FIXED-POINT GROVER ADAPTIVE SEARCH FOR QUBO PROBLEMS

ÁKOS NAGY, JAIME PARK, CINDY ZHANG, ATITHI ACHARYA, AND ALEX KHAN

ABSTRACT. We apply and study a Grover-type method for Quadratic Unconstrained Binary Optimization (QUBO) problems. Inspired by Gilliam et al. [5], we construct a marker oracle for such problems and develop a new version of their Grover Adaptive Search, building on results of [8,12]. Our oracle design has improved circuit depth and gate count, including improved 2-qubit gate count and our adaptive search has improved performance guarantees, when compared to that of Gilliam et al. [5].

1. INTRODUCTION

Quadratic Unconstrained Binary Optimization (QUBO) problems provide some of the most interesting NP-Complete problems, such as cluster analysis, maximal graph cuts, and the Ising model. A QUBO problem consists of a (real-valued) quadratic polynomial on n binary variables and thus it can be described, uniquely, up to an overall additive constant, by an n-by-n upper-triangular real matrix. Approximation algorithms for such problems have been extensively studied for a long time via classical methods. Recently, classical–quantum hybrid methods have also been proposed, where the parameters of the quantum algorithm are classically optimized. The two main types of such algorithms are of QAOA-type [1,3,6,11] and Grover-type [5]. Compared to QAOA, Grover-type algorithms have certain set advantages and drawbacks. The promise of QAOA is that, using an easily implementable, low-depth circuit, one can prepare a quantum state whose dominant components in the computational basis correspond to high value configurations of the QUBO problem. On the other hand, Grover-type methods tend to have more complex circuits and amplification of components is all-or-nothing: one needs to set a threshold above which the method searches for values. However, what makes Grover-type methods still appealing is the fact that they usually require much less classical optimizations. Beyond query complexity, $l \in \mathbb{Z}_+$, (which is a parameter in both methods), in QAOA one needs to optimize input parameters on $2l$-dimensional spaces, whereas in a Grover-type method one only has the threshold to tune, which is much simpler. With that in mind, this paper investigates a Grover-type method for QUBO problems.

In [5], Gilliam et al. have proposed a marker oracle design for QUBO problems: given an instance of an integer QUBO problem, say, by n-by-n upper-triangular integer matrix, Q, and a threshold $y \in \mathbb{Z}$, their oracle marks states $|x\rangle$ exactly when $x^T Q x \geqslant y$. This is precisely the ingredient needed to run Grover-type search algorithms to find configurations with values above the threshold. Furthermore, in the same paper Gilliam et al. proposes a *Grover Adaptive Search* for QUBO problems. We explain their construction for the marker oracle in Section 2.1 and their Grover Adaptive Search in Section 4. This paper was inspired by these results of Gilliam et al. and our key results can be viewed as improvements on both their oracle design and adaptive method.

Let us now recall Grover's Search algorithm. In [7], Grover introduced a quantum amplitude amplification method that can be viewed as an unstructured search algorithm whose query complexity scales with the inverse

Date: November 4, 2023.
2020 *Mathematics Subject Classification.* 11E16, 68Q12, 81P65, 81P68.
Key words and phrases. Fixed-point Grover Search, Quadratic Binary Optimization.

square root of the ratio of the number of marked items to the number of all items. Since Grover's seminal work, many more variations of the algorithm have been proven, most notably for our paper, a "Fixed-point" version which fixes one half of the "soufflé problem", that is, the fact that to implement Grover's algorithm one needs to exactly know the exact ratio of marked items to compute the correct number of iterations ("queries") prescribed by the algorithm. Too few or too many queries can "under/overcook" the quantum state. In the Fixed-point Grover Search (FPGS) of Yoder et al. [12] it is enough to know a lower bound for the ratio and use that to compute the number of queries. In other words, one cannot overcook the quantum state. Furthermore, the number of queries prescribed by the FPGS still scales with the inverse square root of *lower bound used for the ratio of marked states*. This later claim can superficially be viewed as retaining the "quadratic speedup", but that is somewhat misleading as finding lower bounds might still be hard. This can be remedied by Li et al. [8] who study hybrid trail-and-error method, in which repeated FPGS circuits are run with exponentially decreasing guesses for the lower bound. One can immediately see that this method will reach a lower bound (unless the ratio was zero) with the optimal number of queries. In fact, Li et al. do more: they compute optimal (ratio and problem agnostic) parameters, with which this version of the FPGS has the best known query complexity (more on this in Section 4). These results of Yoder et al. and Li et al. form the basis of our improved adaptive method in Section 4.

While our study of marker oracle designs for QUBO problems is motivated by Grover-type algorithms, these oracles are also the ingredients for the actual realizations of the threshold-QAOA circuits for QUBO problems; cf. [6]. Similarly, any variation of QAOA (or other quantum algorithms) that uses "Grover-mixers" would require these oracles, for obvious reasons.

Organization of the paper: In Section 2, we introduce QUBO problems, outline of the construction of Gilliam et al. of the oracle for QUBOs [5], and present our modified design. In Section 3, we introduce the Fixed-point Grover Search and, using our oracle, apply it to QUBO problems. Finally, Section 4 considers the adaptive version of the Fixed-point Grover Search.

Code and Data Availability: Supplementary Qiskit code for this paper is available at [9].

