THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

Quantum Computing for Airline Planning and Operations

Marika Svensson

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2023

Quantum Computing for Airline Planning and Operations

Marika Svensson

This thesis has been prepared using LaTeX.

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
Phone: +46 (0)31 772 1000
www.chalmers.se

Printed by Chalmers Reproservice
Gothenburg, Sweden, September 2023

To Lennart

Abstract

Classical algorithms and mathematical optimization techniques have been used extensively by airlines to optimize their profit and ensure that regulations are followed. In this thesis, we explore which role quantum algorithms can have for airlines. Specifically, we have considered the two quantum optimization algorithms; the Quantum Approximate Optimization Algorithm (QAOA) and Quantum Annealing (QA). We present a heuristic that integrates these quantum algorithms into the existing classical algorithm, which is currently employed to solve airline planning problems in a state-of-the-art commercial solver. We perform numerical simulations of QAOA circuits and find that linear and quadratic algorithm depth in the input size can be required to obtain a one-shot success probability of 0.5. Unfortunately, we are unable to find performance guarantees. Finally, we perform experiments with D-wave's newly released QA machine and find that it outperforms 2000Q for most instances.

Keywords: discrete optimization, quantum approximate optimization algorithm, quantum annealing, airline scheduling, column generation

List of Publications

This thesis is based on the following publications:

[A] **Marika Svensson**, Martin Andersson, Mattias Grönkvist, Pontus Vikstål, Devdatt Dubhashi, Giulia Ferrini, and Göran Johansson, "A Hybrid Quantum-Classical Heuristic to solve large-scale Integer Linear Programs". arXiv:2103.15433 (accepted to Physical Review Applied).

[B] Dennis Willsch, Madita Willsch, Carlos D. Gonzalez Calaza, Fengping Jin, Hans De Raedt, **Marika Svensson**, and Kristel Michielsen, "Benchmarking Advantage and D-Wave 2000Q quantum annealers with exact cover problems". Quantum Inf Process 21, 141 (2022).

[C] Pontus Vikstål, Mattias Grönkvist, **Marika Svensson**, Martin Andersson, Göran Johansson, and Giulia Ferrini, "Applying the Quantum Approximate Optimization Algorithm to the Tail Assignment Problem". Phys. Rev. Applied 14, 034009 (2020).

[D] Andreas Bengtsson, Pontus Vikstål, Christopher Warren, **Marika Svensson**, Xiu Gu, Anton Frisk Kockum, Philip Krantz, Christian Križan, Daryoush Shiri, Ida-Maria Svensson, Giovanna Tancredi, Göran Johansson, Per Delsing, Giulia Ferrini, and Jonas Bylander, "Improved success probability with greater circuit depth for the quantum approximate optimization algorithm". Phys. Rev. Applied 14, 034010 (2020).

Other related work

[1] Rishi Sreedhar, Pontus Vikstål, **Marika Svensson**, Andreas Ask, Göran Johansson, and Laura García-Álvarez, "The Quantum Approximate Optimization Algorithm performance with low entanglement and high circuit depth". arXiv:2207.05612.

Acknowledgments

I would first like to thank my advisors at Jeppesen Mattias Grönkvist and Martin Andersson for their continuous support and for many useful discussions. I would also like to thank Andreas Gunnarson for his genuine interest in this project, the Pairing team and my manager Åse Kytte for all support given.

I would also like to thank my advisor at Chalmers University of Technology Göran Johansson, for the vital aspects of physics, guidance within academics, and his newly found interest in integer programming. Then I would like to thank the assistant advisor Devdatt Dubhashi for his support around computational complexity and, generally, theoretical computer science topics.

Furthermore, I would like to thank Pontus Vikstål for collaboration and discussions regarding simulations of quantum computers and finally Giulia Ferrini for exciting discussions and support, in this truly interdisciplinary field.

Finally, I would also like to thank all the members of the Department of Computer Science and Artificial Intelligence and the research group of Applied Quantum Physics for providing a nice place to work at.

Contents

Part I

Overview

CHAPTER 1

Introduction

Airlines have used classical algorithms and mathematical optimization [1] techniques extensively for decades [2]. In this thesis, we explore which role quantum algorithms can have in this context. In the following sections, we give a brief overview of the airline industry, airline scheduling problems, and quantum computing.

1.1 The Airline Scheduling Process

Airlines operate within an industry that is highly competitive with large operational costs. In particular, the largest costs are related to fuel consumption and crew. The airlines are furthermore governed by many operational rules imposed by aviation authorities and unions. It is also common that airlines themselves have internal rules that need to be respected. Additionally, airlines also must deal with uncertainties due to weather conditions and other disruptions. These characteristics force airlines to carefully schedule their flights, crew, and aircraft to maximize revenue and minimize cost.

Scheduling Problems

An airline has two significant decisions before scheduling [3], namely the fleet size and structure and which routes to cover. Typically an airline begins by determining the fleet, i.e., how many aircraft of each fleet should exist and how many aircraft there are in total. Given that the set of aircraft is determined, the airline can consider which origins and destinations they wish to cover, i.e., route planning. Generally, after these planning stages are completed, the airline can consider the scheduling of flights, aircraft, and crew.

The scheduling problem [4], [5] has historically been divided into subproblems Flight Scheduling, Fleet Assignment, Crew Scheduling, Tail Assignment, and lastly Recovery Planning. Commonly, Crew Scheduling is further partitioned into Crew Pairing and Crew Rostering. In this setting, the output of a subproblem is input to the following subproblem and solved sequentially; see Fig. 1.1. However, by considering the subproblems individually, the over-

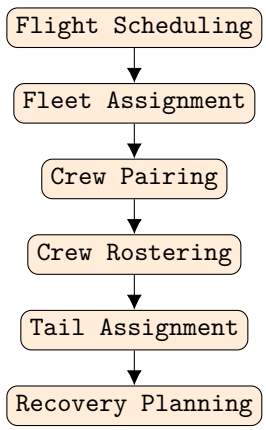


Figure 1.1: Sequential airline scheduling process

all scheduling problem is not solved optimally. Therefore more integrated solutions have also been considered, where two or more subproblems can be solved iteratively, considering some aspects of other subproblems, or a completely new integrated model of multiple problems is proposed. Although there exists no proposal to integrate all subproblems into a single problem, probably due to the complexity and size such a problem would have. Examples of integrated problems are Crew Scheduling with Tail Assignment and

Fleet Assignment with Flight Scheduling. There are some drawbacks when considering integrated optimization models since the subproblems are complex and large, even when considered individually. Integrating two or more subproblems can be very difficult to model, and the integrated optimization models will obviously become even larger in the number of required variables and constraints. The integrated optimization problems can therefore be more cumbersome to find good solutions to and fail in practice to perform better than the sequential approach in Fig. 1.1.

The goal of Flight Scheduling [6]–[15] is to construct a flight schedule, which is a list of flight legs specified by their arrival and departure airport, dates or frequency, and time. This stage is typically completed six months in advance. The considerations an airline usually has in Flight Scheduling are demand, ticket price, market share, airport slots, and non-stop flights versus connecting flights for certain origins and destinations. Since the sold flight tickets constitute the airline's revenue, the objective becomes to maximize the expected revenue.

Using the output of Flight Scheduling, i.e., the flight schedule, it is possible to consider Fleet Assignment. In Fleet Assignment [16]–[33], the goal is to cover the flight schedule with the existing fleet and maximize the profit. This means that we need to assign a fleet type to each flight leg while not exceeding the number of aircraft in each fleet. To maximize the profit, the goal is to match the demand to the capacity of the aircraft, as this minimizes the spill cost of lost passengers and operating costs.

Once the flight schedule and the fleet type are known for each flight, we can consider the Crew Scheduling problem. The Crew Scheduling problem [34]–[37] is typically partitioned into two problems, Crew Pairing and Crew Rostering. The reasons for separating the problems are the sheer size and complexity and the constraints and goals that differ. In Crew Pairing, the goal is to find anonymous legal pairings in the most cost-effective way such that the flight schedule is covered and the working contractual rules for the anonymous crew are respected. The cost is typically measured in the working time, and a legal pairing is a sequence of duties, that are sequences of flight legs typically representing one day of work, with rest and layovers in between that start and ends in a crew base and respect a set of additional rules.

Crew Rostering, on the other hand, focuses on finding monthly personalized rosters for each crew member such that all flights are covered, and the weekly

and monthly rules for individual crew members are respected. The personalized rosters for each crew consist of pairings generated in the Crew Pairing phase and personal activities such as vacation, reserves, and training with time off in between. The typical objective is to maximize the roster qualities by fairness and/or requests from individual crew members. Crew Pairing is, therefore, more important than Crew Rostering when it comes to increasing profit.

Once there is a Fleet Assignment and/or Crew Scheduling solution, individual tails, the numbers that identify a particular aircraft, must be assigned to each flight in the flight schedule while performing the required maintenance checks. This problem is an Aircraft Assignment problem, and specifically here the Tail Assignment problem [38]–[42]. The solution of Tail Assignment is a set of maintenance feasible aircraft routes, that is, sequences of flight legs, assigned to minimize the assignment cost.

Once all these problems are solved, the airline has successfully scheduled its flights, aircraft, and crew to maximize profit while respecting all rules. However, there are many uncertainties, as noted previously. As a result, schedules can be disrupted by unforeseen circumstances. To manage this new situation, it is often necessary to reevaluate the schedules, which is handled in the Recovery Planning phase [43], [44].

Ch. 2 and 3 will explain further the mathematical models and solution techniques commonly used.

1.2 Quantum Computing

Quantum mechanics [45] was discovered and developed during the first quarter of the 1900s, introducing concepts such as quantum superposition, quantum entanglement, and quantum measurement. The theory allowed for a greater understanding of the universe throughout the century, as it correctly described nature in regimes where classical mechanics failed.

The theory of computational complexity [46]–[48] and the performance of the classical computer progressed at great speed during the latter half of the 1900s, following Moore's law [49], [50] which says that the power of classical computers will double every two years. However, this law is currently breaking down as the hardware components become so small that quantum mechanical effects disturb their functionality. One possible way of further-

ing our computing capability is to propose a new model of computation. In 1979-1981 Benioff [51], Manin [52] and Feynman [53] independently proposed such concepts, namely a model of computation based on quantum mechanics. Feynman argued that to simulate quantum mechanical systems, such a model of computation might be required, as it seems to be intractable for a classical computer to do exactly without exponential resources and time. The idea was thus that a machine where information is embedded in quantum mechanical systems might be more powerful than a machine where the information is embedded in classical mechanical systems. Today we view quantum superposition and entanglement as resources of this model that are distinctly different from the resources of classical computation. Quantum entanglement allow for non-local operations compared to classical computing and quantum superposition allow the device to be in a superposition of all classical states. Indeed, quantum computers stand today as a possible model of computation that may violate the extended Church-Turing thesis, which says that a probabilistic Turing machine can simulate any reasonable model of computation in polynomial time. The principles of quantum computing can be found in Ref. [54] but also will be introduced in Ch. 4.

Following the proposals by Benioff, Manin and Feynman, much insight has been obtained about controlling single quantum systems such as ion traps and superconducting qubits, quantum information, quantum algorithms, and how to construct a quantum computer [54], [55]. Deutsch showed in 1985 that universal quantum computing [56] is possible to realize in theory, along with a problem that can be solved in constant time by a quantum algorithm compared to linear time by a deterministic classical algorithm [57], albeit a probabilistic classical algorithm also solves the problem in constant time. Bernstein and Vazirani [58] were the first to show separation between quantum computation and classical computation, as they gave a problem that a quantum algorithm solves in constant time whereas linear time is required for both the deterministic and probabilistic classical algorithms. They also proposed a version of the quantum Fourier transform, which gives an exponential speedup compared to classical algorithms. Exponential speedup was also obtained by Simon's algorithm [59] shortly after, but what is now considered a defining major breakthrough in the field was discovered by Shor [60], who presented an algorithm, that uses the quantum Fourier transform as a building block, that solves the discrete log problem and in extension the integer factoriza-

tion problem. This quantum algorithm provides exponential speedup over the best known classical algorithm, and is applicable to a problem we encounter on a regular basis, namely the RSA encryption [61], which is broken by Shor's algorithm. Moreover, Grover presented an unstructured database search algorithm with quadratic speedup [62], which has later also been used to propose several algorithms to solve discrete optimization problems. However, these algorithms rely on quantum error correction and hence fault-tolerant quantum computers, which have yet to be shown experimentally viable. Furthermore, some algorithms require specific oracle access, which is nontrivial to achieve experimentally.

What has been proposed instead is to consider hybrid quantum-classical variational algorithms such as the Quantum Approximate Optimization Algorithm (QAOA) [63] and the Variational Quantum Eigensolver (VQE) [64] in the so-called Noisy Intermediate-Scale Quantum [65] (NISQ) era of quantum computers. We will discuss the quantum variational algorithms designed to solve optimization problems further in Ch. 4.

