldots, n$, their corresponding expected returns $\mu \in \mathbb{R}^n$ and covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$. Furthermore, we consider a given risk factor $q \geq 0$, which determines the considered risk appetite. The resulting objective function is

$$\min_{x \in \{0,1\}^n} \left(q x^T \Sigma x - \mu^T x \right). \tag{12}$$

In other words, we want to minimize the weighted variance minus the expected portfolio return. Setting $q = 0$ implies a risk neutral investor, while increasing q increases its risk averseness.

In the presented form, portfolio optimization is a QUBO problem. We can extend it by imposing a budget constraint of the form

$$\sum_{i=1}^{n} x_i = B, \tag{13}$$

where $B \in \{0, \ldots, n\}$ denotes the number of assets to be chosen. Equality constraints can be recast as penalty terms

$$\lambda \left(\sum_{i=1}^{n} x_i - B \right)^2, \tag{14}$$

and added to the objective (since we minimize), where $\lambda > 0$ is a large number to enforce the constraint to be satisfied. This results in a quadratic term that will again lead to a QUBO problem. However, with the methodology introduced in Sec. III C, we can also model more complex constraints, e.g. a budget inequality constraint of the form

$$\sum_{i=1}^{n} c_i x_i \leq B, \tag{15}$$

where $c_i \in \mathbb{R}$ denote the asset prices, which does usually make more sense in practice than (13). In the following, for simplicity, we do not consider any budget constraint.

To demonstrate the developed methodology, we consider a portfolio of $n = 3$ assets, risk factor $q = 0.5$, and returns described by

$$\mu = \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} 2 & 0 & -4 \\ 0 & 4 & -2 \\ -4 & -2 & 10 \end{pmatrix}, \tag{16}$$

which leads to

$$\min_{x \in \{0,1\}^3} \left(-2x_1x_3 - x_2x_3 - 1x_1 + 2x_2 - 3x_3 \right). \tag{17}$$

The objective function with added constant $-y$ (where y is the current threshold) has an associated A_y operator and the oracle O recognizes negative values, as introduced in Sec. III. We set the initial threshold $y_1 = 0$, and stop searching if an improvement has not been found in three consecutive iterations of the algorithm.

For each iteration of GAS, we determine the number of applications of the Grover iterate as defined in Alg. 1. We apply A_y for the current threshold, and then apply the Grover iterate $G_y = A_y D A_y^\dagger O$ for the predetermined number of applications. If the measured value is feasible, i.e. less than y, we update the threshold. This process repeats until we have seen no improvement for three consecutive iterations.

Classically, we can determine the original minimum value by keeping track of the total shift, or by calculating the value of the objective function for the minimum key. The results of this simulated experiment are shown in Fig. 8.

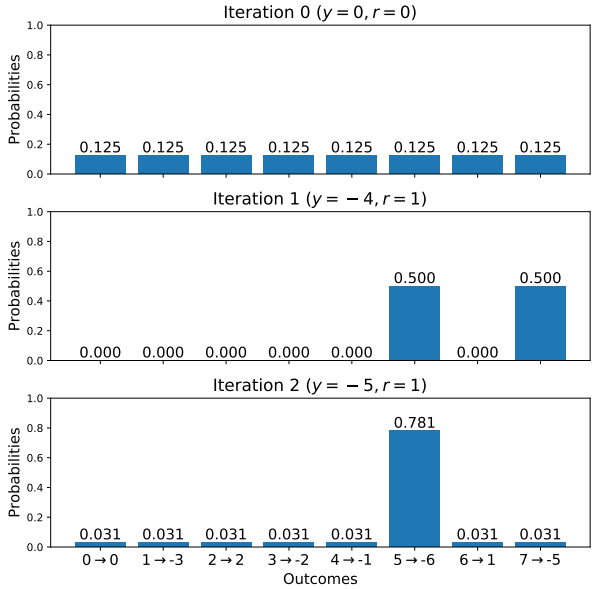

FIG. 8. The output probabilities of GAS for three iterations, with respective thresholds y and r applications of the Grover operator.

B. Trials on Real Hardware

The experiments discussed in this section were run on the IBM ibmq_toronto device, with Quantum Volume 32 [32]. The configuration details for ibmq_toronto at the time of our experiments is given in Appendix C. Readout error-mitigation techniques were applied to the results of each circuit [20, 33].