Acknowledgment: We are grateful to Amazon Web Services and QLab for providing credits to access IonQ's QPUs. We also thank to Constantin Gonciulea for providing feedback on an earlier version of the work and QuForce for bringing the authors together for the project.

2. Quantum DictionariesBinary Optimization

Let $\mathbb{F}_2^n$ denote the space of length-n (binary) bit strings. A bit string $x = (x_0, x_1, \ldots, x_{n-1}) \in \mathbb{F}_2^n$ determines an n-qubit state $|x\rangle := |x_0\rangle |x_1\rangle \cdots |x_{n-1}\rangle$. A *quantum dictionary*, as introduced in [4], corresponding to a function, $F : \mathrm{dom}(F) \to \mathbb{F}_2^d$, where $\mathrm{dom}(F) \subseteq \mathbb{F}_2^n$, is the following quantum state on $n+d$ qubits:

$$|\textsc{qdict}(F)\rangle := \frac{1}{\sqrt{|\mathrm{dom}(F)|}} \sum_{x \in \mathrm{dom}(F)} |x\rangle |F(x)\rangle_d \,,$$

where the meaning of $|F(x)\rangle_d$ is the following: If $y \in \left[0, 2^d\right) \cap \mathbb{Z}$ and its binary representation if $y = \sum_{i=0}^{d-1} y_i 2^{d-1-i}$, then $|y\rangle_d := |y_0\rangle |y_1\rangle \ldots |y_{n-1}\rangle$, and for an arbitrary $y \in \mathbb{Z}$, $|y\rangle_d = |\bar{y}\rangle_d$, where $y \equiv \bar{y} \mod 2^d$ and $\bar{y} \in \left[0, 2^d\right) \cap \mathbb{Z}$. An integer-valued function $f : \mathbb{F}_2^n \to \mathbb{Z}$ canonically determines a quantum dictionary via first defining $F(x)$ to be the digits of $f(x)$, then setting, by a slight abuse of notation, $|\textsc{qdict}(f)\rangle = |\textsc{qdict}(F)\rangle$. Let us handle signs via the Two's complement convention. In particular, a binary number $y_0 y_1 \ldots y_{d-1}$ is negative exactly when $y_0 = 1$. In fact, every quantum dictionary can be realized in this way. We call a unitary, U, such that $U|0\rangle|0\rangle_d = |\text{QDICT}(F)\rangle$ an *encoder oracle* for the quantum dictionary $|\text{QDICT}(F)\rangle$.

Given a function $f : \mathbb{F}_2^n \to \mathbb{R}$, the associated (Unconstrained) Binary Optimization problem is the task of finding an element $x \in \mathbb{F}_2^n$ such that $f(x)$ is maximal. Note that every binary function is polynomial, which can be seen by simple dimension count. Since many interesting Binary Optimization problems, such as finding maximal graph cuts or the Max 2-SAT problems, are quadratic, most of the contemporary research centers around Quadratic Unconstrained Binary Optimization (QUBO) problems. The first main contribution of the paper is an oracle design for QUBO problems. More concretely, we construct encoder oracle of (quadratic) quantum dictionaries. These oracles have applications, for example, in Grover type algorithms and threshold QAOA [6]. While designs for such oracles have already existed, cf. [5], ours has better circuit depth, gate count, and CNOT count.

Remark 2.1. *One can also consider rational valued function, $f : \mathrm{dom}(f) \to \mathbb{Q}$, and encode them in a double dictionary*

$$|\text{QDICT}_{\mathbb{Q}}(f)\rangle := \frac{1}{\sqrt{|\mathrm{dom}(f)|}} \sum_{x \in \mathrm{dom}(f)} |x\rangle |f_1(x)\rangle_{d_1} |f_2(x)\rangle_{d_2},$$

where $f_1(x)$ is the integer part of $f(x)$ and $f_2(x)$ is the fractional part, multiplied by 2^{d_2}, for some large enough d_2. In other words, $|\text{QDICT}_{\mathbb{Q}}(f)\rangle = |\text{QDICT}(2^{d_2} f)\rangle$.

We construct encoding oracles for integer-valued quantum dictionaries in two steps. First, we outline a modified version of the encoding operator given in [5] that is convenient to encode quadratic polynomials. Then we express quadratic polynomials in a basis of functions that can be more efficiently encoded.

2.1. **Quadratic encoder.** Let $I \subseteq \{0, 1, \ldots, n-1\}$ and $p_I(x) := x_{i_1} x_{i_2} \cdots x_{i_j}$ be an arbitrary monomial and consider a quantum circuit with $n + d$ qubits. Following [5], we construct an oracle that sends $|x\rangle|0\rangle_d$ to $|x\rangle|p_I(x)\rangle_d$, for any $x \in \mathbb{F}_2^n$.