Moreover, the idea of quantum computing has given rise to quantum complexity classes. The complexity class of greatest interest to us is Bounded-error Quantum Polynomial-time (BQP) [54], which is the quantum analog of Bounded-error Probabilistic Polynomial-time (BPP) [47]. Finally, we note that it has been shown that BPP$\subseteq$ BQP$\subseteq$PSPACE [58], but it is not known if there is a separation BPP$\neq$BQP, meaning that it is still not proven that quantum computers are more powerful than classical computers. This can be considered counterintuitive, as we just have mentioned quantum speedup. However, for the problems and quantum algorithms presented, the speedup is either given when we assume oracle access, which can't separate the classes, or when we do not know the hardness of the problem as in the case of integer factorization.

1.3 Contribution

I assisted with the following work related to the papers appended and discussed in this thesis:

- Paper A: I derived problem instances of interest, performed all numerical simulations, and was the main author of the manuscript

- Paper B: I derived the problem instances and assisted with reviewing the manuscript
- Paper C: I assisted in reviewing the manuscript and assisted in some numerical circuit simulations
- Paper D: I presented the instances to consider and assisted in reviewing the manuscript

1.4 Thesis Outline

In Ch. 2, the mathematical background is given for Multi-Commodity Network Flow problems and solution methodologies. A further explanation of the mathematical models used for the Aircraft Assignment problem Tail Assignment is then given in Ch. 3. Ch. 4 gives a review of the model of quantum computation and some quantum algorithms designed to solve integer programming problems. The appended papers are summarized in Ch. 5, and finally, we give our conclusions and suggestions for future research possibilities in Ch. 6.

CHAPTER 2

Network Flows and Mathematical Optimization

This chapter introduces the general minimum cost Multi-Commodity Network Flow Problem (MCNFP) as it models many airline planning problems, sometimes with additional constraints. We also summarize solution methodologies and focus on Dantzig-Wolfe decomposition, Column Generation, and in the integrality case Branch-and-Price.

2.1 The Multi-Commodity Network Flow Problem

Surveyed in [66] and [67], the MCNFP [68] model optimization problems in areas such as logistics, transportation, and telecommunication. The problem consists of a directed graph $G = (V, A)$ with a set of nodes V and arcs A. In addition, we have a set of K commodities that, in essence, differentiates the problem from the Single-Commodity Network Flow Problem (SCNFP). The goal is to ship B^k units of each commodity $k \in K$ across the graph from source nodes s_k to sink nodes t_k such that the sum of arc costs c_{ij}^k is as small as possible, whilst respecting capacity constraints on each arc u_{ij} and arc-flow variable x_{ij}^k.

Thus, similarly to the SCNFP, we wish to move commodities from the source

to the sink subject to mass balance constraints, capacity constraints and a cost minimization. However, here the capacity constraints link all commodities together, which causes the MCNFP to be a more difficult problem to solve than the SCNFP. In particular, if we have integral variables as the Linear Programming (LP) relaxation [69] of the SCNFP, in this case, has integer solutions, whereas the MCNFP does not. Clearly, if the linking constraints are ignored, the MCNFP decomposes to $|K|$ SCNFPs that can be solved separately. There are two main formulations of these problems, an arc-flow formulation, and a path-flow formulation. We will now discuss these.

Arc-flow Formulation

The arc-flow formulation of the minimum cost MCNFP is the following

$$z^* = \text{minimize} \quad \sum_{k \in K} \sum_{(i,j) \in A} c_{ij}^k x_{ij}^k, \tag{2.1}$$

$$\text{subject to} \quad \sum_{(i,j) \in A} x_{ij}^k - \sum_{(j,i) \in A} x_{ji}^k = b_i^k, \qquad \forall i \in V,\ \forall k \in K, \tag{2.2}$$

$$\sum_{k \in K} x_{ij}^k \leq u_{ij}, \qquad \forall (i,j) \in A, \tag{2.3}$$

$$x_{ij}^k \geq 0, \qquad \forall (i,j) \in A,\ \forall k \in K, \tag{2.4}$$

where

$$b_i^k = \begin{cases} B^k & \text{if } i = s_k \\ -B^k & \text{if } i = t_k \\ 0 & \text{otherwise} \end{cases}. \tag{2.5}$$

The decision variables x_{ij}^k represent the flow of commodity k on each arc $(i,j) \in A$ in the directed graph. Eq. (2.1) is the linear cost function with cost coefficients c_{ij}^k associated with sending commodity k across arc (i,j). There are $|V||K|$ mass balance constraints, sometimes referred to as continuity constraints, in Eq. (2.2). Here the supply, demand, and continuity are secured by the coefficients b_i^k for each node and commodity. Each commodity has its own source node s_k and sink node t_k, and there can in general be several source and/or sink nodes, but here we only consider the case when we have a single source node and a single sink node for each commodity. The linking

constraints of the commodities are given by the upper bound on the arc capacity u_{ij} in Eq. (2.4), clearly these $|A|$ constraints are forcing us to solve the problem in this form, whereas if these constraints are ignored, it is possible to decompose the problem into $|K|$ separate problems.

Path-flow Formulation

We can reformulate the arc-flow formulation as a path-flow formulation. Since there are for each commodity k a set of P^k possible simple directed paths from s_k to t_k, we can associate the flow on each path with a decision variable f_p. We can then relate the arc-flow decision variable to the path-flow decision variable accordingly

$$x_{ij}^k = \sum_{p \in P^k} \delta_{ij}(p) f_p, \tag{2.6}$$

where $\delta_{ij}(p)$ is 0 if arc (i, j) is not in path p and 1 otherwise. We can also express the cost of a path p for commodity k as $c_p^k = \sum_{(i,j)\in A} \delta_{ij}(p) c_{ij}^k$. Regarding the mass balance constraints, we can remove some, as we require that the sum of all directed paths from source to sink for each commodity deliver B^k units. Lastly, the capacity constraint is easily expressed in path-flow variables by replacing the arc-flow variables. Since we require that each arc-flow variable is non-negative, we also require this for the path-flow variables. The path-flow formulation is thus

$$z^* = \text{minimize} \quad \sum_{k \in K} \sum_{p \in P^k} c_p^k f_p, \tag{2.7}$$

$$\text{subject to} \quad \sum_{p \in P^k} f_p = B^k, \qquad \forall k \in K, \tag{2.8}$$

$$\sum_{k \in K} \sum_{p \in P^k} \delta_{ij}(p) f_p \leq u_{ij}, \qquad \forall (i,j) \in A, \tag{2.9}$$

$$f_p \geq 0, \qquad \forall p \in P^k, \ \forall k \in K. \tag{2.10}$$

We now have an equivalent formulation of the problem where we have reduced $|V||K| + |A|$ constraints in the arc-flow formulation to $|K| + |A|$ constraints, but we have increased the number of variables from $|A||K|$ to $\sum_{k \in K} |P^k|$ which is in general exponential in the size of the directed graph.

2.2 Solution Approaches

The solution approaches to MCNFP [68, Chapter 17] are typically classified into three categories: price-directive decomposition, resource-directive decomposition, and partitioning methods.

The price-directive decomposition decomposes the problem into a main problem and $|K|$ subproblems where prices are put on the linking constraints. The mass balance constraints and individual arc flow constraints define the subproblems for each commodity along with an objective function to be minimized. The objective function is the original arc cost for a path with additional prices added. The role of the subproblems is to find improving paths for the main problem. The main problem sets the prices and connects the individual subproblems via the linking constraints. This is often done via Lagrangian decomposition, Dantzig-Wolfe decomposition and/or Column Generation. The two latter approaches will be presented in the following sections.

The idea of resource-directive decomposition is that we view the problem as a capacity allocation problem. We then separate the MCNFP into a resource allocation problem and $|K|$ additional minimum cost flow problems that depend on a fixed resource vector $\mathbf{r}$. This decomposition changes the arc-flow formulation to the following two problems

$$z^* = \text{minimize} \quad \sum_{k \in K} z_k(\mathbf{r}), \tag{2.11}$$

$$\text{subject to} \quad \sum_{k \in K} r_{ij}^k \leq u_{ij}, \qquad \forall (i,j) \in A, \tag{2.12}$$

$$r_{ij}^k \geq 0, \qquad \forall (i,j) \in A,\ \forall k \in K, \tag{2.13}$$

and

$$z_k(\mathbf{r}) = \text{minimize} \quad \sum_{(i,j) \in A} c_{ij}^k x_{ij}^k, \tag{2.14}$$

$$\text{subject to} \quad \sum_{(i,j) \in A} x_{ij}^k - \sum_{(j,i) \in A} x_{ji}^k = b_i^k, \qquad \forall i \in V, \tag{2.15}$$

$$0 \leq x_{ij}^k \leq r_{ij}^k, \qquad \forall (i,j) \in A. \tag{2.16}$$

These problems are solved iteratively, where we reallocate capacities such that the solution is improved in each iteration.

For partitioning methods, one uses the fact that the MCNFP is an LP with embedded single-commodity network flow problems. One useful method that has been developed for SCNFP is that spanning tree solutions are basic feasible solutions in the simplex algorithm [70]. By generating improving spanning trees, the SCNFP can be solved. This idea is expanded upon for MCNFP, where additional arcs are required to ensure that the linking constraints are satisfied.

To summarize, all three categories of solution approaches to the MCNFP share the fact that we separate the problem into several problems, that each only considers one commodity. In this thesis, we focus on a price-directive method, namely the Dantzig-Wolfe decomposition and Column Generation. Our motivation for this choice is that we consider the current state-of-the-art implementation to solve problems of interest in the airline planning process.

Dantzig-Wolfe Decomposition

The Dantzig-Wolfe decomposition was first presented in [71], and constitutes a method for decomposing a linear program such that we obtain a formulation based on extreme rays and points in domains that are defined by constraints in the original formulation. If we consider the linear program

$$\text{minimize} \quad \mathbf{c}_1^T\mathbf{x}_1 + \mathbf{c}_2^T\mathbf{x}_2 + \cdots + \mathbf{c}_N^T\mathbf{x}_N, \tag{2.17}$$

$$\text{subject to} \quad D_1\mathbf{x}_1 + D_2\mathbf{x}_2 + \cdots + D_N\mathbf{x}_N \leq \mathbf{d}, \tag{2.18}$$

$$\begin{bmatrix} A_1\mathbf{x}_1 & & & \\ & A_2\mathbf{x}_2 & & \\ & & \ddots & \\ & & & A_N\mathbf{x}_N \end{bmatrix} \leq \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_N \end{bmatrix}, \tag{2.19}$$

$$\mathbf{x}_1,\ \mathbf{x}_2, \ldots, \mathbf{x}_N \geq \mathbf{0}, \tag{2.20}$$

where all variables are linked by the D_i matrices, and the A_i matrices are separate constraints for each vector $\mathbf{x}_i$.

Assume for a moment that we want to formulate a problem with only the constraints in Eq. (2.18) involving the linking constraints. This means that we somehow wish to remove the rest of the constraints. Let us begin by defining the sets

$$X_i = \{\mathbf{x}_i \geq 0 \mid A_i\mathbf{x}_i \leq \mathbf{b}_i\} \quad \forall i = 1, \ldots, N. \tag{2.21}$$

If these sets are convex and nonempty, a point $\mathbf{x}_i \in X_i$ can be written as a convex combination of the extreme points $\bar{\mathbf{x}}_i^p \in P_i$ and a conical combination of extreme rays $\tilde{\mathbf{x}}_i^r \in R_i$ [72, Chapter 3]

$$\mathbf{x}_i = \sum_{p \in P_i} \lambda_i^p \bar{\mathbf{x}}_i^p + \sum_{r \in R_i} \lambda_i^r \tilde{\mathbf{x}}_i^r, \quad \sum_{p \in P_i} \lambda_i^p = 1, \quad \lambda_i^l \geq 0,\ l \in P_i \cup R_i. \tag{2.22}$$

It is then possible to express the original linear program by substituting in Eq. (2.22), which gives us the following equivalent formulation

$$\text{minimize} \quad \sum_{i=1}^{N} \Big(\sum_{p \in P_i} \lambda_i^p \mathbf{c}_i^T \bar{\mathbf{x}}_i^p + \sum_{r \in R_i} \lambda_i^r \mathbf{c}_i^T \tilde{\mathbf{x}}_i^r \Big), \tag{2.23}$$

$$\text{subject to} \quad \sum_{i=1}^{N} D_i \Big(\sum_{p \in P_i} \lambda_i^p \bar{\mathbf{x}}_i^p + \sum_{r \in R_i} \lambda_i^r \tilde{\mathbf{x}}_i^r \Big) \leq \mathbf{d} \quad | \ \vec{\pi}, \tag{2.24}$$

$$\sum_{p \in P_i} \lambda_i^p = 1, \quad i = 1, \ldots, N \quad | \ q_i, \tag{2.25}$$

$$\lambda_i^l \geq 0, \quad \forall l \in P_i \cup R_i, \quad i = 1, \ldots, N. \tag{2.26}$$

We have at this point successfully removed the constraints we desired in our formulation by considering the extreme points and rays in the polyhedrons X_i. The reformulated problem now has decision variables λ_i^l, and we will denote this problem as the Master Problem (MP).

