IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 7 August 2023, accepted 2 October 2023, date of publication 12 October 2023,
date of current version 19 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3324253

APPLIED RESEARCH

Multiple Query Optimization Using a Gate-Based Quantum Computer

TOBIAS FANKHAUSER[1], MARC E. SOLÈR[2], RUDOLF MARCEL FÜCHSLIN[3,4], AND KURT STOCKINGER[3]

[1]Institute of Computer Science, University of Zurich, 8050 Zürich, Switzerland
[2]Department of Informatics, University of St. Gallen, 9000 St. Gallen, Switzerland
[3]School of Engineering, ZHAW Zurich University of Applied Sciences, 8401 Winterthur, Switzerland
[4]European Centre for Living Technology, 30123 Venice, Italy

Corresponding author: Kurt Stockinger (Kurt.Stockinger@zhaw.ch)

This work was supported by Zurich University of Applied Sciences.

ABSTRACT Quantum computing promises to solve difficult optimization problems in chemistry, physics and mathematics more efficiently than classical computers. However, it requires fault-tolerant quantum computers with millions of qubits; a technological challenge still not mastered by engineers. To lower the barrier, hybrid algorithms combining classical and quantum computers are used, where quantum computing is only used for those parts of computation that cannot be solved efficiently otherwise. In this paper, we tackle the multiple query optimization problem (MQO), an important NP-hard problem in database research. We present an implementation based on a scheme called quantum approximate optimization algorithm to solve the MQO on a gate-based quantum computer. We perform a detailed experimental evaluation of our implementation and compare its performance against a competing approach that employs a quantum annealer – another type of quantum computer. Our implementation shows a qubit efficiency of close to 99%, which is almost a factor of 2 higher than the state-of-the-art implementation. We emphasize that the problems we can solve with current gate-based quantum technology are fairly small and might not seem practical yet compared to state-of-the-art classical query optimizers. However, our experiments on using a hybrid approach of classical and quantum computing show that our implementation scales favourably with larger problem sizes. Hence, we conclude that our approach shows promising results for near-term quantum computers and thus sets the stage for a challenging avenue of novel database research.

INDEX TERMS Optimization, databases, multiple query optimization, quantum approximate optimization algorithm, experimental evaluation.

I. INTRODUCTION

In database research, various optimization problems have been formulated since the seventies [22], [35], [37], with the query optimization problem being a typical NP-hard representative. These problems, although extensively examined, become even more difficult as data processing becomes more complex [41]. Recent approaches to the query optimization problem have also become increasingly sophisticated, now employing deep learning methods [22], [28]. Simultaneously, progress in the construction of quantum computers has sparked new interest in applied quantum computing [4], [7], [13], [32], [38], whereas previously, the quantum computing community was mainly concerned with more theoretical studies due to the lack of practical quantum devices [30]. It is hoped that quantum computers can one day solve complex problems more efficiently than classical computers [30]. Recently, quantum computing has been applied for various problems such as optimizing transaction schedules [18] or financial portfolios [43]. Multiple query optimization appears to be an ideal "toy problem" in various respects: First, already small instances of the problem exhibit the structural features of more complex settings. Second, the problem is of actual practical relevance (which may even be growing). Third, the problem has also been studied extensively from a theoretical point of view.

The associate editor coordinating the review of this manuscript and approving it for publication was Genoveffa Tortora.

Present quantum computing architectures can be divided into two large classes: gate-based approaches and adiabatic quantum computation (AQO, see [15]). AQO is based on the adiabatic theorem of quantum mechanics, which states that a quantum system remains in its ground state under (i.e. the state with the lowest energy) suitable conditions, even if the system is changed over time (more about this later). At first glance, AQO looks very promising: As we will discuss, quantum systems can be set up such that their ground state is equivalent to the solution of a combinatorial optimization problem [26]. All we have to do is to set up a quantum system with a known and easy-to-prepare ground state. Then, we gradually modify the system such that at the end of the modification process, the system's ground state can be identified with the solution to the problem under consideration.

Quantum computers by D-Wave use such an approach [9]. Presently, there is a debate on whether such systems lead to a quantum advantage. One reason for these concerns is that ''suitable conditions'' include a sufficiently slow change of the quantum system, which may grow exponentially with problem size. There is a method, quantum approximate optimization algorithms (QAOA), which is structurally similar to AQO, but is designed to be run on a gate-based architecture. There is considerable hope ([27]) that one can realize a quantum advantage for some large classes of problems. Whereas [27] focuses on the well-researched MaxCut-problem, this work analyzes a more practical question, namely multiple query optimization.

We point out that gate-based quantum computers differ fundamentally from adiabatic quantum computations. Many algorithms for gate-based architectures (Grover, Shor, QAOA) are interference-based. From a physical perspective, they exploit the fact that quantum states can interfere. Appropriate algorithms enforce the target state to interfere positively, whereas all other states interfere destructively [40]). Adiabatic computation exploits the adiabatic theorem of quantum mechanics by implementing a sufficiently slowly varied energy landscape. Besides direct comparisons for different algorithms, the evaluation of performance boundaries (which means the prediction of which problems can be reliably solved on a given architecture) may turn out to be simpler for gate-based architectures.

A. MULTIPLE QUERY OPTIMIZATION

The Multiple Query Optimization problem (MQO) is a generalization of the query optimization problem [37]. MQO is known to be an NP-hard problem [36]. The goal is to minimize the total cost of executing a series of queries against a database by exploiting shared intermediate results [37]. MQO has been approached by classical methods such as the shortest-path algorithms [37]. Since then, integer linear programming [12] and genetic algorithms [5] have been proposed, improving prior methods. More recent approaches employ dynamic programming while considering multiple cost metrics [42]. In the case of the more general class of query optimization problems, learning algorithms have successfully been applied [39]. Modern approaches extend this approach with the usage of deep learning methods [22], [28].

We illustrate the problem with an example. Assume three tables T_1, T_2, T_3 containing information about books and their authors. Table T_1 contains pairs of entries of the form (a_i, r_i). The variable a_i gives an author's name, r_i a rating of the author's style. Table T_2 contains triplets of values (t_i, a_i, s_i). Thereby, t_i is the title of a book written by a_i, and s_i is again a rating but related to the content of the book with title t_i. Finally, T_3 contains entries of the form t_i, n_i with n_i the book ID of title t_i. We are looking for the book IDs of all books with a combined rating $r_i + s_i > v$ for some constant v, and we formulate this condition with a query $\sigma_{r+s>v}$. We assume the table T_1 to be small, and tables T_2 and T_3 to be large. The table $T_1 \bowtie T_2$ contains entries of the form (a_i, r_i, t_i, s_i) and is, after the application of the query $\sigma_{r+s>c}$ probably rather small. It is reasonable to expect the time needed for $(\sigma_{r+s>v}(T_1 \bowtie T_2)) \bowtie T_3$ to be shorter than for $(\sigma_{r+s>v}(T_1 \bowtie (T_2 \bowtie T_3))$, because $(T_2 \bowtie T_3)$ requires joining two large ltables.

The model studied here considers a series of queries $q_1, \ldots, q_Q$, where each query has a fixed number P of plans, uniquely identified as $p_{i1}, \ldots, p_{iP}$ for query i. Each plan p_{ij} has associated a cost c_{ij}. Finally, we denote savings between pairwise plans p_{ik} and p_{jl} as s_{ikjl}. These savings occur due to sharing of intermediate results between query plans.

In an MQP, for every query q_i, exactly one plan $p_{i\pi(i)}$ has to be selected. Formally, the problem consists in finding a function $\pi(i)$ such that Y, given by

$$Y = \sum_{i=1}^{Q} c_{i\pi(i)} - \sum_{i,j=1}^{Q} s_{i\pi(i)j\pi(j)}, \tag{1}$$

is minimized. The search space of this problem is significant as there are P possibilities to choose from for every query, resulting in P^Q possibilities.

B. CONTRIBUTIONS OF THE PAPER

The MQO scales exponentially with the number of query plans (as shown above). Recently, a quantum approach was suggested for the MQO by Trummer and Koch [41], utilizing a D-Wave quantum annealer [9], a particular type of quantum computer designed to run special optimization problems.

The work presented in this article builds on Trummer and Koch's approach, but utilizes a gate-based quantum computer, a more general type of quantum computer. Gate-based quantum computers can, in principle, run every quantum algorithm [2], [23], while quantum annealers in [41] are restricted to specific problems. The generality of gate-based quantum computers come with the cost of featuring significantly fewer qubits and are more error-prone than quantum annealers.

In summary, our contributions are the following:

1) We study algorithms to tackle MQO on a gate-based quantum computer. To the best of our knowledge, besides [34] this is among the first papers in the literature to address MQO with a gate-based quantum computer.
2) We run experiments on a quantum simulator and real quantum hardware, and analyze how the algorithms scale with the problem size.
3) We compare our work with a competing quantum query optimization approach and demonstrate that our gate-based implementation has a qubit efficiency[1] of close to 99%. This is almost a factor of 2 higher compared to the state-of-the-art quantum annealing-based implementation.
4) Even though our approach only solves small MQO problems, which might not seem practical yet due to the limitations of current quantum technology, it is a promising step for novel database research at the intersection of classical and quantum computing.