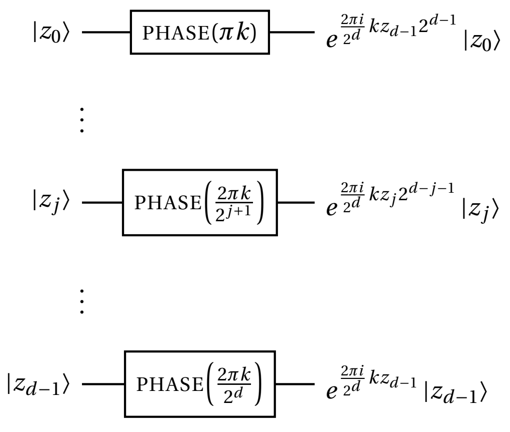
Let us make two definitions: Let QFT_d be the Quantum Fourier Transform on d qubits, that is for any $y \in \mathbb{Z}$, we have

$$\text{QFT}_d |y\rangle_d = \frac{1}{\sqrt{2^d}} \sum_{z=0}^{2^d-1} e^{\frac{2\pi i}{2^d} yz} |z\rangle_d .$$

Then

$$\text{QFT}_d^\dagger |z\rangle_d = \frac{1}{\sqrt{2^d}} \sum_{y'=0}^{2^d-1} e^{-\frac{2\pi i}{2^d} zy'} |y'\rangle_d .$$

Now let $\mathscr{P}_d(k)$ be the following d-qubit gate


$|z_0\rangle$
PHASE(πk)
$e^{\frac{2\pi i}{2^d} k z_{d-1} 2^{d-1}} |z_0\rangle$
$|z_j\rangle$
PHASE$\left(\frac{2\pi k}{2^{j+1}}\right)$
$e^{\frac{2\pi i}{2^d} k z_j 2^{d-j-1}} |z_j\rangle$
$|z_{d-1}\rangle$
PHASE$\left(\frac{2\pi k}{2^d}\right)$
$e^{\frac{2\pi i}{2^d} k z_{d-1}} |z_{d-1}\rangle$

Thus $\mathscr{P}_d(k)\,|z\rangle_d = e^{\frac{2\pi k z}{2^d} i}\,|z\rangle_d$, up to a global ($z$-independent) phase.

Now we can prove a well-known lemma.

Lemma 2.2. *For all $y, k \in \mathbb{Z}$ we have*

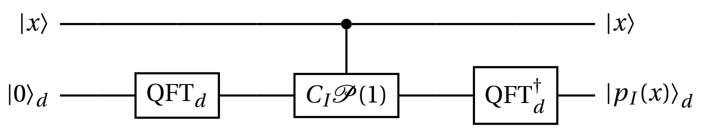
$$\mathrm{QFT}_d^\dagger \circ \mathscr{P}_d(k) \circ \mathrm{QFT}_d\,|y\rangle_d = |y+k\rangle_d\,. \tag{2.1}$$

Proof. First we compute

$$\begin{aligned}\mathscr{P}_d(k)\circ\mathrm{QFT}_d\,|y\rangle_d &= \mathscr{P}(k)\left(\frac{1}{\sqrt{2^d}}\sum_{z=0}^{2^d-1} e^{\frac{2\pi i}{2^d}yz}\,|z\rangle_d\right)\\ &= \frac{1}{\sqrt{2^d}}\sum_{z=0}^{2^d-1} e^{\frac{2\pi i}{2^d}yz}\,\mathscr{P}(k)\,|z\rangle_d\\ &= \frac{1}{\sqrt{2^d}}\sum_{z=0}^{2^d-1} e^{\frac{2\pi i}{2^d}(y+k)z}\,|z\rangle_d\\ &= \mathrm{QFT}_d\,|y+k\rangle_d\,,\end{aligned}$$

which is equivalent to equation (2.1). $\square$

Recall that $p_I(x) = x_{i_1}x_{i_2}\cdots x_{i_j}$. Now by Lemma 2.2 it is immediate that



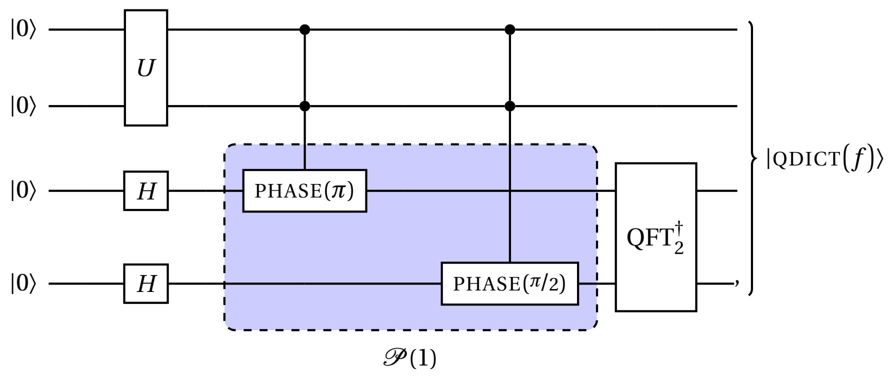
where C_I means control by the qubits $i_1, i_2, \ldots i_j$.

Finally, to create the full quantum dictionary, $|\mathrm{QDICT}(f)\rangle$, we need to add an oracle, call U, for which we have

$$U\,|0\rangle = \frac{1}{\sqrt{|\mathrm{dom}(f)|}}\sum_{x\in\mathrm{dom}(f)} |x\rangle\,,$$

to the beginning of the circuit. If $\mathrm{dom}(f) = \mathbb{F}_2^n$, then $U = H^{\otimes n}$.