Column Generation

If it is possible to find all extreme points and rays and the number of extreme points and rays is not too large, the problem obtained from the Dantzig-Wolfe decomposition can be solved directly. However, this is not the case in general. Rather, the number of extreme points and rays can be very large, even exponentially large in the input size. We, therefore, seek a method where we only consider some subset of all variables. This is where the Column Generation algorithm [73] becomes essential.

The problem with only a subset of variables $R_i' \subset R_i$ and $P_i' \subset P_i$ for $i = 1, \ldots, N$ is called the Restricted Master Problem (RMP). By solving the RMP, the optimal primal and dual variables can be obtained. Additionally,

the reduced cost of a variables λ_i^p and λ_i^r for given dual variables $\vec{\pi}$ and q_i are

$$\bar{c}_i^p = \mathbf{c}_i^T \bar{\mathbf{x}}_i^p - (D_i \bar{\mathbf{x}}_i^p)^T \vec{\pi} - q_i \quad \text{and} \qquad \bar{c}_i^r = \mathbf{c}_i^T \tilde{\mathbf{x}}_i^r - (D_i \tilde{\mathbf{x}}_i^r)^T \vec{\pi}, \tag{2.27}$$

respectively. Since a variable with negative reduced cost can improve the solution of the RMP, we wish to find the smallest by solving the Pricing Problem (PP)

$$\text{minimize}_{\mathbf{x}_i \in X_i} (\mathbf{c}_i - D_i^T \vec{\pi})^T \mathbf{x}_i - q_i, \tag{2.28}$$

which is equivalent to solving the following two minimization problems

$$\min\Big(\text{minimize}_{p \in P_i} (\mathbf{c}_i - D_i^T \vec{\pi})^T \bar{\mathbf{x}}_i^p - q_i,\ \text{minimize}_{r \in R_i} (\mathbf{c}_i - D_i^T \vec{\pi})^T \tilde{\mathbf{x}}_i^r \Big).$$

By solving the problems, we can find a column that can enter the basis (i.e., the RMP) which will either be an extreme point

$$\begin{pmatrix} \mathbf{c}_i^T \bar{\mathbf{x}}_i^p \\ D_i \bar{\mathbf{x}}_i^p \\ 1 \end{pmatrix}$$

or an extreme ray

$$\begin{pmatrix} \mathbf{c}_i^T \tilde{\mathbf{x}}_i^r \\ D_i \tilde{\mathbf{x}}_i^r \\ 0 \end{pmatrix}.$$

When there are no variables $p \in P_i$ or $r \in R_i$ in any pricing problem $i = 1, \ldots, N$ with a negative reduced cost, the optimal RMP has been found, and also the optimal solution to the original problem presented in Eq. (2.17)-(2.20).

The Column Generation algorithm viewed from a Dantzig-Wolfe decomposition perspective is explained as iteratively solving the RMP and the $2 \cdot N$ pricing problems based on the dual variables. If we then solve each pricing problem, and we find new extreme points or rays with negative reduced cost, we introduce these variables in the basis (the RMP). Repeating this process will generate an improved solution in each iteration. When no extreme point or ray with a negative reduced cost is found, the problem is solved optimally. However, we are not necessarily restricted to using Dantzig-Wolfe decomposition to employ the Column Generation algorithm for a problem, and it is possible to use it on the MCNFP directly where we have $|K|$ pricing problems

with the form

$$z_k^* = \text{minimize} \quad \sum_{(i,j)\in A} (c_{ij}^k - \pi_{ij}) x_{ij}^k - \sigma^k, \tag{2.29}$$

$$\text{subject to} \quad \sum_{(i,j)\in A} x_{ij}^k - \sum_{(j,i)\in A} x_{ji}^k = b_i^k, \qquad \forall i \in V, \tag{2.30}$$

$$x_{ij}^k \geq 0, \qquad \forall (i,j) \in A, \tag{2.31}$$

and the Master Problem in the path-flow formulation of the MCNFP.

Branch-and-Bound and Branch-and-Price

Given that we have an MCNFP where the variables are continuous, linear programming based techniques work well. However, it is often the case that we have integer valued variables. In this case, we do not accept fractional solutions nor is the domain convex.

One standard method used to find integer solutions is Branch-and-Bound [74], [75]. Consider here that we are applying Branch-and-Bound to an integer linear program

$$\text{ILP} = \min \left\{ \sum_{i=1}^{n} c_i x_i : \vec{x} \in S \right\}$$

where $S = \left\{ \vec{x} \in \mathbb{Z}_+^n : \sum_{i=1}^n a_{ji} x_i \geq b_j \ \forall j = 1, \ldots, m \right\}$. First, we consider what happens if the variables are relaxed to be continuous. We know from linear programming theory that we will obtain either a lower bound, an integer solution, or that the problem is infeasible.

If we find either an integer solution or an infeasible solution, we have either obtained the optimal solution or found that there is no integral solution, and we can stop. However, if we obtain some fractional solution $\mathbf{x}^0$, we have obviously not reached our goal. In Branch-and-Bound we, therefore, choose to create $k \geq 2$ subproblems, where we choose the subproblems to be in disjoint domains $S_1, \ldots, S_k$ that exclude the solution $\mathbf{x}^0$.

One way of creating disjoint domains with respect to the fractional variables in solution $\mathbf{x}^0$ is to add constraints for fractional variables in our solution $x_j \leq \lfloor x_j^0 \rfloor$ in one of the subproblems and $x_j \geq \lceil x_j^0 \rceil$ in another. The original problem and subproblems can be visualized as a tree, where the nodes represent the optimization problems, with one parent and two or more children.

Since we now have two or more new problems to explore, we again meet a similar situation as for our relaxed ILP, but for the newly created subproblems. This means that if a subproblem is found to be infeasible, the node is pruned, i.e., the node representing one of the subproblems is not explored further as we do not create any more subproblems. Instead, we choose a new subproblem to explore that has been generated previously, and we say that the node is *pruned by infeasibility*. If we find that a solution to a subproblem is integral, we also do not create any further subproblems and prune this node. As we have found an optimal partial solution, a candidate incumbent z_i, for this specific region and instead choose a new subproblem to explore. Imagine now that we also have access to some other feasible solution to the ILP, which we denote z^* and call the incumbent. If we find that a subproblem yields a fractional solution greater or equal to z^*, this region cannot contain any integer solutions that would improve upon the one we already have, and there is no point in creating more subproblems to the node we are currently exploring. This particular node is thus *pruned by bound*, and we choose a new subproblem to explore that has been generated previously. On the other hand, if the fractional solution is smaller than the incumbent, we create new subproblems, before we pick a new node to explore.

With these ingredients, we can state the Branch-and-Bound algorithm. The algorithm begins by first initializing a list $\mathcal{L}$ with the node N_0 that represents our relaxed ILP. We also initialize an incumbent $z^* = \infty$. We then pick a node in our list and attempt to solve this problem. If there are no feasible solutions, we pick another node from our list as this node is pruned by infeasibility. If we obtain an integral solution the node is also pruned, now by integrality, and if the integral solution is less than our incumbent, we update it. We then pick a new subproblem from the list. If we obtain a fractional solution worse than our incumbent, we prune this node by bound. Finally, if a subproblem can not be pruned by infeasibility, bound or integrality, the subproblem is partitioned into $k \geq 2$ nodes representing k subproblems, which are children to the current subproblem we are exploring in the tree. The k subproblems are then added to a list $\mathcal{L}$ of unexplored subproblems and a new subproblem is chosen to be explored. When there are no unexplored subproblems left, the algorithm terminates and returns the incumbent solution and the corresponding assignment.

We remark that the Branch-and-Bound algorithm is exponential in the

worst case. However, exhaustive search is avoided by pruning nodes of the tree, giving more acceptable running times in practice. Finally, when we solve each node with the Column Generation method, the algorithm is called Branch-and-Price [76].

CHAPTER 3

Airline Scheduling Models

Mathematical optimization models for airline scheduling problems can differ significantly depending on the type of network, planning horizon, network representation, uncertainties, objective function, and considered constraints. Indeed, it does not seem surprising that depending on the assumptions made for a problem at hand in the airline planning process that we end up with very different mathematical models for something we categorize as approximately the same problem. Furthermore, a so-called good model depends very much on how well the model represents reality and how fast a sufficiently good solution can be obtained. Therefore, we must consider at least the trade-off between the time spent to find a solution versus the solution quality, and we must be aware that the models typically vary for different airlines, and can change if the industry changes.

Now, since the papers discussed in Ch. 5 have only considered a variant of Aircraft Assignment called Tail Assignment, we will restrict the coming section to Aircraft Assignment and elaborate on Tail Assignment specifically. Even though we disregard the details of the other problems in the airline planning process, we note that the problems Fleet Assignment, Crew Pairing, and Crew Rostering have, in many cases, in common with Aircraft Assignment

the property that they can be modeled as networks, which we must send either crew or aircraft through, and that the problems are large. Therefore, the model types and solution approaches for the problems are similar.

3.1 Aircraft Assignment and its Variations

As mentioned previously, many aspects of Aircraft Assignment can vary, leading to several names being used, both in industry and academia. Four common names are *Aircraft Routing* [77]–[81], *Aircraft Maintenance Assignment* [79], [82]–[92], *Through Assignment* [93]–[97] and *Tail Assignment* [38], [42], [98]. The name *Aircraft Assignment* is the common denominator, in the sense that Aircraft Routing, Through Assignment, Aircraft Maintenance Assignment, and Tail Assignment can be considered to be an Aircraft Assignment problem since we wish to assign flights, i.e., routes, to aircraft. In contrast, a general Aircraft Assignment model does not necessarily capture all the modeling aspects of Tail Assignment.

Having different modeling considerations leads of course to the fact that the mathematical models differ, to varying degrees. The models can differ in the ultimate goal, as some airlines consider Aircraft Assignment to be a feasibility problem, whereas others consider it to be an optimization problem. Here, robustness, which refers to how sensitive the solution is to disruptions, might also be more or less important to consider. A typical case of feasibility is Aircraft Maintenance Assignment, where it is often only required to find maintenance feasible routes assigned to aircraft while disregarding the cost of the assignment. Through Assignment, on the other hand, ignores many of the maintenance constraints, and the goal is instead to maximize the through values. The through values are defined as the desirability of one-stop services, i.e., multi-leg flights without aircraft changes. By maximizing this quantity, we minimize the number of aircraft changes for desirable connections, which allows the airline to raise the ticket price and increase the profit. Aircraft Routing refers to when we consider both maintenance requirements and through values.

Often, these three problems are considered earlier in the planning process than Tail Assignment as we do not always expect to obtain an executable solution without some manual changes, whereas Tail Assignment focuses on being able to produce a solution that can be executed without extra ma-

nipulation. For example, it can be desirable to solve Aircraft Maintenance Assignment directly after Fleet Assignment to show that it is possible to find at least some assignment that respects a subset of maintenance rules before considering crew scheduling.

Regardless of what variant of Aircraft Assignment we consider, one further example of diverging models is when we consider cyclic problems, which are problems where the flight schedule is approximately repeated each week or day. Here, it can be sufficient to require the aircraft to land at maintenance stations at the end of each day. Thus complying with some maintenance constraints without explicitly having them in the model, but implicitly enforcing them via the network structure [79]. With this assumption, once a solution is obtained, the schedule can be repeated to get a solution with a longer time horizon. The solutions are then often modified for dated problems, i.e., problems with distinct start and end dates, to consider deviations and improve some feasibility and/or optimality issues. If, on the other hand, the flight schedule has no such regularity, solving cyclic problems is not very useful, and we instead consider only the dated problems. Here it is usually impossible to model all maintenance requirements via the network structure, and we need to have explicit constraints in our model[1].

Another consideration is flexibility, meaning that the model can be adjusted according to what priorities an airline have depending on when they solve it, i.e., if we are several months, weeks, or days away from day-of-operation.

To reiterate, we generally wish to assign all flights to aircraft. Since this assignment corresponds to aircraft routes, we also want these routes to be, at the very least, maintenance feasible in some sense[2]. The maintenance requirements are given by aviation authorities[3], aircraft manufacturers, and airlines, which typically provide stricter constraints compared to the former two sources of constraints.

Tail Assignment

In Tail Assignment, the goal is to assign each flight exactly once to aircraft such that the operational cost is minimized and all operational constraints

[1] Maintenance are often modeled as restricted resources in the network.
[2] This means that if Through Assignment is solved, additional changes are most likely required to comply with maintenance constraints.
[3] Federal Aviation Administration in case of the USA.

such as maintenance, preassigned activities, and prohibited activities are respected. One notable difference is that this problem considers tail (aircraft) specific constraints, which models such as Aircraft Routing often do not consider. Furthermore, even though we have spoken about flight legs so far, Tail Assignment plan in its model something we call activities. Activities can be flight legs, sequences of flight legs, maintenance, and other ground activities. To note, one strength of Tail Assignment is that the model can capture aspects ranging from Aircraft Routing to Aircraft Maintenance Assignment, Through Assignment, Fleet Assignment, and Recovery Planning. Although Aircraft Assignment is typically separated for each fleet type, Tail Assignment is more general and allows multiple fleets which can be required for the final feasibility of the solution. To model all requirements, Tail Assignment only solves dated problems but has, in principle, no limit on the time horizon. This means that Tail Assignment does not capture cyclic problems, which can be considered a weakness of the model. However, in terms of flexibility, scope, and preciseness in the sense that the solution should be executable without extra manipulation, the model is very well-suited for real-world problems.