Let us consider a simple example which can be run on current quantum hardware:

$$\min_{x\in\{0,1\}^2}(-2+x_1+x_2). \tag{18}$$

As in the previous subsection, we set the initial threshold $y_1 = 0$, and stop searching after three iterations with no improvement. Note that due to the nosiness inherent in the present era of quantum hardware, there is a possibility that an invalid key-value pair will be measured. To prevent this we repeat the computation several times, and take the outcome with the maximum probability as the measured result.

The results of the experiment are shown in Fig. 9. Note that in this example, the valid measurement outcomes are $0 \to -2$, $1 \to -1$, $2 \to -1$, and $3 \to 0$. In the first iteration we see the four outcomes in an approximately-equal superposition, and sample randomly (we do not apply the Grover iterate). We measure $1 \to -1$, and thus we update the threshold to $y_2 = -1$ and the number of iterations to $r = 1$. In the second iteration of Grover Adaptive Search we see the results of the amplification, where $0 \to -2$ (the minimum) has the highest probability.

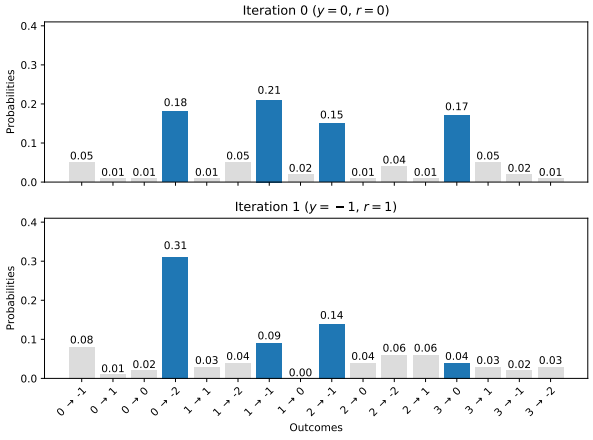

FIG. 9. The output probabilities of GAS for two iterations run on real quantum hardware, with respective thresholds y and r applications of the Grover operator. The valid measurement outcomes are shown in blue, while the invalid measurement outcomes are shown in grey.

V. CONCLUSION

In this paper we introduced an efficient way to implement the oracles required for solving *Constrained Polynomial Binary Optimization* problems using *Grover Adaptive Search*. This problem class is very general and contains for instance QUBO problems. Our approach significantly reduces the number of gates required compared to standard quantum arithmetic approaches, i.e. it lowers the requirements to apply GAS on real quantum hardware for practically relevant problems. We demonstrated our algorithm on the portfolio optimization problem, i.e. a QUBO, where we could reliably find the optimal solution, and on real quantum hardware. Within this manuscript we focused mainly on problems with integer coefficients. Leveraging the Fejér distribution to approximate fractional values may be an interesting approach and subject to future research.

VI. ACKNOWLEDGMENTS

This paper was prepared for information purposes by the Future Lab for Applied Research and Engineering (FLARE) Group of JPMorgan Chase & Co. and its affiliates, and is not a product of the Research Department of JPMorgan Chase & Co. JPMorgan Chase & Co. makes no explicit or implied representation and warranty, and accepts no liability, for the completeness, accuracy or reliability of information, or the legal, compliance, tax or accounting effects of matters contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction.

IBM, the IBM logo, and ibm.com are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. The current list of IBM trademarks is available at https://www.ibm.com/legal/copytrade.

Appendix A: Quantum Dictionary

The *Quantum Dictionary* was introduced in [18] as a quantum computing pattern for encoding functions, in particular polynomials, into a quantum state using geometric sequences. The paper shows how quantum algorithms like search and counting applied to a quantum dictionary allow to solve combinatorial optimization and QUBO problems more efficiently than using classical methods.

Encoding a geometric sequence can be done using the phase gate as we explained earlier, but it is worth men-

tioning an alternative method described in [18], which uses the R_y family of gates.

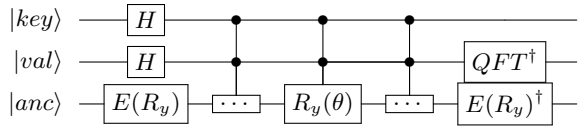

FIG. 10. Quantum Dictionary circuit.

The R_y gate is applied to an ancillary register containing one of its eigenstates prepared by an operator $E(R_y)$, conditioned on both the key and value registers. The rotation angle θ is different for each application, representing a number that contributes to the values corresponding to keys that have the conditioned key as a subset. In particular, when encoding a polynomial, a rotation will be applied for each of its coefficients.