II. QUANTUM COMPUTING FOUNDATIONS

A. QUANTUM MECHANICS

Quantum algorithms are known for speeding up certain computations, such as integer factorization [31]. In principle, these algorithms achieve significant speedup - first, by not having to examine every branch in the search space sequentially, as by a classical brute force approach.[2] Instead, a quantum computer holds many branches during computation, which is possible because a quantum register can be understood as a superposition of basis states. Secondly, quantum algorithms exploit so-called *entanglement*.

In what follows, we present an overview over facts which are important for quantum computing. Although we motivate them, we mostly present results. A thorough study with proofs and detailed expositions must be taken from one of the multiple well-written textbooks.

Certain types of quantum systems, e.g. electrons or polarized photons, are represented in quantum mechanics as a normalized vector in a two-dimensional complex vector space, i.e. a two-dimensional complex Hilbert space. Such vectors, denoted by a so-called ket $|q\rangle$, are the superposition of two basis vectors $|0\rangle$ and $|1\rangle$:

$$|q\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{2}$$

whereby $\alpha, \beta \in \mathbb{C}$ and

$$|\alpha|^2 + |\beta|^2 = 1 \tag{3}$$

The vector $|q\rangle$ is called a qubit. Without delving into mathematical and physical details: qubits can be combined to form a so-called *quantum register*. Note: To keep the wording efficient, we do not distinguish between registers and their states, in the same sense as we do not distinguish between a qubit and its state. This convention enhances efficiency, but one could argue that it shadows the difference between hardware and the range of states this hardware can represent.

If one has n coupled qubits, the resulting quantum register is again a normalized element of a complex Hilbert space V, whereby this space has dimension 2^n. A basis state of this Hilbert space can be understood as a bit sequence $|x_1x_2\ldots x_n\rangle$ with $x_j \in \{0,1\}$. Mathematically, this coupling of single qubits is simply a tensor product $|x_1\rangle \otimes |x_2\rangle \otimes \ldots \otimes |x_n\rangle$. For national convenience, the rather clumsy tensor product is also written as $|x_1x_2\ldots x_n\rangle$. The bit sequence of the basis state encodes a natural number j. Therefore, the basis states are often just written as $|j\rangle$ with $j \in 0,\ldots,2^n-1$. Note that in this notation, x_1 is the most significant bit; however, sometimes, in a part of the literature, the order of bits is reversed.

From a mathematical point of view, there are many different choices of a basis for the vector space V. The basis $|j\rangle$ with j understood as a bit sequence is called the basis of the quantum register or the computational basis in the literature. In this work, basis vectors are always an element of the computational basis. We further note that in quantum mechanics, the terms basis vector and basis state are used interchangeably, whereby the latter is more commonly used.

A quantum register $|Q\rangle$ needs not to be in a basis state but can consist of a superposition of basis states:

$$|Q\rangle = \sum_{j=0}^{2^n-1} c_j\,|j\rangle \tag{4}$$

with $c_j \in \mathbb{C}$

$$\sum_{j=0}^{2^n-1} |c_j|^2 = 1. \tag{5}$$

A Hilbert space is (by definition) always equipped with an inner product. For an orthonormal basis, we have for the basis vectors $|j\rangle, |k\rangle$:

$$\langle j|k\rangle = \delta_{jk} = \begin{cases} 1, & \text{if } j = k, \\ 0, & \text{if } j \neq k. \end{cases} \tag{6}$$

For two general vectors

$$|X\rangle = \sum_{j=0}^{2^n-1} x_j\,|j\rangle\,, \quad |Y\rangle = \sum_{j=0}^{2^n-1} y_j\,|j\rangle\,, \tag{7}$$

the inner product is given by:

$$\langle X|Y\rangle = \sum_{j=0}^{2^n-1} \overline{x}_j y_j. \tag{8}$$

Note the complex conjugation denoted by the overlining. Furthermore, with eq. 5 and eq. 8 we get $\langle X|X\rangle = |\,|X\rangle\,|^2$.

[1] We define qubit efficiency as a measure of capacity utilization as a ratio between the number of qubits used for variable encoding and the number of qubits available. In the literature, this measure is qualitatively referred to as hardware efficiency as, e.g. in [6].

[2] The conception of a quantum computer evaluating the entire search space in parallel and selecting the best solution is an incorrect oversimplification [1].

One aspect of superposition is so-called *quantum parallelism*: One can understand an operation on a quantum register as the simultaneous operation on many basis states. This interpretation is sensible because quantum operations are (in a mathematical sense) linear operations in a vector space that is built up from basis states. Note here that there is no fundamental difference between a superposition of (basis) states and pure basis states: Whether a state is "simple" or a superposition is not an intrinsic property of the system but depends on the (external) choice of the basis.

Another aspect is that quantum registers can represent entangled states. An entangled state is characterized by the property that it cannot be understood as the tensor product of individual qubits. We exemplify this with two qubits $|q_j\rangle = \alpha_j |0\rangle + \beta_j |1\rangle$ for $j \in 1, 2$. Two qubits can be coupled by taking the tensor product (by multiplying out and observing that $|x\rangle \otimes |y\rangle \neq |y\rangle \otimes |x\rangle$ for $x \neq y$):

$$\begin{aligned} |q_1\rangle \otimes |q_2\rangle &= (\alpha_1 |0\rangle + \beta_1 |1\rangle) \otimes (\alpha_2 |0\rangle + \beta_2 |1\rangle) \\ &= \alpha_1\alpha_2 |0\rangle \otimes |0\rangle + \alpha_1\beta_2 |0\rangle \otimes |1\rangle \\ &\quad \beta_1\alpha_2 |1\rangle \otimes |0\rangle + \beta_1\beta_2 |1\rangle \otimes |1\rangle \\ &= \alpha_1\alpha_2 |00\rangle + \alpha_1\beta_2 |01\rangle \\ &\quad + \beta_1\alpha_2 |10\rangle + \beta_1\beta_2 |11\rangle . \end{aligned} \tag{9}$$

For a register such as $|Q\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle$ there is no possible choice for $\alpha_1, \alpha_2, \beta_1, \beta_2$ such that a $|Q\rangle$ could be written as a tensor product of two individual qubits. Such a state is called an entangled state. Note: Whereas one can always understand a classical register as a sequence of independent bits, a quantum register can be in a state that one cannot understand as a sequence of individual qubits.

Another way to understand this is to consider that eq. 3 implies that a single qubit is determined by three real numbers (remember that $\alpha, \beta \in \mathbb{C}$, therefore defined by two reals). Consequently, the tensor product of n qubits is defined by $3n$ real numbers. In contrast, (the content of) a general quantum register is given by $2^n - 1$ real numbers, s. eq. 5. For a two-qubit register, those states representing a combination (a "product") of individual qubits are spanned by six parameters, whereas general states are defined by seven reals. The exploitation of the existence of these "additional" states (besides the sequences of qubits) is another reason for the power of quantum computing.

Quantum computation, by gate-based computation or quantum annealers, is based on the dynamics of quantum states. Quantum dynamics (to be precise: its non-relativistic formulation) is given by the Schrödinger equation:

$$i\hbar \frac{\partial}{\partial t} |\Psi(t)\rangle = H |\Psi(t)\rangle \tag{10}$$

Thereby, H is a linear operator (with respect to some basis, a 2^n-dimensional matrix). The matrix H is called the Hamiltonian of the system and is related to the system's energy which results from the interaction of qubits with external fields or from couplings between qubits.

The matrix H has to satisfy two requirements for a physically sensible interpretation. First, H has a set of orthogonal eigenvectors, which form a (complete) basis of the Hilbert space V, and second, all eigenvalues are real numbers. If these two requirements are satisfied, H is called a Hermitian matrix. Note: because the matrix is of finite size 2^n, there are also 2^n eigenvalues (To simplify some arguments, we assume in what follows that these eigenvalues are mutually different. The treatment of systems with multiple identical eigenvalues can be done but requires some care.)

The eigenvalues of H are interpreted as the energies of the respective eigenstates: If $|\phi_j\rangle$ is an eigenstate of H and λ_j the eigenvalue of $|\phi_j\rangle$, then the system carries energy λ_j if it is in state $|\phi_j\rangle$. The state with the lowest energy (the ϕ_j with the smallest λ_j) is called the *ground state* of the system.