Example 2.3. *Let $n = d = 2$ and $f(x) = x_0x_1$. Since $\mathrm{QFT}_d\,|0\rangle_d = H^{\otimes d}\,|0\rangle_d$, the oracle takes the form*

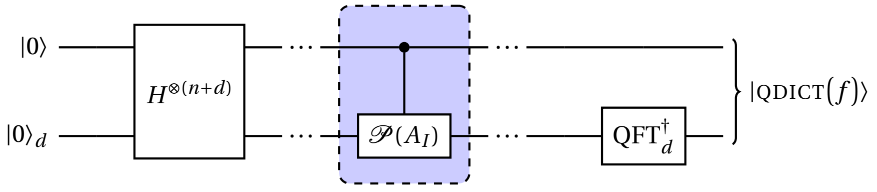


For the rest of the paper we assume that $\mathrm{dom}(f) = \mathbb{F}_2^n$ and use $U = H^{\otimes n}$. Furthermore, since $\mathrm{QFT}_d\,|0\rangle_d = H^{\otimes d}\,|0\rangle_d$, we can replace QFT_d with $H^{\otimes d}$ in the oracle, as the latter has depth 1 and uses only single-qubit gates.

Let $\mathbb{P}_n$ be the power set of $\{0,1,\ldots,n-1\}$. Now, for an arbitrary polynomial,

$$f(x) = \sum_{I\in\mathbb{P}_n} A_I x^I,$$

and $d \in \mathbb{Z}_+$ large enough so that all values of f can be digitized on d bits, we have that



one for each $I = \{i_1, i_2, \ldots, i_{|I|}\} \in \mathbb{P}_n$, controlled by qubits $x_{i_1}, x_{i_2}, \ldots, x_{i_{|I|}}$

Remark 2.4. *Let us briefly explain why we use this method of quantum addition.*

First, this method allows for a straightforward controlled addition through controlled phase gates, which are easy to implement and highly parallelizable as up to d terms can be digitized in parallel. Second, arbitrary integer values can be added or subtracted with the same type of circuit and with gate-complexities. In fact, note that $P_d(k)$ makes sense for any $k \in \mathbb{R}$, not just for integers. A little more computation shows that if $k \notin \mathbb{Z}$ so that $k_i \in \mathbb{Z}$ and $k_f \in (0,1)$ are its integer and fractional parts, then

$$\mathrm{QFT}_d^\dagger \circ \mathscr{P}(k) \circ \mathrm{QFT}_d |y\rangle_d = \sum_{z=0}^{2^d-1} \frac{e^{2\pi \tilde{k} i} - 1}{e^{\frac{2\pi i}{2^d}(y-z+k)} - 1} |z\rangle_d .$$

This yields a Fejér type distribution on the bit strings with the probability of measuring either $y + k_i$ or $y + k_i + 1$ being at least $\frac{8}{\pi^2} > 0.81$; cf. [5, Appendix B.2.].

Thus, the same oracle (but with fractional coefficients) can, to some extent, be used for functions with potentially noninteger coefficients. We postpone the further discussion of this issue to a later paper.

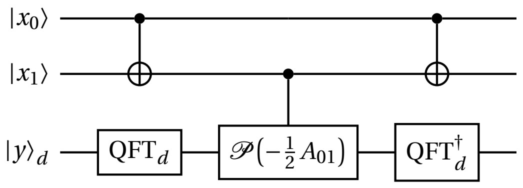
2.2. **New basis for quadratic polynomials and the new oracle design.**

We motivate the idea of the new basis by outlining it in the $n = 2$ case.

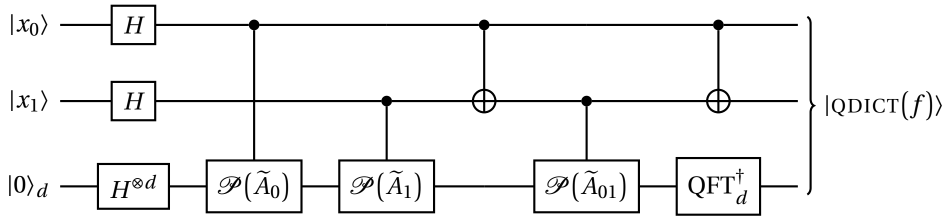
Note first that, since $x_i^2 = x_i$ for binary variables, we have that $x_0x_1 = \frac{1}{2}\big(x_0 + x_1 - (x_0 - x_1)^2\big)$. Now $(x_0 - x_1)^2$ is also a binary variable, in fact, $(x_0 - x_1)^2 = x_0 \text{ XOR } x_1$. Now let f be a generic polynomial, $f(x_0, x_1) = A_\emptyset + A_0x_0 + A_1x_1 + A_{01}x_0x_1$. Since we are interested in finding the maximum of f, we can assume, without any loss of generality, that $f(0,0) = A_\emptyset = 0$. By generic, we mean that $0 \notin \{A_0, A_1, A_{01}\}$. Using d digits, we need $2d$ CNOT gates for each linear terms and $6d$ CNOT gates for the quadratic term, thus a total of $10d$ CNOT gates (not counting the CNOT gates in $\mathrm{QFT}_d^\dagger$). However, we can rewrite f as

$$\begin{aligned} f(x_0, x_1) &= A_0x_0 + A_1x_1 + A_{01}x_0x_1 \\ &= \overbrace{\big(A_0 + \tfrac{1}{2}A_{01}\big)}^{\widetilde{A}_0:=} x_0 + \overbrace{\big(A_1 + \tfrac{1}{2}A_{01}\big)}^{\widetilde{A}_1:=} x_1 + \overbrace{\big(-\tfrac{1}{2}A_{01}\big)}^{\widetilde{A}_{01}:=} (x_0 - x_1)^2, \end{aligned}$$

and observe that $(x_0 - x_1)^2 = x_0 \mathrm{XOR} x_1$. Since the effect of a CNOT gate on the target qubit is exactly the XOR gate, we get that the addition of the term $-\frac{1}{2}A_{01}(x_0 - x_1)^2$ can be implemented as



which now has CNOT count only $2+2d$. Thus the whole oracle (for arbitrary f) can be realized as