In [38] Tail Assignment is presented for two different formulations, one which is path-flow based and exponentially large in the number of variables, or rather feasible routes, and linear in the number of constraints, i.e., the number of constraints are the number of flight legs given as input[4]. The second model is arc-flow based where the number of variables is quadratic in the number of activities and linear in the number of aircraft but has more constraints, and the constraints are viewed as complex. Furthermore, although the number of variables is polynomial in the size of the input, the model becomes for practical problems very large. The two models are related via methods described in Ch. 2 as the latter model is classified as a resource-constrained integer minimum cost MCNFP and given explicitly in Eq. (3.1)-(3.7). We can consider the model below to be represented by a directed graph where each node represents a connection between activity i and j for each aircraft. In the network, we use five different types of sets to model our problem. T is the set of aircraft, F is the set of flight legs, P_t is the set of preassigned activities for tail t, R_t are the forbidden activities for tail t and M is the set of maintenance activities. The 0-1 decision variables in Eq. (3.7) represent which activities and thus connections the aircraft should cover where each decision variable

[4]Here we are disregarding some vertical constraints compared to the model in practice.

x_{ijt} is associated to a cost c_{ijt}, giving us the objective function in Eq. (3.1) to minimize. Constraint in Eq. 3.2 is a continuity constraint that ensures that a path is associated with an aircraft, albeit the requirement on the source and sink for each aircraft needs to be added such that exactly one aircraft is sent on each path. The covering constraint is given in Eq. 3.3 and ensures that all flight legs are covered exactly once. Then, the requirement for preassigned activities and forbidden activities for each aircraft are given in Eq. (3.4)-(3.5).

$$\text{minimize} \quad \sum_{i\in F}\sum_{j\in F}\sum_{t\in T} c_{ijt}x_{ijt}, \tag{3.1}$$

$$\text{subject to} \quad \sum_{j\in F} x_{jit} - \sum_{j\in F} x_{ijt} = 0, \qquad \forall i \in F,\ \forall t \in T, \tag{3.2}$$

$$\sum_{j\in F}\sum_{t\in T} x_{ijt} = 1, \qquad \forall i \in F, \tag{3.3}$$

$$\sum_{j\in F} x_{ijt} = 1, \qquad \forall i \in P_t,\ \forall t \in T, \tag{3.4}$$

$$\sum_{j\in F} x_{ijt} = 0, \qquad \forall i \in R_t,\ \forall t \in T, \tag{3.5}$$

$$r_{im} \leq l_m, \qquad \forall i \in F,\ \forall m \in M, \tag{3.6}$$

$$x_{ijt} \in \{0,1\}, \qquad \forall i \in F,\ \forall j \in F,\ \forall t \in T. \tag{3.7}$$

The most complex constraints for the model are, unexpectedly, associated with the maintenance in Eq.(3.6). These are defined recursively for each variable $x_{i'it} = 1$ where

$$r_{im} = \begin{cases} r_m^t & \text{if } i \text{ is carry-in activity for aircraft } t \\ s_{im} & \text{if maintenance } m \text{ possible between activities } i' \text{ and } i \\ s_{im} + r_{i'm} & \text{if maintenance } m \text{ not possible between activities } i' \text{ and } i \end{cases}.$$

Here, s_{im} is the resource consumption of maintenance m for activity i, r_m^t is the initial maintenance consumption for maintenance task m and aircraft t, r_{im} is the total resource consumption up to activity i, and l_m is the upper bound on maintenance m.

Notably, in [98] the constraints are clarified such that

$$r_{jmt} = s_{jmt} + \sum_{i \in F} v_{ijmt} r_{imt} x_{ijt} \leq l_{mt}$$

where v_{ijmt} is 1 if maintenance m is not possible for aircraft t between flight leg (or activity) i and j. The consequence is that there is a constraint for each activity, maintenance, and aircraft which can lead to some simplifications. However, the resource maintenance consumption r_{jmt} is still defined recursively and remains complicated.

For both versions, it is possible to decompose the problem via, e.g., Dantzig-Wolfe decomposition discussed in Ch. 2. The decomposition method presents us the master problem, which is an LP relaxed Set-Partitioning problem and $|T|$ pricing problems, where each pricing problem is a resource-constrained shortest path problem. Explicitly, the first model we mentioned for Tail Assignment is obtained by modifying the decision variables to path variables, i.e., route variables x_r, giving us the Set Partitioning model below

$$\text{minimize} \quad \sum_{r \in R} c_r x_r, \tag{3.8}$$

$$\text{subject to} \quad \sum_{r \in R} a_{fr} x_r = 1, \quad \forall f \in F, \tag{3.9}$$

$$x_r \in \{0, 1\}, \qquad \forall r \in R. \tag{3.10}$$

This model associates each route with a route cost c_r in Eq. (3.8) and ensures that each flight is covered exactly once in Eq. (3.9). The number of variables is in the worst case exponential in the size of flight legs, but since we require that all routes must be feasible according to constraints in Eq. (3.2), (3.4), (3.5), (3.6), and (3.7) this number is reduced. However, explicit enumeration is typically intractable still and a well-known issue for these types of problems.

Finally, we note here that to find feasible integer solutions, via standard methods such as Branch-and-Price, we are faced with two NP-hard optimization problems.

CHAPTER 4

The Model of Quantum Computation and Quantum Optimization

The previous chapters are aimed at giving a solid understanding of classical algorithms and modeling considerations typically considered in the airline industry. However, since this thesis is concerned with how quantum algorithms can be employed for such problems, we now present a foundation of quantum computing [54] in Sec. 4.1 and following that we present, in Sec. 4.2, the two quantum algorithms that have been explored in the appended papers.

4.1 Model of Quantum Computation

Two popular models of quantum computation are the quantum circuit model [99] and the quantum Turing machine [56], [58]. The models are equivalent since they can simulate each other in polynomial time [100]. Here we will present the quantum circuit model since it corresponds relatively straightforwardly with algorithms such as QAOA.

Quantum States and Qubits

The quantum mechanical systems we study in this thesis are of some finite dimension N. For such a quantum system, we can express its state as a column vector of l_2 norm 1 in a N dimensional complex linear space $\mathbb{C}^N$, i.e., a Hilbert space $\mathcal{H}$. We can fix an orthonormal basis $|0\rangle, |1\rangle, \ldots, |N-1\rangle$, using the Dirac notation[1] and express a pure state[2] as a superposition of the basis states

$$|\psi\rangle = \sum_{i=0}^{N-1} \alpha_i |i\rangle$$

where the l_2-norm requirement means that $\sum_{i=0}^{N-1} |\alpha_i|^2 = 1$. A quantum bit is the typical building block of a quantum circuit and is the quantum analog of the classical bit. The quantum bit is a two-dimensional quantum system with states in $\mathbb{C}^2$, the orthonormal basis is most commonly chosen to be

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

which means that a general state of a quantum bit is expressed as $|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle$, where α_0 and α_1 are complex numbers. The vector space we consider has the inner product

$$\langle\phi|\psi\rangle = \sum_{i=0}^{N-1} \beta_i^* \alpha_i$$

where $\langle\phi|$ is the conjugate transpose of $|\phi\rangle = \sum_{i=0}^{N-1} \beta_i |i\rangle$, i.e., $\langle\phi| = (|\phi\rangle)^\dagger$.

Suppose now that we have two distinct quantum mechanical systems in Hilbert spaces $\mathcal{H}_1$ and $\mathcal{H}_2$ of dimensions N_1 and N_2 with orthonormal basis $\{|i\rangle_1\}_{i=0}^{N_1-1}$ and $\{|j\rangle_2\}_{j=0}^{N_2-1}$, respectively. We can describe the composite quantum system by the tensor product of the two Hilbert spaces $\mathcal{H}_1 \otimes \mathcal{H}_2$ and obtain an orthonormal basis via tensor products[3] accordingly

$$|ij\rangle_{12} = |i\rangle_1 \otimes |j\rangle_2, \ \forall i = 0, \ldots N_1 - 1, \forall j = 0, \ldots, N_2 - 1.$$

[1] In the Dirac notation we call the column vector a ket vector and the row vector a bra vector.

[2] A mixed state is a probability distribution of pure states.

[3] Typically we abbreviate the tensor product $|i\rangle \otimes |j\rangle$ as $|i\rangle |j\rangle$ or $|ij\rangle$.

Clearly, the dimension of the composite system's Hilbert space is $N_1 \times N_2$. We can express a general quantum state for the composite system in exactly the same manner as for a single system in the basis of the composite system

$$|\phi\rangle_{12} = \sum_{i=0}^{N_1-1} \sum_{j=0}^{N_2-1} \alpha_{ij} |i\rangle_1 |j\rangle_2 .$$

If we consider qubits again, in this case two qubits, we get the orthonormal basis $|0\rangle = |0\rangle \otimes |0\rangle, |1\rangle = |0\rangle \otimes |1\rangle, |2\rangle = |1\rangle \otimes |0\rangle$ and $|3\rangle = |1\rangle \otimes |1\rangle$.

In order to describe a composite system that consists of n quantum mechanical systems with dimensions $N_1, N_2, \ldots, N_n$ we need a Hilbert space of dimension $N_1 \times N_2 \times \cdots \times N_n$. In the case where we consider n qubits the Hilbert space is of dimension 2^n and the states are column vectors in $\mathbb{C}^{2^n}$ with orthonormal basis $\{|i\rangle\}_{i=0}^{2^n-1}$ where $|i\rangle$ is a column vector where the entries are zeros, except in position $i+1$ which has entry one.

Unitary Evolution of Quantum States

Unitary transformation is one of the basic operations a closed quantum system can undergo. This means that we describe the evolution in time of a closed quantum system with unitary operators, where a unitary operator U is such that $U^\dagger U = I$ when we take its matrix representation. Consequently, time evolution and any unitary transformation preserve the norm of quantum states, and evolution is reversible. The unitary time evolution operator relates a state $|\psi\rangle$ at time t_1 to the state $|\psi'\rangle$ as time t_2 as

$$|\psi'\rangle = U |\psi\rangle .$$

In Dirac notation, we can express the unitary operator via the outer product of the orthonormal basis $|i\rangle \langle j|$. The unitary operator then has the following matrix representation

$$U = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} u_{ij} |i\rangle \langle j| , \tag{4.1}$$

and each column $\mathbf{u}_i$ describes how the operator acts on basis state $|i\rangle$. In the case when time is continuous, the evolution is governed by the Schrödinger

equation

$$i\hbar\frac{\partial |\psi\rangle}{\partial t} = H|\psi\rangle\,, \tag{4.2}$$

where H is the Hamiltonian of the quantum system and $\hbar$ is Planck's constant divided by 2π. We can relate the differential equation to the unitary operator easily when the Hamiltonian is time-independent, giving us

$$U = e^{-iH(t_2-t_1)/\hbar}. \tag{4.3}$$

Finally, we note that we can express any unitary operator as $U = e^{iA}$ for some Hermitian[4] operator A.

Quantum Gates

Quantum gates constitute the second building block element of quantum circuits and are unitary operators that evolve the quantum system in time. Common one qubit gates are the Pauli-gates

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \text{ and } \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

As well as the Hadamard-gate, phase-gate, and $\pi/8$-gate below

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},\ S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix},\ \text{and } T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}.$$

Common two-qubit gates are controlled-U gates that act on a control qubit and a target qubit

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{pmatrix}.$$

This gate does not change the state if the control qubit is in the state $|0\rangle$, but if the control bit is in state $|1\rangle$ it applies a gate

$$U = \begin{pmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{pmatrix}$$

[4] A hermitian matrix A is such that $A^\dagger = A$.

to the target qubit. The CNOT-gate is a controlled-U gate where U is the σ_x-gate. A universal quantum gate set is CNOT, Hadamard (H), phase (S) and $\pi/8$ (T) according to the Solovay-Kitaev theorem [54, Appendix 3], as it is possible to approximate any other unitary gate arbitrarily well.

Quantum Measurements

Quantum mechanics prohibit us to observe a quantum state $|\psi\rangle = \sum_{i=0}^{N-1} \alpha_i |i\rangle$ in the sense that we can determine all amplitudes α_i. What quantum mechanics allows for instead are measurements that constitute a third element required for a quantum computational model. If we measure the state $|\psi\rangle$ in its orthonormal basis[5] we will observe an outcome i with probability $|\alpha_i|^2$ and the system will be in the state $\frac{\alpha_i}{|\alpha_i|} |i\rangle$ after the measurement.