The property of the eigenstates of H to form a complete orthogonal basis (the so-called eigenbasis) has two consequences. The first one is: Each initial state $|\Psi(0)\rangle$ can be written as a superposition of eigenstates:

$$|\Psi(0)\rangle = \sum_{j=0}^{2^n-1} c_j |\phi_j\rangle . \tag{11}$$

The second consequence is that in the eigenbasis, H takes the form of a diagonal matrix with the λ_j on the diagonal. This means that in the eigenbasis, the Schrödinger equation 10 can be solved very easily:

$$|\Psi(t)\rangle = e^{-\frac{i}{\hbar}Ht} |\Psi(0)\rangle = \sum_{j=0}^{2^n-1} c_j e^{-\frac{i}{\hbar}\lambda_j t} |\phi_j\rangle . \tag{12}$$

Note that the exponential of the matrix H, $e^{-iHt/\hbar}$ can be taken literally. For unitary Hermitian matrices, exponentiation is well-defined (this by using the diagonalization used above).

Formally, if a (finite dimensional) matrix A is Hermitian, it satisfies the property that A is equal to its complex conjugate transpose: $A = A^\dagger$ (if a_{jk} is a matrix element of A, it holds for elements $a^\dagger_{jk}$ of $A^\dagger$: $a^\dagger_{jk} = \overline{a}_{kj}$). If A is a Hermitian matrix, it holds: $a_{jk} = \overline{a}_{kj}$.

We define another, in quantum mechanics, important class of matrices: A matrix U is called unitary, if it holds $UU^\dagger = U^\dagger U = \mathbf{1}$ (thereby, $\mathbf{1}$ represents the identity matrix). The property of unitarity has a geometric interpretation. From the definition of the inner product by eq. 8, it follows for a general matrix A and two vectors $|X\rangle, |Y\rangle$ that the inner product of their respective images $|X'\rangle = A|X\rangle, |Y'\rangle = A|Y\rangle$ is given by

$$\langle X'|Y'\rangle = (\langle X|A^\dagger)(A|Y\rangle) = \langle X|A^\dagger A|Y\rangle . \tag{13}$$

For a unitary matrix U, this implies

$$\langle X|U^\dagger U|X\rangle = \langle X|X\rangle = ||X\rangle|^2. \tag{14}$$

This means that the mapping resulting from the application of U does not change the length of vectors. Geometrically

understood, a unitary mapping U is a rotation in the Hilbert space V.

Importantly, the matrix exponential of a Hermitian matrix is a unitary matrix. If one analyzes the solution of the Schrödinger equation, eq. 12, we see that

$$|\Psi(t)\rangle = e^{-\frac{i}{\hbar}Ht}\,|\Psi(0)\rangle = U(t)\,|\Psi(0)\rangle \tag{15}$$

with

$$U(t) = e^{-\frac{i}{\hbar}Ht} \tag{16}$$

implies that the time development of a quantum mechanical system is a time-dependent linear and unitary operation in V.

Up to now, we assumed that H is stationary, which means independent of time. This assumption must be modified in studying quantum computing. From a mathematical perspective, solving the Schrödinger equation in the case of a time-dependent Hamiltonian requires, if done in full generality, sophisticated mathematics. However, we emphasize that in quantum computing, we, first, always work in finite-dimensional settings, and, second, the time-dependence of $H(t)$ can always be assumed to be "benign".

The time evolution described by Equation eq. 10 is not the only process of relevance in quantum computing. There are also so-called *measurements*. For our purposes, it is sufficient to define a measurement as a stochastic, non-linear operation that transforms a general quantum register $|Q\rangle$ into one, randomly chosen basis state. Formally, a measurement M is defined by (we use the notation of eq. 4):

$$M\,|Q\rangle = |j\rangle \qquad \text{with probability } |c_j|^2 \tag{17}$$

The probability of ending up after a measurement in a specific basis state depends on its respective coefficient in the register state on which one applies a measurement.

B. GATE-BASED QUANTUM COMPUTATION

Gate-based quantum computation can be interpreted as a series U_j of unitary transformations applied on some specifically prepared input state $|\Psi_0\rangle$.

The quantum computation is finished with a measurement M. This means: a quantum computation is a process that can be written in the form:

$$|j\rangle = MU_m \ldots U_1\,|\Psi_0\rangle \quad \text{with probability} \quad |c_j|^2 \tag{18}$$

assuming that

$$U_m \ldots U_1\,|\Psi_0\rangle = \sum_{j=0}^{2^n-1} c_j\,|j\rangle\,. \tag{19}$$

The sequence of m unitary operations is often visualized by a quantum circuit, see Figure 1, whereby the individual U_j are termed *quantum gates*. Note that in eq. 19, U_1 is the first operator applied on $|\Psi_0\rangle$; the temporal sequence of the operators U_j have to be read from right to left.

In Figure 1, the operations U_j are generic $n-$qubit operations. From a technological perspective it is very fortunate that a general quantum computation can be implemented by 1- and 2-qubit gates, see [11] and Figure 2.

Gate-based quantum computation can be understood in terms of eq. 10 by temporarily switching on and off different Hamiltonians (e.g. by varying external magnetic or electric fields). If these Hamiltonians act over a properly chosen time interval, they lead to desired unitary operations U_j according to eq. 16.

Note that Figure 2 does not represent a real circuit in the sense that components are connected by wires. A "quantum circuit" represents the temporal order of unitary operations. The "wires" relate outputs with inputs; no qubits are transported. Consequently, quantum computing is always a sequential process without loops.

In general, linear operators change the state of several qubits simultaneously. For further purposes, we define three important operators or gates that act only on a single qubit. Using the standard basis $|0\rangle$, $|1\rangle$, the according matrices read:

$$X_j = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z_j = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad H_j = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{20}$$

The index j defines the qubit on which the operator O_j acts by $O_j(|x_0\rangle \otimes \ldots \otimes |x_j\rangle \otimes \ldots \otimes |x_n\rangle) = |x_0\rangle \otimes \ldots \otimes O_j\,|x_j\rangle \otimes \ldots \otimes \ |x_n\rangle$. The matrices X_j and Z_j are the so-called *Pauli*-matrices, H_j is termed the *Hadamard*-matrix.

There is an unfortunate "overuse" of the letter "H" in quantum mechanics: Hadamard gates, Hamiltonian, Hermitian, ... In this paper, an H with no or a capitalized subscript denotes a Hamiltonian. Hadamard-gates and -matrices are equipped with a small-cap subscript or a number, indicating the qubit the Hadamard-gate acts on.

In this work, the two-qubit gates we use can be understood as the product of two single-qubit gates:

$$\begin{aligned} &A_jB_k(|x_0\rangle \otimes \ldots \otimes |x_j\rangle \otimes \ldots \otimes |x_k\rangle \otimes \ldots \otimes |x_n\rangle) \\ &= |x_0\rangle \otimes \ldots \otimes A_j\,|x_j\rangle \otimes \ldots \otimes B_k\,|x_k\rangle \otimes \ldots \otimes |x_n\rangle\,. \end{aligned} \tag{21}$$

We point out that, first, more general two-qubit gates are possible. Second, if $j = k$, the order of operators may become relevant.

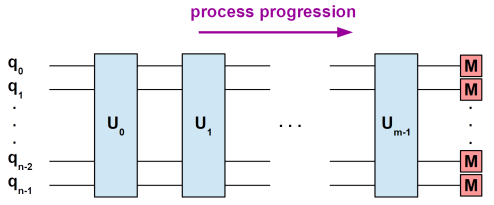

FIGURE 1. Representation of gate-based quantum computation as a sequence of unitary operations, ended by a measurement. Such a diagram is called a *quantum circuit* and is read from left to right. Note well that the "wires" between the "quantum gates" U_j do not mean that qubits are transported from left to right; they define the connections between out- and inputs. The quantum circuit represents the temporal order of operations but not the spatial arrangement of quantum components.

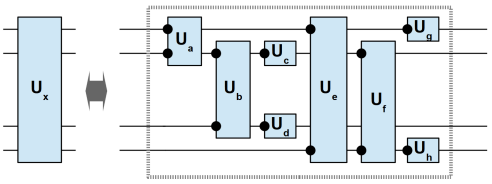

FIGURE 2. Gate-based quantum computation can be realized by a sequence of 2-qubit gates. For clarity, the big dots represent the qubits that are used for inputs of the gates $U_{a,\ldots,h}$. In the usual depiction of quantum circuits, these "input" dots are omitted.

C. QUANTUM ANNEALING

Quantum annealing (QA) can be understood as a form of quantum analogue computation. Adiabatic quantum optimization (AQO) can be used to solve a number of NP-complete or NP-hard problems [14], [26]. Quantum annealing is a generalization of AQO at finite temperature [10].

AQO combines two ideas: First, the adiabatic theorem [8], [14] and second, the possibility of restating many NP-hard problems as questions for the ground state of some specific Hamiltonian.

We start with the adiabatic theorem. Assume, one has a time-dependent Hamiltonian $H(t)$ with a unique ground state $|\phi_0(t)\rangle$ and an energy eigenvalue $\lambda_0(t)$ for all $t \geq 0$. Further, assume that one can prepare the initial state of the system such that it is equal to the ground state: $\Psi(0) = |\phi_0(0)\rangle$. If the time development of the Hamiltonian is sufficiently slow, the system will remain in its ground state $|\phi_0(t)\rangle$ at all times, even if $|\phi_0(0)\rangle \neq |\phi_0(t)\rangle$.