$|x_0\rangle$
$|x_1\rangle$
$|0\rangle_d$
H
H
$H^{\otimes d}$
$\mathscr{P}(\widetilde{A}_0)$
$\mathscr{P}(\widetilde{A}_1)$
$\mathscr{P}(\widetilde{A}_{01})$
$\mathrm{QFT}_d^\dagger$
$|\mathrm{QDICT}(f)\rangle$

making the new CNOT count for the whole oracle to be (at most) $2+6d$, again, not counting the CNOT gates in $\mathrm{QFT}_d^\dagger$. In fact, the only time the two counts equal is when $A_0 = A_1 = 0$, $A_{01} \neq 0$, and $d = 1$.

A general, n-bit, quadratic polynomial, can be given by a symmetric, real, n-by-n matrix, Q via

$$f(x_0, x_1, \ldots, x_{n-1}) = x^T Q x = \sum_{i=0}^{n-1} Q_{ii} x_i + 2 \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} Q_{ij} x_i x_j.$$

If we set $q_i := \sum_{j=0}^{n-1} Q_{ij}$ is the sum of the i^{th} row, then

$$f(x_0, x_1, \ldots, x_{n-1}) = \sum_{i=0}^{n-1} q_i x_i - \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} Q_{ij} (x_i - x_j)^2.$$

The addition of a quadratic term, $x_i x_j$, in the oracle design of [5] is implemented via a 2-controlled $\mathscr{P}$-gate, whereas the addition of the term, $(x_i - x_j)^2$, in our construction is implemented via a 1-controlled $\mathscr{P}$-gate and 2 additional CNOT gates, which is generally more economical both in terms of circuit depth and entangling gate counts. In particular, when decomposed to single-qubit gates and CNOT gates, the former design needs $8d$ for each 2-controlled $\mathscr{P}$-gate, while the latter requires only $2+2d$ CNOT gates. Thus the total CNOT counts for a generic quadratic polynomial are $n(2d) + \frac{n(n-1)}{2}(8d) = (4n^2 - 2n)d$ and $n(2d) + \frac{n(n-1)}{2}(2+2d) = (d+1)n^2 + (d-1)n$, respectively, which is an approximately fourfold improvement. Since up to d terms can be digitized in parallel, the time complexity of the oracle is $O(n^2)$, as long as $d = o(n)$.

Remark 2.5. *In the case a single, quadratic monomial and $d = 1$, there is no advantage; in fact, in this case, our construction is just the well-known 2-controlled phase gate from* [10, Figure 4.8]. *Using this observation and a bit more work, one can also show that our construction is never worse (in terms of gate count, CNOT count, or gate circuit depth), than that of* [5].

A further generalization of this construction to higher degree polynomials is also currently being prepared by the authors.

2.3. **Experimental testing of the encoder oracle testing on IonQ's Quantum Computers:** The above marker oracle was tested on IonQ's Aria 2 QPU. We used 9 qubits (5-bit QUBO with 4 ancillas) with 5000 shots. The matrix of the quadratic form was

$$Q = \begin{pmatrix} 2 & -1 & 0 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & -1 \\ -1 & 0 & 0 & 2 & 0 \\ 0 & 0 & -1 & 0 & 2 \end{pmatrix},$$

with threshold $y = 5$ that is, in fact, the maximum. Out of the $2^5 = 32$ configurations only 3 equal to the threshold, thus $\lambda = \frac{3}{32}$. The oracle was decomposed to 1-qubit and CNOT gates, yielding a circuit depth of 47 and a CNOT count of 66. The experimental results are given in Table 1.

x	$f(x)$	$y_0 = 0$	$y_0 = 1$	x	$f(x)$	$y_0 = 0$	$y_0 = 1$
00000	0	3.44%	0.74%	00001	2	1.68%	0.66%
10000	2	1.68%	0.36%	10001	4	1.06%	0.26%
10000	1	3.18%	0.52%	10001	3	1.04%	0.30%
10000	1	4.08%	0.86%	10001	3	2.02%	0.48%
10000	2	2.16%	0.68%	10001	2	3.90%	0.78%
10100	4	2.02%	0.40%	10101	4	2.30%	0.36%
01100	3	1.34%	0.36%	01101	3	3.42%	0.58%
11100	3	2.90%	0.32%	11101	3	4.54%	0.34%
00010	2	1.34%	0.36%	00011	4	1.14%	0.38%
10010	2	5.26%	0.74%	10011	4	2.82%	0.44%
01010	3	2.52%	0.34%	01011	5	0.52%	1.16%
11010	1	2.84%	0.38%	11011	3	1.60%	0.52%
00110	4	1.58%	0.20%	00111	4	1.88%	0.26%
10110	4	2.94%	0.88%	10111	4	5.94%	1.74%
01110	5	1.06%	1.52%	01111	5	0.94%	2.46%
11110	3	2.20%	0.36%	11111	3	4.18%	0.74%

TABLE 1. Test results with threshold 5; values below 5 should not be marked ($y_0 = 0$) and above or equal to 5 should be marked ($y_0 = 1$). The green numbers are the percentages of the given state being measured with the correct marking, while the red ones are the percentages of incorrect markings.