Measurement is the second elementary operation a quantum system can undergo and is not described by a unitary operator. Instead, we can describe a measurement in the computational basis, or any other basis, with projective operators. A projective measurement is described by an observable M, which is a Hermitian operator with eigenvalues $\lambda_m,\ m = 1, 2, \ldots, K$. The observable is related to projection operators $\{P_m\}_{m=1}^{K}$. Each projective operator P_m is a projection onto the eigenspace corresponding to eigenvalue λ_m. We write the observable with respect to the different outcomes m as the sum

$$M = \sum_{m=1}^{K} \lambda_m P_m \ \ s.t. \ \ \sum_{m=1}^{K} P_m = \mathbb{1} \text{ and } P_m P_{m'} = \delta_{mm'} P_m.$$

Assuming that the system was in state $|\psi\rangle$ prior to the measurement of the observable, we have a probability

$$P(m) = \langle\psi| P_m |\psi\rangle$$

that outcome m is observed and if outcome m is observed the state collapses to

$$|\psi'\rangle = \frac{P_m |\psi\rangle}{\sqrt{\langle\psi| P_m |\psi\rangle}}.$$

A measurement in the computational basis is a projective measurement and therefore given by the projection operators $P_i = |i\rangle \langle i|\,,\ \forall i = 0, 1, \ldots, N-1$.

[5] Usually referred to as the computational basis.

In the one qubit case, where we measure σ_z, this corresponds to

$$P_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad \text{and} \quad P_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

with eigenvalues $+1$ and -1, respectively.

Quantum Circuits

A classical circuit [47] is a finite directed acyclic graph $G = (E, V)$ with n input nodes that take the input bit values, m output nodes, and internal nodes that each is one of the gates AND, OR, and NOT. The edges of the graph, also called wires, each carry one bit. Each internal node performs logical operations on the bits. Here, we note that the gates AND and OR have fanin, the number of incoming edges, two and fanout, the number of outgoing edges, one. The NOT-gate, on the other hand, has fanin one and fanout one. We note that for classical circuits, we are permitted to copy a bit, which is not allowed in quantum circuits due to the no-cloning theorem. Such a circuit G implements a function $f : \{0,1\}^n \to \{0,1\}^m$, and is a boolean circuit in the case when there is a single output node. This is the classical circuit model.

The quantum circuit model, see an example of a quantum circuit in Fig. 4.1, is defined similarly to the classical circuit model. The classical bits are replaced by quantum bits, the edges of the graph carry qubits, and quantum gates replace the classical gates. Moreover, fanin must be the same as fanout. Finally, quantum measurements are required to be added in order to observe the outcome such that we obtain the classical output bits.

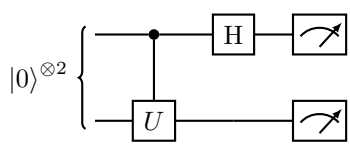


Figure 4.1: Quantum circuit with two input qubits in state $|00\rangle$ where we first apply a controlled-U gate, second the Hadamard gate, and finally both qubits are measured giving two classical bits as output

For the classical circuit model, we can define the notion of efficiency. We

say that the size of the circuit is the number of nodes in the circuit, and the circuit is considered efficient if there exists a polynomial-sized circuit with respect to the input size that computes a function. Similarly, for the quantum circuit model, we use the same notion of efficiency, and we require that the size of the quantum circuit is polynomial in the input size, where each gate acts on at most three qubits[6].

4.2 Quantum Optimization Algorithms

As mentioned previously, various types of optimization problems frequently appear in industry and academia. Examples beyond logistics and transportation are the kidney swap problem in health care, social network optimization on graphs, and quantum transpilation. We, therefore, consider it valuable, both from a theoretical standpoint and in practice, to understand if quantum algorithms that can improve upon classical algorithms designed to solve integer programs. However, since it was shown in [62], [101] that the best speedup we can expect for NP-complete problems is quadratic in the black box setting, we should perhaps consider approximate or heuristic quantum algorithms rather than exact algorithms.

Nonetheless, one can roughly categorize quantum optimization algorithms into two categories. The first category is nonheuristic algorithms. These algorithms have provable complexity behavior with respect to time and space and solution quality. In this category, we have Grover's algorithm and extensions where Grover's algorithm is embedded in a classical algorithm. The second category is heuristic algorithms, such as the Quantum Annealing Algorithm and the Quantum Approximate Optimization Algorithm, which we will discuss in the coming sections.

Adiabatic Quantum Computation and Quantum Annealing

Adiabatic quantum computation [102] is a universal paradigm of quantum computing proven to be as powerful as the circuit model discussed in Sec. 4.1 and is based on the adiabatic theorem [103]. The adiabatic theorem describes what happens to a quantum system, initialized in a non-degenerate eigenstate of an initial Hamiltonian H_0, that is changed continuously and adiabatically, or

[6] There exists a universal gate set that includes the Toffoli gate.

infinitely slowly, to a final Hamiltonian H_1. One example of such a situation is when an external magnetic field is changed slowly for an interacting quantum spin system but has also been used to construct a whole separate paradigm of computation.

We can explicitly construct a time-dependent Hamiltonian that describes this situation as

$$H(t) = \frac{(T-t)}{T} H_0 + \frac{t}{T} H_1, \tag{4.4}$$

where T is a measure of how fast the system changes and governs the evolution time from H_0 to H_1. Assuming that the two Hamiltonians H_0 and H_1 act on an n qubit system, H_0 and H_1 do not commute, that the instantaneous eigenenergies $E_0(t) < E_1(t) < \cdots < E_{2^n-1}(t)$ of $H(t)$ are distinct for the evolution time and the system is initialized in the ground state $|e_0(0)\rangle$ of $H(0) = H_0$, we have the following bound on the time required to ensure that the evolution is adiabatic

$$\frac{\max_{s\in[0,1]} |\langle e_1(s)| \partial_s H(s) |e_0(s)\rangle|}{\min_{s\in[0,1]} |E_1(s) - E_0(s)|^2} \ll T,$$

where $s = t/T$. The bound above tells us that as long the bound is respected, the initial state $|e_0(s=0)\rangle$ has evolved to $|\psi\rangle = |e_0(s=1)\rangle$ after the evolution time $t = T$.

Now we direct our attention to the fact that the adiabatic theorem can be used to construct an algorithm for solving discrete optimization problems [104], which we refer to as the Quantum Adiabatic Algorithm[7] (QAA). Solving some discrete optimization problem is achieved by encoding the problem as the final Hamiltonian H_1 such that its groundstate is the optimal solution and choosing an initial Hamiltonian H_0 where the groundstate is known, easy to construct, and does not commute with H_1. Most common is the choice $H_0 = -\sum_{i=1}^{n} \sigma_i^x$ with groundstate $|+\rangle = \sqrt{2^{-n}} \sum_{i=0}^{2^n-1} |i\rangle$, which we can prepare easily.

The caveat with QAA is that we require the evolution time to be at most polynomial in the input size n but knowing the instantaneous eigenenergies of the time-dependent Hamiltonian $H(t)$ is often as challenging as solving the optimization problem itself. It is therefore unknown for many problems

[7] The quantum adiabatic algorithm is used synonymously with the term adiabatic quantum computation.

what kind of speedup is possible, or if there even is a speedup, over classical algorithms.

The issue of analyzing the instantaneous eigenenergies has resulted in the approximate version of QAA called Quantum Annealing (QA), where the evolution time is not guaranteed to ensure adiabatic evolution. One resulting difference between QA and QAA is that we now consider the overlap $p = |\langle\psi^*|\psi\rangle|^2$ between the state $|\psi\rangle$ we have obtained after evolving the system for some time T and the desired solution $|\psi^*\rangle$. By repeating the QA algorithm

$$k = \frac{\ln(1 - p_{target})}{\ln(1 - p)}$$

times, i.e., the process of initializing the system in the ground state of the initial Hamiltonian and evolving the system for some time T to the final Hamiltonian, the probability of finding the solution can be increased to p_{target}. To date, it is also unknown for many problems if QA can provide speedup or any significant advantage over classical algorithms, but it remains nonetheless an interesting heuristic to explore.

Quantum Approximate Optimization Algorithm

QAOA is a variational hybrid quantum-classical algorithm, parameterized by the positive integer p that determines the depth of the quantum algorithm[8].

Although the algorithm is capable of universal quantum computing [105]–[108], the most common goal is to find approximate solutions to minimization (or maximization) problems. One property of QAOA discussed in [63] is that the solution quality is monotonically increasing with the parameter p, assuming the angles $\vec{\gamma}$ and $\vec{\beta}$ in Eq. (4.5) are optimal[9] and an ideal quantum computer. Furthermore, in [63] it is shown that under the assumption that the angles are small if $p \to \infty$ the algorithm becomes exact and finds the optimal solution (if the QAA can find the optimal solution).

A QAOA circuit creates, in an ideal setting, the state

$$|\vec{\gamma}, \vec{\beta}\rangle = \prod_{i=0}^{p-1} U_m(\beta_{p-i}, H_m) U_c(\gamma_{p-i}, H_c) |\phi\rangle_{initial} \tag{4.5}$$

[8] The algorithm depth of QAOA is also called the number of layers in literature.

[9] The angles are required to be the optimal solution to the problem defined in Eq.(4.7)-(4.9). Hence for a minimization problem $M_p \leq M_{p-1}$.

where we, for each layer i, first apply the unitary cost operator U_c and second the mixer operator U_m that acts on n qubits. The most common choice of operators has been $U_m = e^{-i\beta H_m}$ and $U_c = e^{-i\gamma H_c}$, where we refer to H_m and H_c as the mixer and cost Hamiltonian, respectively. Since NP-complete problems can be encoded into an Ising spin glass Hamiltonian [109], the cost Hamiltonian is often given explicitly in this form, that is, $H_c = \sum_{i\in V} h_i\sigma_i^z + \sum_{\{i,j\}\in E} J_{ij}\sigma_i^z\sigma_j^z$. Moreover, we note that the goal of finding the solution to a minimization problem is the same as the goal of finding the ground state of the cost Hamiltonian. Here, we view H_c as representing an undirected graph $G = (V, E)$ with vertex set V and edge set E. For each edge $\{i, j\} \in E$, there exists a weight J_{ij}, and for each vertex $i \in V$ there is an associated weight h_i. In some ways, this makes various problems defined on undirected graphs particularly intuitive since we must, for example, translate an integer linear program to the Ising spin glass Hamiltonian, and it is not always straightforward how properties of the integer linear program are connected to graph properties. For example, we can state that we consider QAOA for a MaxCut problem with three regular graphs and the underlying graph to the cost Hamiltonian has that specified property.

Next, we will discuss the choice of mixer Hamiltonian. The choice is, in principle, free but the mixer operator should somehow be capable of connecting the initial state to states we accept as a solution. The most commonly[10] seen mixer Hamiltonian is $H_m = \sum_{i=1}^{n} \sigma_i^x$ and a natural initial state is then $|+\rangle = \sqrt{2^{-n}} \sum_{i=0}^{2^n-1} |i\rangle$. However, other choices of mixer Hamiltonian and initial state can be beneficial, as discussed in [110], by restricting QAOA into some subspace of the whole Hilbert space.

By considering the expectation value function

$$E_p(\vec{\gamma}, \vec{\beta}) = \langle\vec{\gamma}\vec{\beta}|H_c|\vec{\gamma}\vec{\beta}\rangle \tag{4.6}$$

[10] When choosing this mixer Hamiltonian, the algorithm is sometimes called vanilla QAOA and refers to the paper by Farhi et al. [63]. Furthermore, the initial state chosen with this mixer Hamiltonian is $|+\rangle$ and is the ground state of $-H_m$ that is typically the initial Hamiltonian in the Adiabatic Quantum Algorithm.

and the non-linear continuous optimization problem

$$M_p = \text{minimize} \quad E_p(\vec{\gamma}, \vec{\beta}), \tag{4.7}$$
$$\text{subject to} \quad \vec{\gamma} \in D_\gamma, \tag{4.8}$$
$$\vec{\beta} \in D_\beta, \tag{4.9}$$

for some fixed p, the probability of obtaining a string that is either the optimum or some distance away when measuring is high provided that p is sufficiently large and that the optimization problem in Eq. (4.7)-(4.9) can be solved. Thus, if a measurement is performed in the computational basis, we obtain a solution candidate string $\vec{z} \in \{-1, +1\}^n$, which can be evaluated for the cost Hamiltonian. I.e., if the process of (1) constructing the QAOA state with optimal angles and (2) measuring the state in the computational basis is repeated sufficiently many times, we should obtain a solution string that is near the expectation value function for the fixed angles.

We also note that for the mixer operator $U_m = \prod_{i=1}^{n} e^{-i\beta\sigma_i^x}$ the domain defined in Eq.(4.9) becomes $[0, \pi]^{\times p}$ and if the cost Hamiltonian has integer eigenvalues the domain in Eq. (4.8) is $[0, 2\pi]^{\times p}$. A priori, what depth of the

Algorithm 1 QAOA

Input: $p \geq 1$

$(\vec{\gamma}, \vec{\beta}) \leftarrow$ solve Eq. (4.7)-(4.9)

Construct the state $|\vec{\gamma}, \vec{\beta}\rangle$

Measure $|\vec{\gamma}, \vec{\beta}\rangle$ in the computational basis

Repeat two former steps N times

Output: Best solution string found

algorithm is sufficient is not known for many optimization problems, which can cause issues when noise is present in the system. In practice, we might therefore need to increase the algorithm depth until some condition holds. However, the QAOA algorithm can be simply stated as in Alg. 1.