AQO is based on the fact that it is possible to encode many important, even NP-complete problems in terms of coupled qubits, see [26]. We show a simple example, the subset-sum problem: Assume a set of integer numbers $\mathcal{M} = \{n_1, \ldots, n_M\}, n_j \in \mathbb{Z}$. Is there a subset $\mathcal{X} \subseteq \mathcal{M}$ such that the sum over the elements of $\mathcal{X}$ equals zero?

We assume some binary variables $q_j \in \{0, 1\}$. We analyze the expression

$$W = \left(\sum_j n_j q_j\right)^2 = \sum_{j,k} n_j n_k q_j q_k. \tag{22}$$

One then searches for those settings of the q_j that minimize W. If the q_j can be chosen such that $W = 0$, we have a solution for the subset sum problem: $\mathcal{X}$ consists of that n_j for which $q_j = 1$. Otherwise, we get the best possible approximation to a subset that sums up to zero.

For later reference, we note the generalization

$$W = \sum_{j,k} Q_{jk} q_j q_k \tag{23}$$

$$= \sum_{j \neq k} Q_{jk} q_j q_k + \sum_j Q_{jj} q_j. \tag{24}$$

Thereby, we exploited that $q_j^2 = q_j$ for $q_j \in \{0, 1\}$. Avoiding q_j^2-terms is important, because, from a technological point of view, it is possible to couple different qubits, but not a qubit with itself. Optimizing a general W in eq. 23 is called a *quadratic unbounded binary optimization* (QUBO).

Quantum annealing exploits the fact that it is possible to realize "programmable" Hamiltonians. Transforming the QUBO polynomial in eq. 22 into a Hamiltonian requires replacing the q_j by operators. Taking into account that the Pauli-matrix Z is diagonal and observing that $Z|0\rangle = |0\rangle$ and $Z|1\rangle = -|1\rangle$ we get

$$q = \langle q | \frac{1}{2}(1 - Z) | q \rangle . \tag{25}$$

We want to construct a Hamiltonian H_P such that $\langle q_n \ldots q_1 | H_P | q_1 \ldots q_n \rangle = W$ which we can achieve with

$$H_P = \sum_{j \neq k} Q_{jk}(1 - Z_j)(1 - Z_k) + \sum_j Q_{jj}(1 - Z_j). \tag{26}$$

The programmability of the system results from the fact that we can choose the Q_{jk}. Choosing $Q_{jk} = \frac{1}{4} n_j n_k$ for $j \neq k$ and $Q_{jj} = \frac{1}{2} n_j^2$ realizes an H_P, which has a ground state that solves the subset sum problem. One invokes the following procedure:

1) We assume an initial Hamiltonian H_S with a ground state $|\phi_0\rangle$ that is easy to prepare.
2) The quantum annealer uses a time-dependent Hamiltonian of the form:

$$H(t) = (1 - s(t))\, H_S + s(t) H_P. \tag{27}$$

Thereby, $s(t)$ is a sufficiently smooth function with $s(0) = 0$ and $s(\tau) = 1$. This implies $H(0) = H_S, H(\tau) = H_P$. If $s(t)$ varies sufficiently slow (which implies a slow changing $s(t)$ with a sufficiently large τ), $|\phi_0\rangle$ will evolve into the ground state of H_P, according to the adiabatic theorem.

After measuring the states of the qubits q_j, one gets the minimum of W. Imperfections in the annealing process may make it necessary to repeat this procedure a number of times.

QA may look like a silver bullet for attacking hard optimization problems. But, see [26], the time τ grows exponentially with the size $N$of the problem: $\tau = O(\exp(\alpha N^\beta))$. However, QA may be competitive, because the coefficients α, β can be smaller than those of the according classical algorithm.

III. QUANTUM OPTIMIZATION ALGORITHMS

A. QUANTUM APPROXIMATE OPTIMIZATION ALGORITHM - OVERVIEW

QA offers a mechanism for solving optimization problems that can be implemented in present quantum hardware. But besides this practical advantage, the "programming" of the Hamiltonian H_P turns out to be rather simple. The question arose, whether QA really needed dedicated hardware or whether some QA-like procedure can be realized with a gate-based quantum computer. Farhi et al. presented the Quantum Approximate Optimization Algorithm (QAOA)

which achieves this, at least in the sense of an approximation ([13], [16], [17], for a very clear webinar, see [3]).

Assume two lists containing p parameters each: $\beta = (\beta_1, \ldots, \beta_p)$ and $\gamma = (\gamma_1, \ldots, \gamma_p)$. The goal of QAOA is to construct a parameterized state $|\psi(\beta, \gamma)\rangle$ such that a function $\mathcal{F}(\psi(\beta, \gamma))$ is minimal for an H_P that is assumed to encode a given optimization problem. For a general $|\Psi\rangle$, we define:

$$\mathcal{F}(|\Psi\rangle) = \langle\Psi| H_P |\Psi\rangle \tag{28}$$

With an H_S as in eq. 27 and to be specified in what follows, QAOA uses for $\psi(\beta, \gamma)$ the ansatz:

$$|\psi(\beta, \gamma)\rangle = e^{-i\beta_p H_S} e^{-i\gamma_p H_P} \ldots e^{-i\beta_1 H_S} e^{-i\gamma_1 H_P} |\psi_0\rangle . \tag{29}$$

QAOA is based on a number of observations:

1) The solution of the optimization problem we analyze is a binary bit string, which is encoded by one single basis state $|k_{sol}\rangle$.
2) It is hard to find the right $|k_{sol}\rangle$ with classical means. It is also not feasible to compute $|\psi(\beta, \gamma)\rangle$ from eq. 29 with a classical computer in an efficient manner.
3) However, for an arbitrary single basis state $|k\rangle$, it may be easy to compute $\mathcal{F}(|k\rangle)$ with a classical computer.
4) As we will show, for optimally chosen β, γ, the state $|\psi(\beta, \gamma)\rangle$ is expected to be a superposition of basis states which is dominated by the basis state $|k_{sol}\rangle$ that solves the optimization problem encoded by H_P.

Assuming all this, QAOA can be described by the procedure:

1) Initialize β, γ.
2) Prepare $|\psi(\beta, \gamma)\rangle$ using a quantum computer.
3) Measure $|\psi(\beta, \gamma)\rangle$ in the basis of the quantum register. This delivers a state $|k_{guess}\rangle$.
4) Compute $\mathcal{F}(|k_{guess}\rangle)$ with a classical computer and determine new parameters $\beta_{\mathbf{new}}, \gamma_{\mathbf{new}}$ until some stop criterion is met.

QAOA is a heuristic and hybrid approach that combines classical and quantum computation. In this heuristics, we assume that the dominating basis state of $|\psi(\beta, \gamma)\rangle$ after having reached the stop criterion is equal to $|k_{sol}\rangle$, i.e. the solution. This is not guaranteed, or only so, if $p \to \infty$ in eq. 29 and an analogy to QA is achieved, see next section.

We emphasize that steps 2 and 3 in the above procedure maybe performed repeatedly in order to enhance the quality of step 4.

B. MOTIVATION FOR QAOA

QAOA is motivated because it can be understood as an approximative solution of the Schrödinger equation eq. 10 with a time-dependent Hamiltonian $H(t)$ as in eq. 27.

To understand this, we first note that for a constant Hamiltonian, the solution of the Schrödinger equation is given by $|\psi(t)\rangle = \exp(-iHt)|\Psi(0)\rangle$. We then assume a discretization of time $t_k = k\Delta t$. For small Δt, we get $|\psi(\Delta t)\rangle \approx \exp(-iH(0)t)|\Psi(0)\rangle$ and with iteration

$$|\psi(k\Delta t)\rangle \approx e^{-iH(t_{k-1})\Delta t} |\psi((k-1)\Delta t)\rangle \tag{30}$$

If $t = p\Delta t$, this motivates

$$|\psi(t)\rangle \approx e^{-iH(t_{p-1})\Delta t} \ldots e^{-iH(0)\Delta t} |\psi(0)\rangle . \tag{31}$$

Sometimes, eq. 31 is written as

$$|\psi(t)\rangle = e^{-i\int_0^t H(t')dt'} |\psi(0)\rangle . \tag{32}$$

Note, however, that eq. 32 has to be interpreted with some care if the commutator of the Hamiltonian at different times does not vanish; for more details, consult the literature. The question of commutators is also relevant for eq. 31.