While the oracle is 2–5 times more likely to correctly mark states, further experiments and simulations (not listed here) show that contemporary QPUs are too noisy to run our circuits without error correction.

3. FIXED-POINT GROVER SEARCH FOR QUBO

Fixed-point Grover Search (FPGS) [12] is a variant of Grover's search algorithm that retains the original version's query complexity while does not suffer from the soufflé problem (more on these below). In this section we introduce the algorithm, show how our oracle design from Section 2.2 can be used to implement FPGS for QUBO problems, and argue that this method is better suited for an adaptive optimizer algorithm than the original.

As opposed to Grover's algorithm, where the only input is the set of *good* (or *target*) configurations, $T \subset \mathbb{F}_2^n$, the FPGS algorithm requires an additional one, that can be chosen to be either the target probability/amplitude or the query complexity. The target probability is the probability of finding the system in a good state after running the circuit. The price of this flexibility (and of the elimination of half of the soufflé problem) is that one needs to implement not only a pair of oracles, but two families of them, call $S_s(\alpha)$ and $S_t(\beta)$ (where the subscripts refer to the *start* and *target* states, and α, β are real parameters), with the following properties: let us fix gauge so that, for some $\lambda \in (0,1)$, we can write

$$|s\rangle = \sqrt{\lambda}\,|t\rangle + \sqrt{1-\lambda}\,|\overline{t}\rangle\,,$$
$$|t\rangle = \sqrt{\lambda}\,|s\rangle + \sqrt{1-\lambda}\,|\overline{s}\rangle\,,$$

where $\langle t|\overline{t}\rangle = \langle s|\overline{s}\rangle = 0$. Then

$$S_s(\alpha)\big(A\,|s\rangle + B\,|\overline{s}\rangle\big) = e^{i\alpha}A\,|s\rangle + B\,|\overline{s}\rangle\,,$$
$$S_t(\beta)\big(D\,|t\rangle + C\,|\overline{t}\rangle\big) = e^{i\beta}C\,|t\rangle + D\,|\overline{t}\rangle\,.$$

Let $G(\alpha,\beta) := S_s(\alpha)S_t(\beta)$. Once in possession of such oracles and a target probability $P \in (0,1)$, the main result of [12] can be summarized as follows: for any $\mu \in (0,\lambda]$ large enough, there exists $l = l(P,\mu) \in \mathbb{Z}_+$ and $\delta = \delta_{P,\mu} \in (0,1)$, such that if, for all $j \in \{1,\dots,l\}$, we set

$$\alpha_j := 2\,\mathrm{arccot}\Big(\tan\Big(\tfrac{2\pi j}{2l+1}\Big)\tanh\Big(\tfrac{\mathrm{arccosh}(1/\delta)}{2l+1}\Big)\Big),$$

then

$$P_{\text{success}} := |\langle t|G(\alpha_l,\alpha_1)\cdots G(\alpha_2,\alpha_{l-1})\circ G(\alpha_1,\alpha_l)|s\rangle|^2 \geqslant P.$$

Moreover, l,δ can be explicitly computed (see next section).

We implement $S_s(\alpha)$ and $S_t(\beta)$ in straightforward ways. If U_s is the state preparation oracle, that is, $|s\rangle = U_s\,|0\rangle$, and $\mathrm{MCP}_n(\alpha)$ is the $(n-1)$-controlled phase gate on n qubits, then

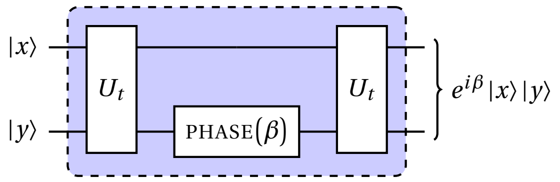
$$S_s(\alpha) = U_s \circ X^{\otimes n} \circ \mathrm{MCP}_n(\alpha) \circ X^{\otimes n} \circ U_s^\dagger, \tag{3.1}$$

works. Note that MCP_n can be implemented with $O(n^2)$ CNOT gates and circuit depth, and no ancillas; cf. [2].

In general, the implementation of $S_t(\beta)$ is case specific. In the case of an instance of a QUBO problem, let $T := \{x \in \mathbb{F}_2^n | f(x) < 0\}$. Now the results of Sections 2.1 and 2.2 can be immediately used to get an oracle, U_t, on $n+1$ qubits, such that for all $x \in \mathbb{F}_2^n$ and $y \in \mathbb{F}_2$, we have

$$U\,|x\rangle\,|y\rangle = \begin{cases} |x\rangle\,|y \oplus 1\rangle\,, & \text{if } x \in T, \\ |x\rangle\,|y\rangle\,, & \text{if } x \notin T, \end{cases}$$

where the last qubit, $|y\rangle$, is the first ancilla of the original oracle. Now



$S_t(\beta)$

works for $S_t(\beta)$, if the $(n+1)^{\text{st}}$ qubits is a clean ancilla kept in $|0\rangle$ before and after the usage of the oracle. Since QFT can be implemented on d qubits use $O(d^2)$ gates and depth (and $O(d\ln(d))$ for the approximate QFT), the (worst case) CNOT count of U_t is $O(n^2)$, as long as $d = O(n)$.