The issue of solving Eq. (4.7)-(4.9) appears daunting, as this problem is NP-hard [111] and we plan to solve it with a classical algorithm that queries a quantum computer to get values $E_p(\vec{\beta}, \vec{\gamma})$ for angles in their respective domains.

We can consider a few different solution approaches. One approach is to ob-

tain a closed-form expression of the expectation value function. In that case, we can either find good angles by numerical optimization methods without a quantum computer or possibly determine good angles without using numerical optimization techniques by analyzing the closed-form expression [63], [112]. A second approach is to approximately simulate a quantum computer with, for example, matrix product states and tensor networks as in [113]. A third approach is to utilize machine learning techniques [114]–[116] and other numerical optimization techniques where each query of the expectation value function is obtained by using a quantum computer. Here there have been proposals that can reduce the number of queries we require by finding good angles for small instances and using them for larger instances and/or unseen instances and interpolating angles [117] for larger algorithm depths. Many of these proposals use the fact that the angles of QAOA appear to concentrate for certain problems and distributions, see [112], [118]. In this thesis, we have focused on the interpolation strategy, and note that since this problem is NP-hard, in general, we might not expect to have access to optimal angles.

At the time of writing this thesis, it seems that we are not sure if QAOA is capable of solving problems better and/or faster than the best classical algorithms. Although we have strong evidence that a classical computer can't simulate QAOA exactly [119], this does not say anything about what problems QAOA can solve or how much resources, with respect to time, the algorithm requires. Some interesting performance results have indeed shown that a classical algorithm outperforms QAOA or achieves the same approximation ratio for problems such as MaxCut [120]–[122] and MAX-3-XOR [123]. On the other hand, QAOA was observed to outperform a classical algorithm for MAX-k-XOR when $k > 4$ [124]. Furthermore, Farhi has been able to analyze QAOA extensively for Maximum Independent Set [125] and the Sherrington-Kirkpatrick model [112]. Results like these are vital for our understanding of the algorithm, and more such results are desirable. Notably, many of these results are restricted to constant algorithm depth or logarithmic depth in the number of qubits n. Negative results with restricted algorithm depth and restricted graphs are thus not excluding QAOA from outperforming classical algorithms for greater algorithm depths and other graph structures, and the ultimate question of whether QAOA can be advantageous is still open.

Finally, we would like to mention the issue of noise for QAOA [126]–[130]. If we indeed need to have logarithmic or polynomial algorithm depth, as we

increase the algorithm depth noise, will be more important to consider as shown in [131], [132] where noise deprecates QAOA's performance. It also seems likely that establishing error mitigation techniques can be helpful or required [133].

CHAPTER 5

Summary of Papers

In this chapter, we give a summary of the appended papers. Papers A, C, and D are concerned with the algorithm QAOA. Paper B, in contrast, is related to the algorithm Quantum Annealing.

5.1 Paper A

In this work, we proposed a hybrid quantum-classical heuristic algorithm that augments the classical Branch-and-Price algorithm. Branch-and-Price is augmented in a similar fashion as in [134] with the classical integer program solver PAQS [135]. The main distinction here is that we propose to use a quantum algorithm to solve the current integer program, which is the integer version of the RMP. Although the heuristic is believed to be useful for several real-world problems as it is tied to Branch-and-Price, we naturally explored the method for extracted and simplified Tail Assignment RMP instances, as the focus of this thesis is quantum algorithms for airline scheduling problems.

Consequently, the problems we considered were both Exact Cover and Set Partitioning, and QAOA was the considered quantum algorithm. The results were obtained for an ideal quantum computer via numerical simulations of

QAOA circuits. It was found that balancing the objective and constraint parts of the Hamiltonian is important to reach a better performance for QAOA when attempting to solve Set Partitioning and that setting the penalty unnecessarily high can lead to an increased requirement on the algorithm depth.

It was also found for Exact Cover that QAOA, in general, requires lower algorithm depth as the number of feasible solutions increases. This coincides with the fact that the average node degree of the underlying graph decreases. In particular, we also observed this effect for the Set Partitioning problem, where we only accepted the optimal solution. In Paper C, it was found that a higher average node degree coincided with a worse performance of QAOA. This means that the numerical results in both papers point to the fact that the node degree can affect the performance of QAOA.

5.2 Paper B

This paper evaluated RMP instances extracted from Tail Assignment ranging from small to intermediate size for the Quantum Annealing algorithm on the D-wave machines Advantage and 2000Q. 2000Q and Advantage were compared for instances up to 100 decision variables, which is considerably larger than the instances we studied in Paper A. Instances with 120 decision variables were also studied with Advantage, but not possible to solve with 2000Q. The instances were both sparsely connected and close to fully connected, allowing us to analyze how the graph density and instance size affect the performance of both machines.

The results show that the new and larger machine Advantage solves the integer program instances in close to half the time required by 2000Q, with respect to programming and readout time. In Fig. 3, the annealing time is varied from 1-2000 μs against the success rate, for which Advantage outperforms 2000Q for most of the instances (with the exception of some of the smaller sparse graphs), meaning that the annealing time is shorter for Advantage compared to 2000Q. The results indicate that the connectivity of the machine's topology, which is higher in Advantage compared to 2000Q, is one important factor that enables Advantage to be superior to 2000Q. Thus, showing that quantum annealing could be useful in practice when the hardware is scaled up in size.

5.3 Paper C

Here, we studied the success probability of QAOA for Exact Cover instances with exactly one solution derived from Tail Assignment. The results were obtained for an ideal quantum computer via numerical simulations of quantum circuits. It was shown that the interpolation strategy presented in [117] could be utilized for the Exact Cover instances and that QAOA could, in the ideal case, give near unit success probability for an algorithm depth that was smaller than the number of qubits (i.e., instance sizes). It was also found that the performance of QAOA decreased for instances with a high average node degree compared to instances with a lower average node degree.

5.4 Paper D

Here, we implemented QAOA on a quantum processor consisting of superconducting transmon qubits for Exact Cover instances with two decision variables. We experimentally investigated the algorithm and processor for one and two layers, demonstrating that the success probability increased, as expected, as the algorithm depth increased. The maximum probability obtained was 96.6% for algorithm depth 2, where theory predicted 96.3% when gate fidelities were considered, compared to the ideal case, which predicted 100% success probability. Thus, the results show agreement between experiments and theory in the energy landscapes and algorithm performance, indicating low error rates.

CHAPTER 6

Concluding Remarks and Future Work

This chapter summarizes the conclusions, starting with Paper A and C since they are highly connected. We then give the conclusions for Paper B and D that consider existing devices. Finally, we discuss future opportunities related to Multi-Commodity Network Flow problems.

6.1 Paper A and C

Although the numerical results indicate that we, in many cases, can find feasible solutions and even the optimal solution for small instances of Set Partitioning and Exact Cover with polynomial algorithm depth, we recognize that these sizes are orders of magnitude smaller than the problems solved in practice. Our results can, therefore, not be compared to classical solvers in any meaningful way yet, nor can they arbitrarily be extrapolated to larger instance sizes.

To understand what algorithm depth is required for larger instances, a larger quantum device and/or constructing a mathematical proof of the required algorithm depth is needed. This feat has been achieved in [125] and [136] for Maximum Independent Set, for example. Such problems fall into the

category of Ising models where all h_i terms are zero and all edge terms J_{ij} are one. As we, and others as far as we know, have yet to be successful in analyzing the behavior of QAOA by analytical means in a more general setting, it would be a highly valuable result to obtain. One possible avenue to achieve this is to find characteristics in Set Partitioning and Exact Cover that are related to characteristics of the underlying Ising spin glass Hamiltonian graph. Another possible method is to explore if Exact Cover and Set Partitioning have the overlap gap property, as this is exploited in the proof for Maximum Independent Set.

Whilst understanding the performance of QAOA for a general Ising spin glass Hamiltonian in the ideal setting remains an important open question, there is one important aspect that can depreciate the performance of QAOA, and that is noise. Furthering the understanding of noise as in [126]–[130], gives more insight into if QAOA truly is, or can be, noise resilient.

Other variants of QAOA as the Quantum Alternating Operator Ansatz [132], warm starting QAOA or RQAOA can also be interesting to investigate. We can, for example, view RQAOA [122] as an error mitigation technique as it reduces the instance size in each iteration and can possibly shorten the algorithm depth, beyond the fact that some evidence has been presented that RQAOA also can outperform QAOA for a certain problem of any size. Introducing further constraints as is done in the Quantum Alternating Operator Ansatze by fixing the Hamming weight of the solution string can also prove to be fruitful for some problems. We believe that constructing new initial states and mixing operators will continue to be an interesting research direction.

Finally, it would be interesting to understand the amount of entanglement that exists in QAOA circuits, this has to some extent been studied in [137], [138], but remains an important open question.

6.2 Paper B

The benchmark results obtained from both Advantage and 2000Q demonstrated that Quantum Annealing machines could solve intermediate integer program problems. The results also showed that the machines perform better when they are scaled up in size and have improved connectivity. However, we are still lacking knowledge regarding the instantaneous energy gap for problems such as Exact Cover. This could be a future research possibility as well

as conducting further empirical studies for average cases of Set Partitioning, Exact Cover, and Set Cover. In particular, if we in the future can embed problems with nearly 1000 decision variables, more interesting and realistic distributions are possible to study.

6.3 Paper D

The demonstration of toy problems on a superconducting quantum processor showed the quality of the device. It does not, however, say anything about the performance of any problem of interest. A future research possibility could be to use larger systems available, e.g., IBM's quantum processor, and possibly introduce some error correcting scheme as discussed in [132].

6.4 Quantum Algorithms and Integer Network Flows

Thus far, we have focused on the near-term gate-based algorithm QAOA and quantum annealing to some extent. However, the nature of many airline scheduling problems is such that there exist a vast number of constraints, and the number of variables is large, both in a Branch-and-Price augmented scheme and in an arc-flow formulation. It might accordingly be worthwhile to question the usefulness of variational algorithms for such large problems in the long-term development of quantum computers. It can therefore be interesting to consider fault-tolerant algorithms, such as Montanari's Branch-and-Bound algorithm [139], or other algorithms that are based on Dürr and Hoyer's search algorithm [140]. Such algorithm ideas have been presented by Ambainis for maximum flow in networks in [141], which of course is not applicable to multi-commodity network flows at this point.

To summarize, more effort is required to understand simple multi-commodity network flow problems in relation to quantum algorithms in various decompositions as these problems model airline scheduling problems.

References

[1] C. A. Floudas and P. M. Pardalos, *Encyclopedia of Optimization*. Springer US, 2009, ISBN: 9780387747590.

[2] G. Yu and B. Thengvall, "Airline optimization," in *Encyclopedia of Optimization*, C. A. Floudas and P. M. Pardalos, Eds. Springer US, 2009, pp. 26–30, ISBN: 978-0-387-74759-0.

[3] P. Belobaba, A. Odoni, C. Barnhart, and P. Belobaba, *The Global Airline Industry* (Aerospace Ser). John Wiley & Sons, Incorporated, 2015, ISBN: 9781118881149.

[4] C. Barnhart and A. Cohn, "Airline schedule planning: Accomplishments and opportunities," *Manufacturing & Service Operations Management*, vol. 6, no. 1, pp. 3–22, 2004, ISSN: 15234614.

[5] A. E. E. Eltoukhy, F. T. S. Chan, and S. H. Chung, "Airline schedule planning: A review and future directions," *Industrial Management & Data Systems*, vol. 117, no. 6, pp. 1201–1243, 2017, ISSN: 02635577.

[6] S. Yan and H.-F. Young, "A decision support framework for multi-fleet routing and multi-stop flight scheduling," *Transportation Research Part A: Policy and Practice*, vol. 30, no. 5, pp. 379–398, 1996, ISSN: 0965-8564.

[7] K. Wei, V. Vaze, and A. Jacquillat, "Airline timetable development and fleet assignment incorporating passenger choice," *Transportation Science*, vol. 54, no. 1, pp. 139–163, 2020, ISSN: 00411655.

[8] S. Yan and C.-H. Tseng, "A passenger demand model for airline flight scheduling and fleet routing," *Computers & Operations Research*, vol. 29, no. 11, pp. 1559–1581, 2002, ISSN: 0305-0548.

[9] L. H. Lee, C. U. Lee, and Y. P. Tan, "A multi-objective genetic algorithm for robust flight scheduling using simulation," *European Journal of Operational Research*, vol. 177, no. 3, pp. 1948–1968, 2007, ISSN: 0377-2217.

[10] S. Yan, C.-H. Tang, and M.-C. Lee, "A flight scheduling model for taiwan airlines under market competitions," *Omega*, vol. 35, no. 1, pp. 61–74, 2007, ISSN: 0305-0483.

[11] S. Yan, C.-H. Tang, and T.-C. Fu, "An airline scheduling model and solution algorithms under stochastic demands," *European Journal of Operational Research*, vol. 190, no. 1, pp. 22–39, 2008, ISSN: 0377-2217.