Second, we analyze eq. 31 for $H(t)$ as in eq. 27, i.e. the situation of quantum annealing. Again, we observe that we may have the situation, in which H_S and H_P do not commute: $[H_S, H_P] \neq 0$. Even in the case of a non-vanishing commutator, for Hermitian operators A, B and real-valued time t, the Lie-Trotter-Suzuki decomposition holds:

$$e^{i(A+B)t} = e^{iAt} e^{iBt} + O(t^2). \tag{33}$$

We get

$$e^{-i((1-s(t))H_S + s(t)H_P)\Delta t} \approx e^{-i(1-s(t))H_S \Delta t} e^{-is(t)H_P \Delta t}. \tag{34}$$

We point out that usually, one works with a constant Δt. This is no necessity, we can allow for varying Δt_k. If we consider this and combine eq. 34 with eq. 31, and set

$$\begin{aligned} \beta_k &= (1 - s(t_k))\Delta t_k, \\ \gamma_k &= s(t_k)\Delta t_k, \end{aligned} \tag{35}$$

we see that an approximation to quantum annealing eq. 31 is equivalent to eq. 29 (we remark that there are different sign conventions in the literature, partially caused be the fact that, historically, authors looked for the maximal eigenvalue of some problem encoding H_P). This implies that if we can find an encoding of an optimization in terms of a QUBO (eq. 23), we can always find a Hamiltonian H_P that enables an approach as in eq. 29, see [20] and [26].

The question remains how to choose $|\psi_0\rangle$. A usual choice is

$$\begin{aligned} |\psi_0\rangle &= H_1 \otimes \ldots \otimes H_n |q_1 = 0\rangle \otimes \ldots \otimes |q_n = 0\rangle , \\ &= H_1 |q_1 = 0\rangle \otimes \ldots \otimes H_n |q_n = 0\rangle , \\ &= H^{\otimes n} |0\rangle, \\ &= \bigotimes_{j=1}^{n} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \end{aligned} \tag{36}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} |j\rangle . \tag{37}$$

The third line establishes the notation for a single qubit operation that is applied to every qubit. The last line shows why the application of the Hadamard-gate on each qubit is sensible: We construct a state in which each basis state appears with the same weight.

We further note that the states $(|0\rangle \pm |1\rangle)$ are eigenstates of the Pauli-matrix X. It is then easy to see that $|\psi_0\rangle = H^{\otimes n}|0\rangle$ is the ground state of

$$H_S = -\sum_{j=1}^{n} X_j, \tag{38}$$

in accordance with the adiabatic theorem uses in QA.

The class of hybrid classical-quantum algorithms are considered promising since they can better handle erroneous, small-scale quantum devices [30] than "pure" quantum algorithms. QAOA has been suggested as one of the major hybrid classical-quantum algorithms [13] for solving generic optimization problems.

IV. MULTIPLE QUERY OPTIMIZATION WITH QAOA

We refer to the MQO as described in Sec. I.

A. MQO IN THE QUBO - FORMALISM

We define binary variables $q_{ij} \in \{0, 1\}$ with $i \in \{1, .., Q\}$ and $j \in \{1, .., P\}$. A QUBO - formulation of the MQO (see [41] and eq. 23) is given by

$$W = \sum_{i=1}^{Q}\sum_{k=1}^{P} c_{ik} q_{ik} - \sum_{i\neq j}^{Q}\sum_{k,l=1}^{P} s_{ikjl} q_{ik} q_{jl} + E_L + E_M. \tag{39}$$

The first two terms in eq. 39 are similar to those in eq. 1.

In an MQO, there is an important restriction: For each query, one must choose exactly one plan, which is achieved by a proper choice of E_L and E_M.

We start with the observation that if we set

$$E_L = -w_L \sum_{i=1}^{Q}\sum_{j=1}^{P} q_{ij}$$
$$w_L = max(c_{ij}) + \epsilon \tag{40}$$

for some small positive ϵ, we enforce the system to set all $q_{ij} = 1$. At a first glance, this looks silly, but it guarantees that we choose at least one plan per query. The weight w_L is chosen such that the "minimization gain" by setting a $q_{ij} = 1$ is not overcompensated by the according costs c_{ij}.

To enforce that not more than one plan is chosen, we set

$$E_M = w_M \sum_{i=1}^{Q}\sum_{k\neq l}^{P} q_{ik} q_{il},$$
$$w_M = w_L + \sum_{i\neq j}^{Q}\sum_{k,l=1}^{P} s_{ikjl}. \tag{41}$$

The sum of $q_{ik}q_{il}$ over query plans for a specific query is equal to zero if maximally one plan per query is chosen (and equal or bigger than one, if more than one plan is activated). The weight w_M guarantees that selecting two plans for the same query is more costly than the activation benefit by E_L and even compensates potential gains by savings s_{ikjl}. The sum over all savings is necessary, because, for example, if in query i one selects plan k_1 and k_2 and in query j plan l_1 and l_2, four savings would be realized.

B. TRANSLATION INTO A PROBLEM ENCODING HAMILTONIAN

The QUBO - term given by eq. 39 can be translated into a Hamiltonian H_P by invoking the translation of eq. 25. Taking into account that constant terms in Hamiltonians have no (relevant) physical effect (see any text book on quantum mechanics), H_P takes the form:

$$H_P = \sum_{j=1}^{n} a_j Z_j + \sum_{k,l=1}^{n} b_{kl} Z_k Z_l \tag{42}$$

for parameters a_j, b_{kl} that result from collecting terms together. Very relevantly, all the Z_j and the $Z_k Z_l$ commute: $[Z_j, Z_k] = [Z_j Z_k, Z_l] = [Z_j Z_k, Z_l Z_m] = 0$. This implies that

$$e^{-i\gamma H_P} = e^{-i\gamma \sum_{j=1}^{n} a_j Z_j + \sum_{j,k=1}^{n} b_{jk} Z_j Z_k}$$
$$= \prod_{j=1}^{n} e^{-i\gamma a_j Z_j} \prod_{k,l=1}^{n} e^{-i\gamma b_{kl} Z_k Z_l} \tag{43}$$

Thereby (because of the vanishing commutator), the order of the individual factors does not matter. We further note that each factor is a unitary operator (they are exponentials of Hermitian matrices, see the section on quantum mechanics):

$$U_{Z_j}(\theta) = e^{-i\theta Z_j},$$
$$U_{Z_k Z_l}(\theta) = e^{-i\theta Z_k Z_l}. \tag{44}$$

These unitary operators can be realized by standard quantum gates. We do not elaborate on this, but give notice to the fact that $U_{Z_k Z_l}(\theta)$ can be expressed in terms of two so-called CNOT-gates, which sandwich a simple $U_{Z_j}(\theta)$ gate. A similar argumentation can be invoked with respect to H_S. Also the X_j-operators commute with each other, and in consequence, we get:

$$e^{-i\beta H_S} = e^{-i\beta \sum_{j=1}^{n}(-X_j)},$$
$$= \prod_{j=1}^{n} e^{i\beta X_j}. \tag{45}$$

Note well: X_j and Z_j do not commute; that means that in the translation of eq. 29 one must maintain the temporal order of the $\beta-$ and $\gamma-$terms, see Figure 3.

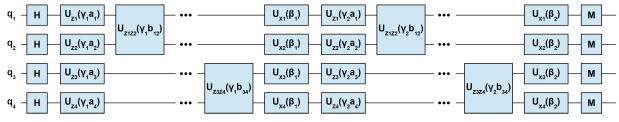

FIGURE 3. Implementation of a QAOA with four qubits and $p = 2$. We use the notation of the MQO discussed in the text. Eq. 29 is translated into gates. Note that the order of terms has to be reversed in the circuit: Whereas the order of terms in the circuit has to be read from left to right, the operators in eq. 29 have to be read from right to left. The computation ends with a measurement M.

C. RUNNING THE ALGORITHM

The hybrid classical-quantum algorithm has several degrees of freedom. First, the number p of steps for the approximation in the functional $\mathcal{F}(\beta, \gamma)$ in eq. 28 which one wants to minimize. ($\mathcal{F}(\beta, \gamma)$ is based on the ansatz given by eq. 29). Second, the number I of times the circuit is run and, third, the parameter lists β and γ that one wants to optimize [30], [44].

We subsequently refer to p as the depth of the quantum circuit. Currently, QAOA is, in general, not well understood for depths $p > 1$ [44]. An exception to this is the recent contribution by Arute, Frank et al., examining circuits with $p = 3$ on Google's private Sycamore quantum device [21]. Although a higher depth theoretically leads to better results, as the quantum annealing evolves "smoother", more gates contribute to more error [13]. Hence, the depth p should be chosen carefully depending on factors such as problem size or error rate of the quantum device. As each additional layer of $\mathcal{F}$ contributes another pair of parameters β and γ, the parameter space to be classically optimized quickly increases, as already noted by the original authors [13]. To mitigate this problem, Zhou et al. provide helpful heuristics and strategies, which they prove by extensive experimentation on graph problems [44].