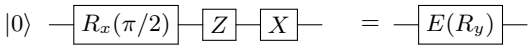

FIG. 11. Eigenstate preparation for R_y.

In [18] we used the R_y operator, with the eigenstate $1/\sqrt{2}(i\,|0\rangle + |1\rangle)$, independent of the rotation angle. This state can be prepared by the circuit in Fig. 11. The corresponding eigenvalue of $R_y(2\theta)$ is $e^{i\theta}$.

Appendix B: Approximating Non-Integer Coefficients

If we relax the assumption that all coefficients are integers, we can approximate non-integers by dividing all values in μ and Σ by the largest (scaling the range of the coefficients to $[-1, 1)$), and approximating each value k by a fraction

$$\frac{k}{2^m}, \tag{B1}$$

with $-2^{m-1} \leq k < 2^{m-1}$, where m is the number of value qubits. As an example, suppose $n = 3$, $m = 5$, $q = 0.5$, and

$$\mu = \begin{pmatrix} -3.77 \times 10^{-3} \\ 1.09 \times 10^{-3} \\ 2.41 \times 10^{-3} \end{pmatrix}. \tag{B2}$$

Scaling the coefficients of μ leads to

$$\mu = \begin{pmatrix} -1.0 \\ 0.29 \\ 0.64 \end{pmatrix}, \tag{B3}$$

and approximating them by fractions leads to

$$\mu = \begin{pmatrix} -16/32 \\ 5/32 \\ 10/32 \end{pmatrix}. \tag{B4}$$

As the approximated function coefficients have a common denominator, we can ignore the denominator and treat the values as integers

$$\mu = \begin{pmatrix} -16 \\ 5 \\ 10 \end{pmatrix}, \tag{B5}$$

which results in the optimization problem

$$\min_{x \in \{0,1\}^3} -(-16x_1 + 5x_2 + 10x_3). \tag{B6}$$

Appendix C: Hardware Specifications

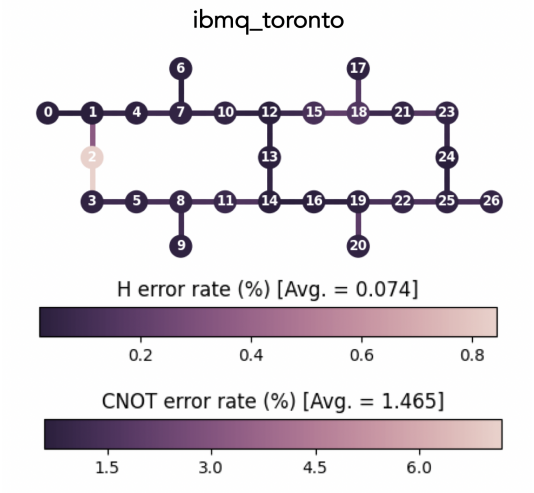

FIG. 12. The error map for ibmq_toronto.

The error map for ibmq_toronto in Fig. 12 was generated on the day of experimentation using Qiskit's visualization library [20].

REFERENCES

[1] Lov K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96 (ACM, New York, NY, USA, 1996) pp. 212–219.

[2] Peter W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," SIAM Journal on Computing **26**, 1484–1509 (1997).

[3] B. P. Lanyon, J. D. Whitfield, G. G. Gillett, M. E. Goggin, M. P. Almeida, I. Kassal, J. D. Biamonte, M. Mohseni, B. J. Powell, M. Barbieri, and et al., "Towards quantum chemistry on a quantum computer," Nature Chemistry **2**, 106–111 (2010).

[4] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd, "Quantum algorithm for linear systems of equations," Physical Review Letters **103** (2009), 10.1103/physrevlett.103.150502.

[5] Patrick Rebentrost, Brajesh Gupt, and Thomas R. Bromley, "Quantum computational finance: Monte carlo pricing of financial derivatives," Phys. Rev. A **98**, 022321 (2018).

[6] Stefan Woerner and Daniel J. Egger, "Quantum risk analysis," npj Quantum Information **5**, 15 (2019).

[7] Daniel J. Egger, Ricardo Gacía Gutiérrez, Jordi Cahué Mestre, and Stefan Woerner, "Credit risk analysis using quantum computers," (2019), arXiv:1907.03044 [quant-ph].

[8] Nikitas Stamatopoulos, Daniel J. Egger, Yue Sun, Christa Zoufal, Raban Iten, Ning Shen, and Stefan Woerner, "Option pricing using quantum computers," (2019), arXiv:1905.02666 [quant-ph].