Remark 3.1. *In the case when f is the cut function of a simple, unoriented, and undirected graph, with n vertices and m edges (in which case the matrix Q is the graph Laplacian), FPGS can be implemented on $n + O\big(\log(m)\big)$ qubits, with gate count being $O\big(m\log(m)\big)$, and circuit depth of $O(m)$.*

3.1. **Time complexity of the circuit.** In order to understand the time-complexity of FPGS for QUBO problems, we need to know three things:

(1) The query complexity, l.
(2) The complexity of the diffusion operator, $S_s(\alpha)$. Let us call this complexity C_s.
(3) The complexity of the operator, $S_t\big(\beta\big)$. Let us call this complexity C_t.

The time complexity is then $l(C_s + C_t)$. From [12], we know that $l \approx \frac{\ln(2/\delta)}{\sqrt{\mu}}$ is sharp. By equation (3.1), we see that C_s is the same as the time complexity of the $\mathrm{MCP}_n(\alpha)$ gate. By [2], we get that $C_s \leqslant O\big(n^2\big)$ (and this is applies to both single qubit and CNOT gate counts). Finally, by the argument of the previous section, $C_t = O\big(n^2\big)$ as well. This yields, for a fixed δ, that the total time complexity if $O\Big(\mu^{-\frac{1}{2}} n^2\Big)$. Note that we still have the choice of the number $\mu \in (0, \lambda]$ that we discuss in the next section.

Remark 3.2. *In the case of maximal graph cuts, we have already seen in Remark 3.1 we have already seen that the complexity becomes $C_t = O(m) \leqslant O\big(n^2\big)$, and thus the complexity of FPGS is $O\Big(\mu^{-\frac{1}{2}}\big(n^2 + m\ln(m)\big)\Big)$.*

4. Adaptive Search

As mentioned above, the choice of μ is problematic; it needs to be at most $\lambda \in (0,1)$, the ratio of marked configurations to all configurations (or, more precisely, the tunneling amplitude between the initial state and the target state), but λ is not known. This is where the fixed-point nature of FPGS becomes helpful: μ is only used to find l, and the smaller μ is, the larger l gets. At the same time, FPGS cannot be "overcooked", thus we can increase l without dropping the probability of success below a P which, in turn, is controlled by δ. Hence, in lieu of the knowledge of μ, one can do the following: pick $\epsilon \in \mathbb{R}_+$ and $l_0 \in \mathbb{Z}_+$ and run FPGS with $l = l_0, (1+\epsilon)l_0, (1+\epsilon)^2 l_0, \ldots$, until a configuration, $x \in \mathbb{F}_2^n$, with $f(x) < 0$ is found. If the necessary query complexity was $\tilde{l} \simeq (1+\epsilon)^N l_0$, then this iterative method is expected to take

$$\sum_{i=0}^{N} (1+\epsilon)^i l_0 = \frac{(1+\epsilon)^{N+1} - 1}{\epsilon} l_0 = \frac{(1+\epsilon)^{N+1} - 1}{\epsilon(1+\epsilon)^N} \tilde{l} = O_\epsilon\big(\tilde{l}\big),$$

queries, thus have the same big-O complexity. This idea was, in fact, studied by Li et al. in [8] and they proved that if one has no prior information about λ and choses to follow the above schedule, than the choices of $\delta \approx 0.5659$ and $\epsilon \approx 0.523$ yields a total query of $l_{\text{total}} \approx \frac{5.643}{\sqrt{\lambda}}$ [8, Theorem 1], and it is best known schedule for ratio oblivious, Grover-type searches [8, Table 1].

For any given $y \in \mathbb{Z}$, we can repeat the algorithm with the polynomial $y - 1 - f(x)$, making the set of marked states to be $T_y := \{x \in \mathbb{F}_2^n | f(x) \geqslant y\}$. Since our goal is not only to find configurations, x, with values, $f(x)$, above a certain threshold, but rather to find configurations with as high values as possible, we regard y as another parameter of the FPGS for QUBO problem.

Now let us assume that we are given an n-dimensional instance of a QUBO problem, f and also a function that describes the stopping condition

$$T_f : \mathbb{R}_+ \times \mathbb{Z}, \big(t, y\big) \mapsto T_f\big(t, y\big) \in \{\text{True}, \text{False}\},$$

where t is the elapsed time and y is the higher known (found) value. Let $\mathrm{G}(y,l)$ be the FPGS for QUBO with threshold y, $\delta = 0.5659$, and l queries. With the above arguments in mind, we propose an adaptive, quantum–classical hybrid method, shown on Figure 1 below.