To summarize, the algorithm proceeds as follows (see Algorithm 1):

Algorithm 1 Quantum Query Optimization Algorithm

Input : Costs c_{ij}, savings s_{ikjl}, queries Q, depth p
Output: Least-cost solution k^*
Formulate the MQO in QUBO form using C, S and Q;
Formulate a Hamiltonian H_P that encodes this QUBO;
Define initial parameters β, γ;
Based on eq. 29, implement a quantum circuit for $\psi(\beta, \gamma)$;
while $r > threshold$ **do**
  Put quantum register into initial state $H^{n\otimes}|0\rangle$;
  for $i \leftarrow 1$ **to** p **do**
    Compute $\psi(\beta, \gamma)$ on quantum register with the quantum circuit produced before;
  end
  $k_{guess} \leftarrow$ Measure quantum register;
  Optimize β, γ, e.g. with gradient descent;
end
return *Least-cost solution* k^*;

As the algorithm is approximate, the optimization loop is typically executed until a threshold is reached instead of finding the exact solution. Often, an approximation ratio r is used as figure of merit for $\mathcal{F}$'s performance [13]. r is defined as follows:

$$r = \frac{\mathcal{F}(|k^*\rangle)}{F_{\min}} \tag{46}$$

where $F_{\min}$ denotes a lower bound for the cost of the MQO problem.

V. EXPERIMENTS AND RESULTS

The primary objective of our experiments is to determine how our implementation of QAOA for solving an MQO scales on current quantum computers. Note that due to the limitations of current quantum technology, a comparison with a classical query optimizer such as the one from Postgres is out of scope for this paper. However, we will compare our solution against the only currently existing quantum query optimization implementation by Trummer and Koch [41].

In particular, we address the following research questions:

- Q1: How can we determine the optimal parameters γ and β for the quantum circuit?
- Q2:What is the optimal number of repetitions for the computation of $|\psi(\beta, \gamma)\rangle$ to find the best solution?
- Q3: For which combination of queries and query plans can we find the optimal solution?
- Q4: What is the run time of our gate-based algorithm?
- Q5: How do our gate-based implementations compare to a quantum annealing-based implementation?
- Q6: What is the complexity of our gate-based algorithms?

In order to address the above-mentioned questions, we need to analyze different parts of our end-to-end algorithm separately. In short, the algorithm's "division of labour" can be summarized as follows. (1) The parameterized quantum part explores the search space. By exploiting quantum properties such as superposition/quantum parallelism and entanglement, the exploration is expected to be much faster than with classical approaches. (2) The classical part is concerned with finding parameters γ and β that, fed to the quantum part, direct the exploration into promising regions of the search space. With a growing problem search space, the parameter search space grows as well [13], [44], making the classical optimization challenging.

We first analyze the classical part of the QAOA on the Qiskit quantum simulator [2]. In particular, we determine the parameters γ and β. Afterwards, we evaluate the quantum part of the hybrid classical-quantum algorithm where we run experiments on the publicly available quantum computer *ibmq Melbourne* [24] with 15 qubits. This device is also used in recent research, including [33]. We also provide an anonymized version of our code.[3]

A. BOOTSTRAPPING THE CLASSICAL OPTIMIZATION PART WITH A FOURIER STRATEGY

In this section, we address research question Q1, i.e. "How can we determine the optimal parameters γ and β for the quantum circuit"?

When the quantum circuit is constructed in Algorithm 1, initial values for parameters γ and β need to be set to start the classical optimization. Usually those values are set randomly. Initializing γ and β at random has multiple disadvantages. First, the randomly generated values often lead the optimizer

[3]The anonymized version of our source code is available at: https://drive.google.com/file/d/1MiZTBzbrm8_SVnruGcPa9hX1PC_W8wO3/view?usp=sharing

to converge toward local optima and therefore the circuit returns sub-optimal solutions. Second, the deeper the circuit (the higher p in eq. 29), the more parameters need to be initialized. Hence finding good initial parameters through random initialization gets exponentially more difficult.

We base our experiments on the FOURIER heuristic strategy introduced in [44] which seems to perform very well in finding good initial parameters for γ and β. Following the FOURIER strategy, one calculates new values for γ_i and β_i through the following transformation based on previously optimized values:

$$\gamma_i = \sum_{k=1}^{p} u_k \sin\left[\left(k-\frac{1}{2}\right)\left(i-\frac{1}{2}\right)\frac{\pi}{p}\right],$$
$$\beta_i = \sum_{k=1}^{p} v_k \cos\left[\left(k-\frac{1}{2}\right)\left(i-\frac{1}{2}\right)\frac{\pi}{p}\right]$$

where the parameters $(u, v) \in \mathbb{R}^{2p}$ correspond to the parameters $(\gamma, \beta) \in \mathbb{R}^{2p}$. With this strategy only the initial parameters γ_1 and β_1 are determined randomly. Because of the rotational symmetry of qubit states, the initial parameter guess can be restricted to $\beta_1, \gamma_1 \in [-\frac{\pi}{2}, \frac{\pi}{2})$.

To verify the results presented in [44], we compare the performance of QAOA with parameters found through the FOURIER strategy against randomly initialized parameters. For the experiment we first generate a random multiple query optimization problem of four qubits, i.e. two queries with two plans each - similar to our running example. On that problem we apply the QAOA with different numbers of repetitions of generating $|\psi(\beta, \gamma)\rangle$ and evaluation of $F(|k_{guess}\rangle$. We perform 30 runs of the QAOA with FOURIER as well as with randomly initialized parameters for each number of repetitions on the simulator. The result can be seen in Figure 4. The average FOURIER strategy performs as well as the best randomly found parameters and is therefore our algorithm of choice for determining the parameters of the further experiments.

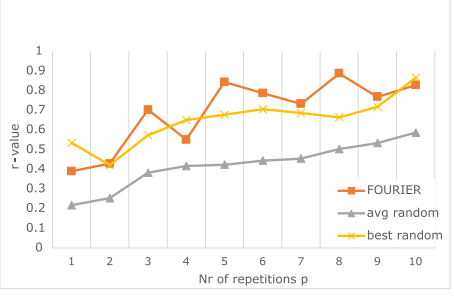

FIGURE 4. Comparison between the FOURIER strategy and randomly initialized parameter sets for γ and β. As can be seen, the FOURIER performs equally as the best of the randomly initialized parameters. The y-axis shows the approximation ratio r.

As a classical optimizer we use the POWELL optimizer provided by Qiskit [2], as it showed the best results in our experiments. We used all optimizers with the default settings provided by Qiskit and 1000 iterations at most.

B. HYBRID CLASSICAL-QUANTUM OPTIMIZATION

Next, we address research questions Q2, i.e. "What is the optimal number of repetitions for the computation of $|\psi(\beta, \gamma)\rangle$ to find the best solution?" and Q3, i.e. "For which combination of queries and query plans can we find the optimal solution?"

In particular, we vary the number of queries as well as the number of query plans and study the probabilities of finding the optimal solution, i.e. the optimal query plan, for various problem sizes.

The QAOA algorithm is currently not well understood beyond $p = 1$. To investigate the behaviour and necessity of more than one layer for larger problem sizes, we apply the QAOA on various randomly generated MQO problem sizes of up to 14 plans. In theory, and empirically proven on the simulator in our experiments, a deeper circuit (higher p in eq. 31) should lead to a better approximation of the minimum cost. However, deeper circuits also increase gate-induced errors [21], [44]. We perform this experiment on the *ibmq Melbourne* quantum computer where we evaluate the accuracy of the QAOA on 50 synthetic problems that are obtained by choosing P plan costs at random for Q queries, while PQ equals the number of qubits on the system and Q and P are varied. Savings are chosen at random and are defined between all pairs of plans except those for the same query.

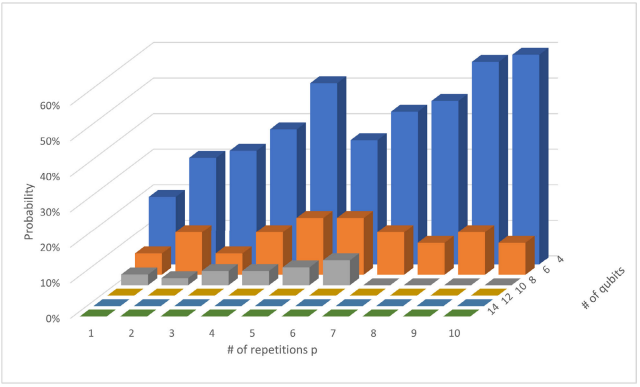

FIGURE 5. The accuracy of measuring the correct solution for problems of size four to 14 qubits with one up to ten layers in the circuit. The accuracy peaks at 59% on problem instances with four qubits i.e. two queries with two plans each. On problems of size six qubits the accuracy ceils at 16% with $p = 6$. Problem instances of size larger than eight qubits have a 0% accuracy on the ibmq Melbourne in our experiment.

Figure 5 shows the probability of finding the optimal solution as a function of the number of qubits, i.e. problem size, and the number of layers p. For problem sizes of eight qubits and above, the probability of finding the optimal solution converges to zero. In other words, deeper circuits can currently not be processed by the ibmq Melbourne device.

C. RUN TIME OF THE GATE-BASED ALGORITHMS

In this section, we address research question Q4, i.e. What is the run time of our gate-based algorithms?