[9] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann, "A quantum approximate optimization algorithm," (2014), arXiv:1411.4028 [quant-ph].

[10] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man Hong Yung, Xiao Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien, "A variational eigenvalue solver on a photonic quantum processor," Nature Communications **5** (2014), 10.1038/ncomms5213.

[11] Giacomo Nannicini, "Performance of hybrid quantum-classical variational heuristics for combinatorial optimization," Phys. Rev. E **99**, 013304 (2019).

[12] Panagiotis Kl. Barkoutsos, Giacomo Nannicini, Anton Robert, Ivano Tavernelli, and Stefan Woerner, "Improving variational quantum optimization using cvar," (2019), arXiv:1907.04769 [quant-ph].

[13] Lee Braine, Daniel J. Egger, Jennifer Glick, and Stefan Woerner, "Quantum algorithms for mixed binary optimization applied to transaction settlement," (2019), arXiv:1910.05788 [quant-ph].

[14] Christoph Dürr and Peter Høyer, "A quantum algorithm for finding the minimum," (1996), arXiv:quant-ph/9607014 [quant-ph].

[15] D. Bulger, W. P. Baritompa, and G. R. Wood, "Implementing pure adaptive search with grover's quantum algorithm," Journal of Optimization Theory and Applications **116**, 517–529 (2003).

[16] W. P. Baritompa, D. W. Bulger, and G. R. Wood, "Grover's quantum algorithm applied to global optimization," SIAM J. on Optimization **15**, 1170–1184 (2005).

[17] Ehsan Zahedinejad and Arman Zaribafiyan, "Combinatorial optimization on gate model quantum computers: A survey," (2017), arXiv:1708.05294 [quant-ph].

[18] Austin Gilliam, Charlene Venci, Sreraman Muralidharan, Vitaliy Dorum, Eric May, Rajesh Narasimhan, and Constantin Gonciulea, "Foundational patterns for efficient quantum computing," (2019), arXiv:1907.11513 [quant-ph].

[19] Thomas G. Draper, "Addition on a quantum computer," (2000), arXiv:quant-ph/0008033 [quant-ph].

[20] Héctor Abraham et. al., "Qiskit: An open-source framework for quantum computing," (2019).

[21] Michele Mosca, "Quantum searching, counting, and amplitude amplification by eigenvector analysis," (1998).

[22] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp, "Quantum Amplitude Amplification and Estimation," Contemporary Mathematics **305** (2002).

[23] Yohichi Suzuki, Shumpei Uno, Rudy Raymond, Tomoki Tanaka, Tamiya Onodera, and Naoki Yamamoto, "Amplitude Estimation without Phase Estimation," (2019), arXiv:1904.10246.

[24] Scott Aaronson and Patrick Rall, "Quantum Approximate Counting, Simplified," (2019), arXiv:1908.10846.

[25] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp, "Tight bounds on quantum searching," Fortschritte der Physik **46**, 493–505 (1998).

[26] Benjamín Barán and Marcos Villagra, "Multiobjective optimization grover adaptive search," in *Recent Advances in Computational Optimization: Results of the Workshop on Computational Optimization WCO 2017*, edited by Stefka Fidanova (Springer International Publishing, Cham, 2019) pp. 191–211.

[27] Svante Janson, "Moments of gamma type and the brownian supremum process area," Probab. Surveys **7**, 1–52 (2010).

[28] Michael A. Nielsen and Isaac L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, 10th ed. (Cambridge University Press, New York, NY, USA, 2011).

[29] Steven A. Cuccaro, Thomas G. Draper, Samuel A. Kutin, and David Petrie Moulton, "A new quantum ripple-carry addition circuit," (2004), arXiv:quant-ph/0410184 [quant-ph].

[30] Thomas G. Draper, Samuel A. Kutin, Eric M. Rains, and Krysta M. Svore, "A logarithmic-depth quantum carry-lookahead adder," Quantum Information and Computation **6**, 351–369 (2006).

[31] Craig Gidney, "Halving the cost of quantum addition," Quantum **2**, 74 (2018).

[32] Andrew W. Cross, Lev S. Bishop, Sarah Sheldon, Paul D. Nation, and Jay M. Gambetta, "Validating quantum computers using randomized model circuits," Physical Review A **100** (2019).

[33] A. Dewes, F. R. Ong, V. Schmitt, R. Lauro, N. Boulant, P. Bertet, D. Vion, and D. Esteve, "Characterization of a two-transmon processor with individual single-shot qubit readout," Phys. Rev. Lett. **108**, 057002 (2012).