[12] J. Hai and C. Barnhart, "Dynamic airline scheduling," *Transportation Science*, vol. 43, no. 3, pp. 336–354, 2009, ISSN: 00411655.

[13] M. Sohoni, L. Yu-Ching, and D. Klabjan, "Robust airline scheduling under block-time uncertainty," *Transportation Science*, vol. 45, no. 4, pp. 451–464, 2011, ISSN: 00411655.

[14] H. Jiang and C. Barnhart, "Robust airline schedule design in a dynamic scheduling environment," *Computers and Operations Research*, vol. 40, no. 3, pp. 831–840, 2013, ISSN: 0305-0548.

[15] B. Kepir, Ç. Koçyiğit, I. Koyuncu, M. B. Özer, B. Y. Kara, and M. A. Gürbüz, "Flight-scheduling optimization and automation for anadolujet," *Interfaces*, vol. 46, no. 4, pp. 315–325, 2016, ISSN: 00922102.

[16] A. Levin, "Scheduling and fleet routing models for transportation systems," *Transportation Science*, vol. 5, no. 3, pp. 232–255, 1971, ISSN: 00411655.

[17] C. Barnhart, T. Kniker, and M. Lohatepanont, "Itinerary-based airline fleet assignment," *Transportation Science*, vol. 36, no. 2, pp. 199–217, 2002, ISSN: 00411655.

[18] J. Rosenberger, E. Johnson, and G. Nemhauser, "A robust fleet-assignment model with hub isolation and short cycles," *Transportation Science*, vol. 38, no. 3, pp. 357–368, 2004, ISSN: 00411655.

[19] H. Sherali, E. Bish, and X. Zhu, "Polyhedral analysis and algorithms for a demand-driven refleeting model for aircraft assignment," *Transportation Science*, vol. 39, no. 3, pp. 349–366, 2005, ISSN: 15265447.

[20] N. Bélanger, G. Desaulniers, F. Soumis, J. Desrosiers, and J. Lavigne, "Weekly airline fleet assignment with homogeneity," *Transportation Research Part B*, vol. 40, no. 4, pp. 306–318, 2006, ISSN: 0191-2615.

[21] B. Smith and E. Johnson, "Robust airline fleet assignment: Imposing station purity using station decomposition," *Transportation Science*, vol. 40, no. 4, pp. 497–516, 2006, ISSN: 15265447.

[22] T. Jacobs, B. Smith, and E. Johnson, "Incorporating network flow effects into the airline fleet assignment process," *Transportation Science*, vol. 42, no. 4, pp. 514–529, 2008, ISSN: 15265447.

[23] J. Dumas, F. Aithnard, and F. Soumis, "Improving the objective function of the fleet assignment problem," *Transportation Research Part B*, vol. 43, no. 4, pp. 466–475, 2009, ISSN: 0191-2615.

[24] V. L. Pilla, J. M. Rosenberger, V. Chen, N. Engsuwan, and S. Siddappa, "A multivariate adaptive regression splines cutting plane approach for solving a two-stage stochastic programming fleet assignment model," *European Journal of Operational Research*, vol. 216, no. 1, pp. 162–171, 2012, ISSN: 0377-2217.

[25] D. T. Sanchez, B. Boyacı, and K. G. Zografos, "An optimisation framework for airline fleet maintenance scheduling with tail assignment considerations," *Transportation Research Part B*, vol. 133, pp. 142–164, 2020, ISSN: 0191-2615.

[26] J. Abara, "Applying integer linear programming to the fleet assignment problem," *Interfaces*, vol. 19, no. 4, pp. 20–28, 1989, ISSN: 00922102.

[27] M. E. Berge and C. A. Hopperstad, "Demand driven dispatch. a method for dynamic aircraft capacity assignment, models and algorithms," *Operations Research*, vol. 41, no. 1, pp. 153–168, 1993, ISSN: 0030364X.

[28] C. A. Hane, C. Barnhart, E. L. Johnson, R. E. Marsten, G. L. Nemhauser, and G. Sigismondi, "The fleet assignment problem: Solving a large-scale integer program," English, *Mathematical Programming*, vol. 70, no. 2, pp. 211–232, 1995.

[29] K. Talluri, "Swapping applications in a daily airline fleet assignment," *Transportation Science*, vol. 30, no. 3, pp. 237–248, 1996, ISSN: 00411655.

[30] G. Desaulniers, J. Desrosiers, Y. Dumas, M. Solomon, and F. Soumis, "Daily aircraft routing and scheduling," *Management Science*, vol. 43, no. 6, pp. 841–855, 1997, ISSN: 00251909.

[31] R. A. Rushmeier and S. A. Kontogiorgis, "Advances in the optimization of airline fleet assignment," *Transportation Science*, vol. 31, no. 2, p. 159, 1997, ISSN: 00411655.

[32] A. Jarrah, J. Goodstein, and R. Narasimhan, "Efficient airline refleeting model for the incremental modification of planned fleet assignments," *Transportation Science*, vol. 34, no. 4, pp. 349–363, 2000, ISSN: 00411655.

[33] B. Rexing, C. Barnhart, T. Kniker, A. Jarrah, and N. Krishnamurthy, "Airline fleet assignment with time windows," *Transportation Science*, vol. 34, no. 1, pp. 1–20, 2000, ISSN: 00411655.

[34] J. Arabeyre, J. Fearnley, F. Steiger, and W. Teather, "The airline crew scheduling problem: A survey.," *Transportation Science*, vol. 3, no. 2, pp. 140–163, 1969, ISSN: 00411655.

[35] C. Barnhart, A. M. Cohn, E. L. Johnson, D. Klabjan, G. L. Nemhauser, and P. H. Vance, "Airline crew scheduling," in *Handbook of Transportation Science*, R. W. Hall, Ed. Boston, MA: Springer US, 2003, pp. 517–560, ISBN: 978-0-306-48058-4.

[36] X. Wen, X. Sun, Y. Sun, and X. Yue, "Airline crew scheduling: Models and algorithms," *Transportation Research Part E*, vol. 149, 2021, ISSN: 1366-5545.

[37] M. Deveci and N. Ç. Demirel, "A survey of the literature on airline crew scheduling," *Engineering Applications of Artificial Intelligence*, vol. 74, pp. 54–69, 2018, ISSN: 0952-1976.

[38] M. Grönkvist, *The tail assignment problem* (Ph.D. dissertation). Chalmers tekniska högskola, 2005, ISBN: 9172916451.

[39] O. Khaled, M. Minoux, V. Mousseau, S. Michel, and X. Ceugniet, "A compact optimization model for the tail assignment problem," *European Journal of Operational Research*, vol. 264, no. 2, pp. 548–557, 2018, ISSN: 0377-2217.

[40] Z. Liang, Y. Feng, X. Zhang, T. Wu, and W. A. Chaovalitwongse, "Robust weekly aircraft maintenance routing problem and the extension to the tail assignment problem," *Transportation Research Part B*, vol. 78, pp. 238–259, 2015, ISSN: 0191-2615.

[41] S. J. Maher, G. Desaulniers, and F. Soumis, "The daily tail assignment problem under operational uncertainty using look-ahead maintenance constraints," *European Journal of Operational Research*, vol. 264, no. 2, pp. 534–547, 2018, ISSN: 0377-2217.

[42] M. Fuentes, L. Cadarso, V. Vaze, and C. Barnhart, "The tail assignment problem: A case study at vueling airlines," *Transportation Research Procedia*, vol. 52, pp. 445–452, 2021, ISSN: 2352-1465.

[43] L. Lettovsky, "Airline operations recovery: An optimization approach," Ph.D. dissertation, School of Industrial Systems Engineering, Georgia Institute of Technology, Atlanta, GA, USA, 1997.

[44] N. Kohl, A. Larsen, J. Larsen, A. Ross, and S. Tiourine, "Airline disruption management—perspectives, experiences and outlook," *Journal of Air Transport Management*, vol. 13, no. 3, pp. 149–162, 2007, ISSN: 0969-6997.

[45] J. Sakurai and J. Napolitano, *Modern Quantum Mechanics*. Sep. 2020, ISBN: 9781108473224.

[46] M. R. Garey and D. S. Johnson, *Computers and intractability : a guide to the theory of NP-completeness* (A series of books in the mathematical sciences). Freeman, 1979, ISBN: 0716710455.

[47] S. Arora and B. Barak, *Computational complexity : a modern approach*. Cambridge University Press, 2009, ISBN: 9780521424264.

[48] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, *Complexity and Approximation. : Combinatorial Optimization Problems and Their Approximability Properties*. Springer Berlin Heidelberg, 1999, ISBN: 9783642584121.

[49] G. E. Moore, "Cramming more components onto integrated circuits, reprinted from electronics, volume 38, number 8, april 19, 1965, pp.114 ff," *IEEE Solid-State Circuits Society Newsletter, Solid-State Circuits Society Newsletter, IEEE, IEEE Solid-State Circuits Soc. Newsl*, vol. 11, no. 3, pp. 33–35, 2006, ISSN: 1098-4232.

[50] G. Moore, "Progress in digital integrated electronics [technical literaiture, copyright 1975 ieee. reprinted, with permission. technical digest. international electron devices meeting, ieee, 1975, pp. 11-13.]," *IEEE Solid-State Circuits Society Newsletter, Solid-State Circuits Society Newsletter, IEEE, IEEE Solid-State Circuits Soc. Newsl*, vol. 11, no. 3, pp. 36–37, 2006, ISSN: 1098-4232.

[51] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines," *Journal of Statistical Physics*, vol. 22, no. 5, pp. 563–591, 1980.

[52] Y. Manin, *Computable and Non-Computable (in Russian)*. Moscow: Sovetskoye Radio, 1980.

[53] R. P. Feynman, "Simulating physics with computers," *International journal of theoretical physics*, vol. 21, no. 6/7, pp. 467–488, 1982.

[54] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge University Press, 2010, ISBN: 9781107002173.

[55] J. D. Hidary, *Quantum Computing: An Applied Approach*. Springer International Publishing, 2019, ISBN: 9783030239220.

[56] D. Deutsch, "Quantum theory, the church–turing principle and the universal quantum computer," *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 400, pp. 117–97, 1985.

[57] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation," *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 439, pp. 553–558, 1992.

[58] E. Bernstein and U. Vazirani, "Quantum complexity theory," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1411–1473, 1997.

[59] D. R. Simon, "On the power of quantum computation," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1474–1483, 1997.

[60] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997.

[61] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978, ISSN: 0001-0782.

[62] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, ser. STOC '96, Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 212–219, ISBN: 0897917855.

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

[64] A. Peruzzo, J. McClean, P. Shadbolt, *et al.*, "A variational eigenvalue solver on a photonic quantum processor," *Nature Communications*, vol. 5, no. 1, Jul. 2014.

[65] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018.

[66] W. I-Lin, "Multicommodity network flows: A survey, part i: Applications and formulations," *International Journal of Operations Research*, vol. 15, no. 4, pp. 145–153, 2018, ISSN: 1813-713X.

[67] W. I-Lin, "Multicommodity network flows: A survey, part ii: Solution methods," *International Journal of Operations Research*, vol. 15, no. 4, pp. 155–173, 2018, ISSN: 1813-713X.

[68] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network flows : theory, algorithms and applications*. Prentice Hall, 1993, ISBN: 013617549X.

[69] R. Garfinkel and G. Nemhauser, *Integer programming* (Series in decision and control). John Wiley, n.d, ISBN: 0-471-29195-1.

[70] G. Dantzig, A. Orden, and P. Wolfe, "The generalized simplex method for minimizing a linear form under linear inequality restraints," *Pacific Journal of Mathematics*, vol. 5, Jun. 1955.

[71] G. B. Dantzig and P. Wolfe, "Decomposition principle for linear programs," *Operations Research*, vol. 8, no. 1, pp. 101–111, 1960, ISSN: 0030364X.

[72] L. S. Lasdon, *Optimization theory for large systems*. Dover Publications, 2002, ISBN: 0486419991.

[73] M. E. Lübbecke and J. Desrosiers, "Selected topics in column generation," *Operations Research*, vol. 53, no. 6, pp. 1007–1023, Nov. 2005, ISSN: 0030-364X.

[74] A. H. Land and A. G. Doig, "An automatic method for solving discrete programming problems," *ECONOMETRICA*, vol. 28, no. 3, pp. 497–520, 1960.

[75] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, "Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning," *Discrete Optimization*, vol. 19, pp. 79–102, 2016, ISSN: 1572-5286.

[76] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations Research*, vol. 46, Feb. 1970.

[77] M. Jünger, M. Elf, and V. Kaibel, "Rotation planning for the continental service of a european airline," in *Mathematics — Key Technology for the Future: Joint Projects between Universities and Industry*, W. Jäger and H.-J. Krebs, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 675–689, ISBN: 978-3-642-55753-8.

[78] M. Daskin and N. Panayotopoulos, "A lagrangian relaxation approach to assigning aircraft to routes in hub and spoke networks," *Transportation Science*, vol. 23, pp. 91–99, May 1989.