Each run is concluded by measuring the state of the qubits, which is a stochastic procedure. For statistics, the quantum circuit is executed and then measured multiple times, where each cycle is called a shot. Typically, a thousand to ten thousand shots are conducted [19]. Table 1 shows the minimum, median and maximum run times for four differently sized problems. For all experiments, a depth of $p = 5$ was used on the ibmq Melbourne quantum computer.

TABLE 1. Run times of quantum circuits (shots) for different problems with the number of queries Q and the number of plans per query P in milliseconds.

			Run time in ms		
	Q	P	**Min.**	**Med.**	**Max.**
14 Qubits	7	2	14.2	28.0	35.3
	2	7	23.8	25.0	30.9
8 Qubits	4	2	14.3	15.1	15.2
	2	4	14.6	15.2	15.7

The run times mainly depend on the number of qubits used, which are determined by the number of plans times the number of queries. If there are many alternative plans for each query (i.e. P is large), the execution time increases slightly. Although the doubling of qubits used leads to a doubling in run times, this correlation might be coincidental. The time complexity is analyzed in more detail at the end of this section. Generally, the execution time is not only determined by the amount of gates, but as follows: (time to prepare the qubit's state) + (time per gate) times (amount of gates) + (time to measure) [19].

D. COMPARISON WITH EXISTING APPROACHES

In this section, we address research question Q5, i.e. How does our gate-based implementation compare to a quantum annealing-based implementation?

To the best of our knowledge, there is only one quantum-based solution to address the MQO suggested by *Trummer and Koch* [41]. In the following paragraph, we compare this approach to ours.

In their paper, Trummer and Koch utilize a D-Wave quantum annealer with more than 1000 qubits [41], while we employ a gate-based quantum computer with 15 qubits [24]. The qubit disparity between quantum annealers and gate-based quantum computers persists to this day, with quantum annealers featuring almost two orders of magnitude more qubits than gate-based quantum computers [9], [23]. Clearly, the D-Wave device's superiority in qubits allows it to solve larger problems with as many as 1074 queries with two plans each, or 540 queries with 5 plans each, as shown in Table 2.

Similarly to our approach, Trummer and Koch assign plans to qubits to form the solution to the MQO. For this, they map plans as variables to the physical qubits, called physical mapping [41]. The physical layout of the qubits on the D-Wave quantum annealer is represented as a so called Chimera graph [29].

From a technological point of view, a general implementation of eq. 23 is not possible; one cannot implement a design

TABLE 2. Comparison of the approaches by Trummer and Koch [42] and ours. Although the D-Wave quantum device can tackle significantly larger problems, there are situations it can only use a fraction of its qubits.

	Trummer and Koch	**This work**
Max. # of queries	537 queries, 2 plans	7 queries, 2 plans
Qubits used	1074 / 1097 (98%)	14 / 14 (100%)
Max. # of plans	108 queries, 5 plans	2 queries, 7 plans
Qubits used	540 / 1097 (49%)	14 / 14 (100%)

in which (at least potentially) all possible pairs of qubits can be coupled (note that a coupling requires some means for physical interaction). The D-wave quantum annealer offers a technically feasible connection scheme, the Chimera graph, that is optimized such that a sufficiently large subset of all possible couplings can be realized.

To model cost savings, which become active when certain pairs of qubits have been selected for the solution, qubits have to interact with each other. For this interaction, plan-encoding qubits have to be connected to each other. As the Chimera graph is not fully connected, it is necessary to encode a plan in multiple qubits to ensure that the necessary interactions can be facilitated.

This mapping introduces two restrictions. First, the physical mapping introduces an overhead, in particular with problems that require strong interaction. An example of such a problem is the MQO with many plans per query. With the requirement to only choose one plan per query, all plans within that query must be connected, augmenting the degree of interaction. As a consequence, with a higher number of plans per query, around half of the quantum annealer's qubits are required for the physical mapping, as shown in Table 2. The second restriction is that the physical mapping with the minimal number of qubits is itself an NP-hard problem [25].

Although our approach is heavily limited by the number of qubits of gate-based quantum devices, it does not suffer both of these restrictions, as all of the available qubits can be used to encode plans and the physical mapping is a one-to-one mapping from a variable to a qubit.

E. COMPLEXITY OF OUR ALGORITHM

In this section, we address research question Q6, i.e. "What is the complexity of our algorithms?"

We provide a sketch of an evaluation of the computational complexity of our algorithm and compare it with the classical brute-force approach. Recall that for the MQO, we denote the number of queries by Q, the number of plans per query by P and the total number of plans by PQ. For each of the Q queries, one of the P plans must be selected. With this requirement, only admissible solutions remain.

$$\begin{array}{cccc} q_1 & q_2 & \cdots & q_Q \\ \underbrace{p_1, p_2, p_3} & \underbrace{p_4, p_5, p_6} & \cdots & p_{PQ-2}, p_{PQ-1}, p_{PQ} \\ P \text{ poss.} & P \text{ poss.} & \ldots & P^Q \text{ possibilities} \end{array}$$

By brute force, P^Q possibilities must be evaluated.[4] Considering a constant time to evaluate the cost of each

[4]If we included non-admissible solutions with no or more than one plan per query, 2^{PQ} evaluations would be required.

solution, the *complexity of the brute force approach* is as follows:

$$O(BruteForce) = O(P^Q) \tag{47}$$

For the quantum approach, we focus on the quantum part and assume the complexity of the classic optimization as constant $O(1)$. Furthermore, we assume the complexity of translating the MQO into a classical cost function, and into a quantum circuit to be constant.

The *space complexity* of the quantum circuit can trivially be answered as $O(PQ)$ because our approach requires exactly one qubit for every plan of the MQO. The *time complexity* for quantum computations is typically determined by the depth of the quantum circuit [19]. This is because the depth indicates the longest succession of gates that must be executed sequentially. The time required for a computation to complete is obtained by multiplying the number of operations with the average time a quantum gate requires [19].

For the complexity sketch, we analyze eq. 43. The dominant term is the couplings between two qbuits $ZiZj$. Since there are $PQ(PQ-1)/2$ possible couplings, the depth of one layer is of the order $O((PQ)^2)$.

In consequence, if time complexity is defined by total circuit depth, we get for QAOA applied on MQO:

$$O(QAOA(MQO)) = O(p(PQ)^2) \tag{48}$$

Compared to the brute force complexity of $O(P^Q)$, this is a significant speedup. For comparison, Grover's algorithm for searching an unstructured list achieves a speedup from $O(n)$ to $O(\sqrt{n})$. This suggests that the quantum part of the algorithm does not only work for very small problems but also for real-world-sized problems, as the number of gates grows moderately.

VI. CONCLUSION

In this paper, we have studied gate-based quantum algorithms to find quasi-optimal solutions for the multiple query optimization problem – a classical problem in database optimization. Our algorithms are designed for near-term gate-based quantum computers, consisting of a parameterized quantum part for exploring the search space, and a classical part for optimizing the parameters. Our approach is the first contribution using gate-based quantum computers for tackling query optimization. While the classical part of hybrid classical-quantum algorithms is typically difficult due to the high-dimensional parameter space, we implemented a recently suggested strategy as a remedy, thus improving the solution quality.

As current-day gate-based quantum computers are restricted in the number of qubits and fault tolerance, our gate-based algorithm can currently not directly compete against implementations on a quantum annealing architecture, for which quantum devices with more capacity exist. However, when comparing our hybrid approach with a competing pure quantum-annealing approach, we found that our approach has two advantages. First, it can utilize the quantum device's qubits more efficiently, as we have a direct mapping from the mathematical formulation to the quantum implementation. In contrast, an ideal mapping with quantum annealers is an NP-complete problem. Second, our algorithm is based on an approach that was previously shown to solve "hard" problems that quantum annealers may not be able to solve and that may be found within the MQO context.

Finally, we analytically sketched the scaling of our algorithm and found that with larger problem sizes, the circuit depth grows polynomially while the solution space grows exponentially. Further, the space requirements only grow linearly with the problem size.

We believe that our paper lays a solid groundwork for using hybrid classical-quantum algorithms to tackle the special problem of multiple query optimization. Even though we can currently only solve relatively small problems due to the limitations of current quantum technology, our results are promising given the rapid development of quantum computing. We thus set the stage for a novel database research agenda at the intersection of classical and quantum computing.