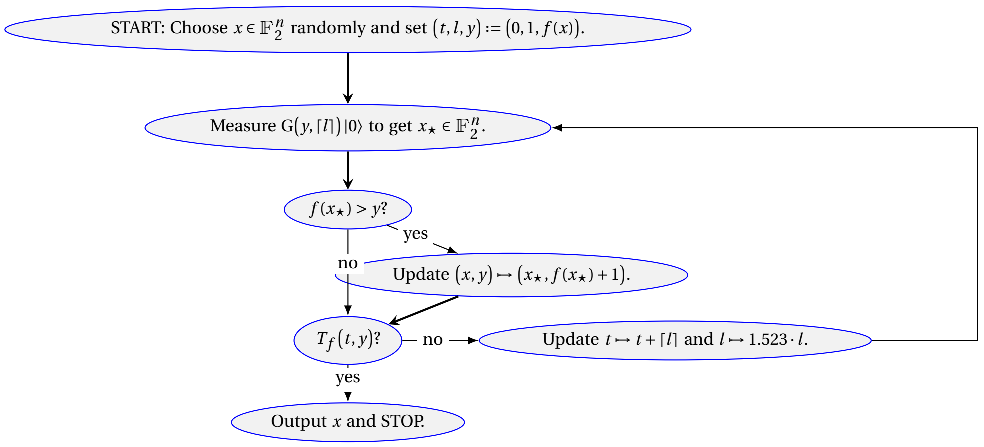

START: Choose $x \in \mathbb{F}_2^n$ randomly and set $(t,l,y) := (0,1,f(x))$.
Measure $\mathrm{G}(y,\lceil l\rceil)\,|0\rangle$ to get $x_\star \in \mathbb{F}_2^n$.
$f(x_\star) > y$?
yes
no
Update $(x,y) \mapsto (x_\star, f(x_\star)+1)$.
$T_f(t,y)$?
no
Update $t \mapsto t + \lceil l\rceil$ and $l \mapsto 1.523 \cdot l$.
yes
Output x and STOP.

FIGURE 1. Fixed-point Grover Adaptive Search for QUBO

5. CONCLUSION

We constructed a marker oracle for QUBO problems with improved complexities to the previously known known designs. These oracles are expected to be useful for both Grover-, and QAOA-type quantum circuits. We also studied adaptive optimization methods, using our constructions in conjunction with FPGS.

We conjecture that our Adaptive FPGS for QUBO has a better chance to have provable and (and stronger) performance guarantees, when compared to the original Grover Adaptive Search of [5] (as one only needs to find large enough, but not the exact, l) or QAOA (as the number of parameters is lower and their role is more straightforward).

REFERENCES

[1] Andreas Bärtschi and Stephan Eidenbenz, *Grover Mixers for QAOA: Shifting Complexity from Mixer Design to State Preparation*, 2020 IEEE International Conference on Quantum Computing and Engineering (QCE), 2020, pp. 72–82. ↑1

[2] Adenilton J. da Silva and Daniel K. Park, *Linear-depth quantum circuits for multiqubit controlled gates*, Phys. Rev. A **106** (2022Oct), 042602. ↑8, 9

[3] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann, *A quantum approximate optimization algorithm*, arXiv preprint arXiv:1411.4028 (2014). ↑1

[4] Austin Gilliam, Charlene Venci, Sreraman Muralidharan, Vitaliy Dorum, Eric May, Rajesh Narasimhan, and Constantin Gonciulea, *Foundational patterns for efficient quantum computing* (2021). ↑2

[5] Austin Gilliam, Stefan Woerner, and Constantin Gonciulea, *Grover Adaptive Search for Constrained Polynomial Binary Optimization*, Quantum **5** (2021), 428. ↑1, 2, 3, 5, 6, 10

[6] John Golden, Andreas Bärtschi, Daniel O'Malley, and Stephan Eidenbenz, *Threshold-Based Quantum Optimization*, 2021 ieee international conference on quantum computing and engineering (qce), 2021, pp. 137–147. ↑1, 2, 3

[7] Lov K. Grover, *Quantum mechanics helps in searching for a needle in a haystack*, Phys. Rev. Lett. **79** (1997Jul), 325–328. ↑1

[8] Tan Li, Shuo Zhang, Xiang-Qun Fu, Xiang Wang, Yang Wang, Jie Lin, and Wan-Su Bao, *Quantum search for unknown number of target items by hybridizing fixed-point method with trail-and-error method*, Chinese Physics B **28** (2019nov), no. 12, 120301. ↑1, 2, 9

[9] Ákos Nagy, *GitHub repository for the implementation of the Fixed-point Grover Search circuit for Quadratic Binary Optimization Problems*. Available at github.com/akos-nagy/Grover_FPS_for_QUBO. ↑2

[10] Nielsen, Michael A. and Chuang, Isaac L., *Quantum Computation and Quantum Information: 10th Anniversary Edition*, Cambridge University Press, 2010. ↑6

[11] Márió Szegedy, *What do QAOA energies reveal about graphs?*, arXiv:1912.12277 (2019). ↑1

[12] Theodore J. Yoder, Guang Hao Low, and Isaac L. Chuang, *Fixed-point quantum search with an optimal number of queries*, Phys. Rev. Lett. **113** (2014), 210501. ↑1, 2, 7, 8, 9

(Ákos Nagy) BEIT CANADA, TORONTO, ONTARIO
Email address: akos@beit.tech
URL: akosnagy.com

(Jaime Park) VANDERBILT UNIVERSITY, NASHVILLE, TENNESSEE
Email address: jaime.s.park@vanderbilt.edu

(Cindy Zhang)

(Atithi Acharya)

(Alex Khan) ALIGNED IT, LLC., BALTIMORE, MARYLAND
Email address: alex.khan@alignedit.com
URL: alignedit.com