REFERENCES

[1] S. Aaronson, *Quantum Computing Since Democritus*. Cambridge, U.K.: Cambridge Univ. Press, 2013.
[2] H. Abraham, R. Agarwal, I. Y. Akhalwaya, G. Aleksandrowicz, T. Alexander, and M. Amy, "Qiskit: An open-source framework for quantum computing," Tech. Rep., 2019. [Online]. Available: https://zenodo.org/records/2562111
[3] R. S. Argonne, "QAOA introduction," Tech. Rep., Aug. 2021. [Online]. Available: https://www.youtube.com/watch?v=4JfwuAX_cvU
[4] F. Arute et al., "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.
[5] M. A. Bayir, I. H. Toroslu, and A. Cosar, "Genetic algorithm for the multiple-query optimization problem," *IEEE Trans. Syst., Man Cybern., C, Appl. Rev.*, vol. 37, no. 1, pp. 147–153, Jan. 2007.
[6] M. Benedetti, M. Fiorentini, and M. Lubasch, "Hardware-efficient variational quantum algorithms for time evolution," *Phys. Rev. Res.*, vol. 3, no. 3, Jul. 2021, Art. no. 033083.
[7] V. Bergholm et al., "PennyLane: Automatic differentiation of hybrid quantum-classical computations," 2018, *arXiv:1811.04968*.
[8] M. Born and V. Fock, "Beweis design adiabatensatzes," *Zeitschrift für Physik*, vol. 51, nos. 3–4, pp. 165–180, Mar. 1928.
[9] D-Wave Systems Inc. (2017). *A the D-Wave 2000QTM Quantum Computer Technology Overview*. Accessed: Jun. 8, 2021. [Online]. Available: https://www.dwavesys.com/resources/white-paper/the-d-wave-advantage-system-an-overview/
[10] A. Das and B. K. Chakrabarti, "Colloquium: Quantum annealing and analog quantum computation," *Rev. Mod. Phys.*, vol. 80, no. 3, pp. 1061–1081, Sep. 2008.
[11] D. P. DiVincenzo, "Two-bit gates are universal for quantum computation," *Phys. Rev. A, Gen. Phys.*, vol. 51, no. 2, pp. 1015–1022, Feb. 1995.
[12] T. Dokeroglu, M. A. Bayır, and A. Cosar, "Integer linear programming solution for the multiple query optimization problem," in *Information Sciences and Systems 2014*. Cham, Switzerland: Springer, 2014, pp. 51–60.
[13] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," 2014, *arXiv:1411.4028*.
[14] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, "A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem," *Science*, vol. 292, no. 5516, pp. 472–475, Apr. 2001.
[15] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Quantum computation by adiabatic evolution," 2000, *arXiv:quant-ph/0001106*.
[16] E. Farhi, J. Goldstone, S. Gutmann, and L. Zhou, "The quantum approximate optimization algorithm and the Sherrington–Kirkpatrick model at infinite size," *Quantum*, vol. 6, p. 759, Jul. 2022.
[17] E. Farhi and A. W. Harrow, "Quantum supremacy through the quantum approximate optimization algorithm," 2016, *arXiv:1602.07674*.

[18] S. Groppe and J. Groppe, "Optimizing transaction schedules on universal quantum computers via code generation for Grover's search algorithm," in *Proc. 25th Int. Database Eng. Appl. Symp.*, Jul. 2021, pp. 149–156.
[19] G. G. Guerreschi and A. Y. Matsuura, "QAOA for max-cut requires hundreds of qubits for quantum speed-up," *Sci. Rep.*, vol. 9, no. 1, pp. 1–7, May 2019.
[20] S. Hadfield, "On the representation of Boolean and real functions as Hamiltonians for quantum computing," *ACM Trans. Quantum Comput.*, vol. 2, no. 4, pp. 1–21, Dec. 2021.
[21] M. Harrigan et al., "Quantum approximate optimization of non-planar graph problems on a planar superconducting processor," *Nature Phys.*, vol. 17, pp. 332–336, Mar. 2021.
[22] J. Heitz and K. Stockinger, "Join query optimization with deep reinforcement learning algorithms," 2019, *arXiv:1911.11689*.
[23] IBM Quantum Team. (2021). *IBMQ_Manhattan V1.21.0*. Accessed: Jun. 9, 2021.
[24] IBM Quantum Team. (2021). *IBMQ_Melbourne V2.3.24*. Accessed: Jun. 2, 2021.
[25] C. Klymko, B. D. Sullivan, and T. S. Humble, "Adiabatic quantum programming: Minor embedding with hard faults," *Quantum Inf. Process.*, vol. 13, no. 3, pp. 709–729, Mar. 2014.
[26] A. Lucas, "Ising formulations of many NP problems," *Frontiers Phys.*, vol. 2, p. 5, Jan. 2014.
[27] D. Lykov, J. Wurtz, C. Poole, M. Saffman, T. Noel, and Y. Alexeev, "Sampling frequency thresholds for quantum advantage of quantum approximate optimization algorithm," *NPJ Quantum Inf.*, vol. 9, no. 1, p. 73, 2022.
[28] R. Marcus, P. Negi, H. Mao, C. Zhang, M. Alizadeh, T. Kraska, O. Papaemmanouil, and N. Tatbul, "Neo: A learned query optimizer," 2019, *arXiv:1904.03711*.
[29] C. C. McGeoch and C. Wang, "Experimental evaluation of an adiabatic quantum system for combinatorial optimization," in *Proc. ACM Int. Conf. Comput. Frontiers*, May 2013, pp. 1–11.
[30] N. Moll, P. Barkoutsos, L. S. Bishop, J. M. Chow, A. Cross, D. J. Egger, S. Filipp, A. Fuhrer, J. M. Gambetta, M. Ganzhorn, A. Kandala, A. Mezzacapo, P. Müller, W. Riess, G. Salis, J. Smolin, I. Tavernelli, and K. Temme, "Quantum optimization using variational algorithms on near-term quantum devices," *Quantum Sci. Technol.*, vol. 3, no. 3, Jul. 2018, Art. no. 030503.
[31] M. A. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2002.
[32] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature Commun.*, vol. 5, no. 1, pp. 1–7, Jul. 2014.
[33] C. Piveteau, D. Sutter, and S. Woerner, "Quasiprobability decompositions with reduced sampling overhead," *NPJ Quantum Inf.*, vol. 8, no. 1, p. 12, Feb. 2022.
[34] M. Schönberger, "Applicability of quantum computing on database query optimization," in *Proc. Int. Conf. Manage. Data*, Jun. 2022, pp. 2512–2514.
[35] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price, "Access path selection in a relational database management system," in *Readings in Artificial Intelligence and Databases*. Amsterdam, The Netherlands: Elsevier, 1989, pp. 511–522.
[36] T. Sellis and S. Ghosh, "On the multiple-query optimization problem," *IEEE Trans. Knowl. Data Eng.*, vol. 2, no. 2, pp. 262–266, Jun. 1990.
[37] T. K. Sellis, "Multiple-query optimization," *ACM Trans. Database Syst.*, vol. 13, no. 1, pp. 23–52, 1988.
[38] R. D. M. Simões, P. Huber, N. Meier, N. Smailov, R. M. Füchslin, and K. Stockinger, "Experimental evaluation of quantum machine learning algorithms," *IEEE Access*, vol. 11, pp. 6197–6208, 2023.
[39] M. Stillger, G. M. Lohman, V. Markl, and M. Kandil, "LEO-DB2's learning optimizer," in *Proc. VLDB*, vol. 1, 2001, pp. 19–28.
[40] M. Streif and M. Leib, "Comparison of QAOA with quantum and simulated annealing," 2019, *arXiv:1901.01903*.
[41] I. Trummer and C. Koch, "Multiple query optimization on the D-wave 2X adiabatic quantum computer," 2015, *arXiv:1510.06437*.
[42] I. Trummer and C. Koch, "Multi-objective parametric query optimization," *ACM SIGMOD Rec.*, vol. 45, no. 1, pp. 24–31, Jun. 2016.
[43] D. Venturelli and A. Kondratyev, "Reverse quantum annealing approach to portfolio optimization problems," *Quantum Mach. Intell.*, vol. 1, nos. 1–2, pp. 17–30, May 2019.
[44] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices," *Phys. Rev. X*, vol. 10, no. 2, Jun. 2020, Art. no. 021067.

TOBIAS FANKHAUSER received the bachelor's degree in computer science from the Zurich University of Applied Sciences, Switzerland. He is currently pursuing the master's degree in artificial intelligence and data science with the University of Zurich. His research interests are machine learning and quantum computing.

MARC E. SOLÈR received the bachelor's degree in computer science from the Zurich University of Applied Sciences, Switzerland. He is currently pursuing the master's degree in computer science with the University of St. Gallen. His research interests are software systems and quantum computing.

RUDOLF MARCEL FÜCHSLIN received the degree in theoretical physics from ETH Zürich and the Ph.D. degree in computational physics from the University of Zurich. He held various positions at German and Italian institutions. Currently, he heads the Group for Applied Complex Systems Sciences, School of Engineering, Zurich University for Applied Sciences, Winterthur, Switzerland, and also the Co-Director of the European Centre for Living Technology, Venice, Italy.

KURT STOCKINGER received the Ph.D. degree in computer science from CERN, University of Vienna. He is a Professor of computer science and the Director of studies in data science with the Zurich University of Applied Sciences (ZHAW) and the Co-Head of the ZHAW Datalab. He is also an External Lecturer with the University of Zurich. He was with the Berkeley Laboratory, Credit Suisse, Caltech, and CERN. His research focuses on data science, with an emphasis on big data, natural language query processing, query optimization, and quantum computing.