[79] G. Desaulniers, J. Desrosiers, Y. Dumas, M. Solomon, and F. Soumis, "Daily aircraft routing and scheduling," *Management Science*, vol. 43, pp. 841–855, Jun. 1997.

[80] J.-F. Cordeau, G. Stojkovic, F. Soumis, and J. Desrosiers, "Benders decomposition for simultaneous aircraft routing and crew scheduling," *Transportation Science*, vol. 35, pp. 375–388, Nov. 2001.

[81] M. Bartholomew-Biggs, S. Parkhurst, and S. Wilson, "Global optimization approaches to an aircraft routing problem," *European Journal of Operational Research*, vol. 146, pp. 417–431, Apr. 2003.

[82] R. Gopalan and K. T. Talluri, "The aircraft maintenance routing problem," *Operations Research*, vol. 46, pp. 260–271, 1998.

[83] A. Sarac, R. Batta, and C. M. Rump, "A branch-and-price approach for operational aircraft maintenance routing," *European Journal of Operational Research*, vol. 175, pp. 1850–1869, 2006.

[84] Z. Liang, W. A. Chaovalitwongse, H. C. Huang, and E. L. Johnson, "On a new rotation tour network model for aircraft maintenance routing problem," *Transportation Science*, vol. 45, pp. 109–120, 2011.

[85] Z. Liang and W. A. Chaovalitwongse, "A network-based model for the integrated weekly aircraft maintenance routing and fleet assignment problem," *Transportation Science*, vol. 47, pp. 493–507, 2013.

[86] N. M. Kabbani and B. W. Patty, "Aircraft routing at american airlines.," in *proceedings of the agifors symposium*, 1992.

[87] K. Talluri, "Swapping applications in a daily airline fleet assignment," *Transportation Science*, vol. 30, pp. 237–248, Aug. 1996.

[88] K. T. Talluri, "The four-day aircraft maintenance routing problem," *Transportation Science*, vol. 32, pp. 43–53, 1998.

[89] T. Feo and J. Bard, "Flight scheduling and maintenance base planning," *Management Science*, vol. 35, pp. 1415–1432, Dec. 1989.

[90] L. Clarke, E. Johnson, G. Nemhauser, and Z. Zhu, "The aircraft rotation problem," *Annals of Operations Research*, vol. 69, Jan. 1997.

[91] C. Sriram and A. Haghani, "An optimization model for aircraft maintenance scheduling and re-assignment," *Transportation Research Part A: Policy and Practice*, vol. 37, pp. 29–48, Jan. 2003.

[92] C. Barnhart, N. Boland, L. W. Clarke, E. L. Johnson, G. L. Nemhauser, and R. G. Shenoi, "Flight string models for aircraft fleeting and routing," *Transportation Science*, vol. 32, pp. 208–220, 1998.

[93] J. Bard and I. Cunningham, "Improving through-flight schedules," *IIE Transactions*, vol. 19, pp. 242–251, Sep. 1987.

[94] A. Jarrah and J. Strehler, "An optimization model for assigning through flights," *IIE Transactions*, vol. 32, pp. 237–244, Mar. 2000.

[95] R. Ahuja, J. Goodstein, J. Orlin, and D. Sharma, "A very large-scale neighborhood search algorithm for the combined through-fleet-assignment model," *Massachusetts Institute of Technology (MIT), Sloan School of Management, Working papers*, vol. 19, Jan. 2003.

[96] R. K. Ahuja, J. Liu, J. Goodstein, A. Mukherjee, J. B. Orlin, and D. Sharma, "Solving multi-criteria through-fleet assignment models," in *Operations Research in Space and Air*, T. A. Ciriani, G. Fasano, S. Gliozzi, and R. Tadei, Eds. Boston, MA: Springer US, 2003, pp. 233–256, ISBN: 978-1-4757-3752-3.

[97] R. Ahuja, J. Liu, J. Orlin, and J. Goodstein, "A neighborhood search algorithm for the combined through and fleet assignment model with time windows," *Networks*, vol. 44, pp. 160–171, Sep. 2004.

[98] M. Danielsson and G. Karlsson, *The tail assignment problem for single and mixed aircraft fleets: Mathematical modelling, solution, and implementation* (master thesis). Chalmers tekniska högskola / Institutionen för matematiska vetenskaper, 2018.

[99] D. Deutsch, "Quantum computational networks," *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 425, no. 1868, pp. 73–90, 1989, ISSN: 00804630.

[100] A. Chi-Chih Yao, "Quantum circuit complexity," in *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, 1993, pp. 352–361.

[101] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, "Strengths and weaknesses of quantum computing," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1510–1523, Oct. 1997.

[102] T. Albash and D. A. Lidar, "Adiabatic quantum computation," *Reviews of Modern Physics*, vol. 90, no. 1, Jan. 2018.

[103] A. Messiah, *Quantum mechanics*. North-Holland, 1961.

[104] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, *Quantum computation by adiabatic evolution*, 2000. arXiv: quant-ph/0001106.

[105] S. Lloyd, *Quantum approximate optimization is computationally universal*, 2018. arXiv: 1812.11075.

[106] M. E. S. Morales, J. D. Biamonte, and Z. Zimborás, "On the universality of the quantum approximate optimization algorithm," *Quantum Information Processing*, vol. 19, no. 9, Aug. 2020.

[107] J. C. Aguma, *An upper bound on the universality of the quantum approximate optimization algorithm*, 2021. arXiv: 2104.01993.

[108] H. Zheng, Z. Li, J. Liu, S. Strelchuk, and R. Kondor, *Speeding up learning quantum states through group equivariant convolutional quantum ansätze*, 2021. arXiv: 2112.07611.

[109] A. Lucas, "Ising formulations of many NP problems," *Frontiers in Physics*, vol. 2, 2014.

[110] S. Hadfield, Z. Wang, B. O'Gorman, E. Rieffel, D. Venturelli, and R. Biswas, "From the quantum approximate optimization algorithm to a quantum alternating operator ansatz," *Algorithms*, vol. 12, no. 2, p. 34, Feb. 2019.

[111] L. Bittel and M. Kliesch, "Training variational quantum algorithms is NP-hard," *Physical Review Letters*, vol. 127, no. 12, Sep. 2021.

[112] 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.

[113] M. Streif and M. Leib, *Training the quantum approximate optimization algorithm without access to a quantum processing unit*, 2019. arXiv: 1908.08862.

[114] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev, and P. Balaprakash, "Learning to optimize variational quantum circuits to solve combinatorial problems," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 03, pp. 2367–2375, Apr. 2020.

[115] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev, and P. Balaprakash, *Reinforcement-learning-based variational quantum circuits optimization for combinatorial problems*, 2019. arXiv: 1911.04574.

[116] D. Wecker, M. B. Hastings, and M. Troyer, "Training a quantum optimizer," *Physical Review A*, vol. 94, no. 2, Aug. 2016.

[117] 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," *Physical Review X*, vol. 10, no. 2, Jun. 2020, ISSN: 2160-3308.

[118] F. G. S. L. Brandao, M. Broughton, E. Farhi, S. Gutmann, and H. Neven, *For fixed control parameters the quantum approximate optimization algorithm's objective function value concentrates for typical instances*, 2018. arXiv: 1812.04170.

[119] E. Farhi and A. W. Harrow, *Quantum supremacy through the quantum approximate optimization algorithm*, 2016. arXiv: 1602.07674.

[120] M. B. Hastings, *Classical and quantum bounded depth approximation algorithms*, 2019. arXiv: 1905.07047.

[121] K. Marwaha, "Local classical max-cut algorithm outperforms p=2 qaoa on high-girth regular graphs," *Quantum*, vol. 5, p. 437, Apr. 2021.

[122] S. Bravyi, A. Kliesch, R. Koenig, and E. Tang, "Obstacles to variational quantum optimization from symmetry protection," *Physical Review Letters*, vol. 125, no. 26, Dec. 2020.

[123] B. Barak, A. Moitra, R. O'Donnell, *et al.*, *Beating the random assignment on constraint satisfaction problems of bounded degree*, 2015. arXiv: 1505.03424.

[124] K. Marwaha and S. Hadfield, "Bounds on approximating max kxor with quantum and classical local algorithms," *Quantum*, vol. 6, p. 757, Jul. 2022.

[125] E. Farhi, D. Gamarnik, and S. Gutmann, *The quantum approximate optimization algorithm needs to see the whole graph: A typical case*, 2020. arXiv: 2004.09002.

[126] J. R. McClean, M. E. Kimchi-Schwartz, J. Carter, and W. A. de Jong, "Hybrid quantum-classical hierarchy for mitigation of decoherence and determination of excited states," *Physical Review A*, vol. 95, p. 042 308, 4 Apr. 2017.

[127] C. Xue, Z.-Y. Chen, Y.-C. Wu, and G.-P. Guo, *Effects of quantum noise on quantum approximate optimization algorithm*, 2019. arXiv: 1909.02196.

[128] J. Marshall, F. Wudarski, S. Hadfield, and T. Hogg, "Characterizing local noise in QAOA circuits," *IOP SciNotes*, vol. 1, no. 2, p. 025 208, Aug. 2020.

[129] E. Fontana, N. Fitzpatrick, D. M. Ramo, R. Duncan, and I. Rungger, "Evaluating the noise resilience of variational quantum algorithms," *Physical Review A*, vol. 104, p. 022 403, 2 Aug. 2021.

[130] J. Kattemölle and G. Burkard, *Effects of correlated errors on the quantum approximate optimization algorithm*, 2022. arXiv: 2207.10622.

[131] G. Quiroz, P. Titum, P. Lotshaw, *et al.*, *Quantifying the impact of precision errors on quantum approximate optimization algorithms*, 2021. arXiv: 2109.04482.

[132] M. Streif, M. Leib, F. Wudarski, E. Rieffel, and Z. Wang, "Quantum algorithms with local particle-number conservation: Noise effects and error correction," *Physical Review A*, vol. 103, no. 4, Apr. 2021.

[133] R. Shaydulin and A. Galda, "Error mitigation for deep quantum optimization circuits by leveraging problem symmetries," in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*, IEEE, Oct. 2021.

[134] T. Gustafsson, *A heuristic approach to column generation for airline crew scheduling* (Lic. dissertation). Chalmers tekniska högskola, 1999.

[135] D. Wedelin, "An algorithm for large scale 0–1 integer programming with application to airline crew scheduling," *Annals of Operations Research*, vol. 57, no. 1, pp. 283–301, 1995.

[136] E. Farhi, D. Gamarnik, and S. Gutmann, *The quantum approximate optimization algorithm needs to see the whole graph: Worst case examples*, 2020. arXiv: 2005.08747.

[137] R. Sreedhar, P. Vikstål, M. Svensson, A. Ask, G. Johansson, and L. García-Álvarez, *The quantum approximate optimization algorithm performance with low entanglement and high circuit depth*, 2022. arXiv: 2207.03404.

[138] M. Dupont, N. Didier, M. J. Hodson, J. E. Moore, and M. J. Reagor, "Calibrating the classical hardness of the quantum approximate optimization algorithm," *PRX Quantum*, vol. 3, no. 4, Dec. 2022.

[139] A. Montanaro, "Quantum speedup of branch-and-bound algorithms," *Physical Review Research*, vol. 2, no. 1, Jan. 2020.

[140] C. Durr and P. Hoyer, *A quantum algorithm for finding the minimum*, 1996. arXiv: quant-ph/9607014.

[141] A. Ambainis and R. Spalek, *Quantum algorithms for matching and network flows*, 2005. arXiv: quant-ph/0508205.

Part II

Appended Papers

PAPER A

A Hybrid Quantum-Classical Heuristic to solve large-scale Integer Linear Programs

Marika Svensson, Martin Andersson, Mattias Grönkvist, Pontus Vikstål, Devdatt Dubhashi, Giulia Ferrini, and Göran Johansson

arXiv:2103.15433 (accepted to Physical Review Applied)

A Hybrid Quantum-Classical Heuristic to solve large-scale Integer Linear Programs

Marika Svensson,[1, 2, *] Martin Andersson,[1] Mattias Grönkvist,[1] Pontus Vikstål,[3] Devdatt Dubhashi,[2] Giulia Ferrini,[3] and Göran Johansson[3]

[1] *Jeppesen, 411 03 Gothenburg, Sweden*
[2] *Department of Computer Science and Engineering, Chalmers University of Technology, 412 96 Gothenburg, Sweden*
[3] *Wallenberg Centre for Quantum Technology, Department of Microtechnology and Nanoscience, Chalmers University of Technology, 412 96 Gothenburg, Sweden*

(Dated: March 29, 2023)

We present a method that integrates any quantum algorithm capable of finding solutions to integer linear programs into the Branch-and-Price algorithm, which is regularly used to solve large-scale integer linear programs with a specific structure. The role of the quantum algorithm is to find integer solutions to subproblems appearing in Branch-and-Price. Obtaining optimal or near-optimal integer solutions to these subproblems can increase the quality of solutions and reduce the depth and branching factor of the Branch-and-Price algorithm and hence reduce the overall running time. We investigate the viability of the approach by considering the Tail Assignment problem and the Quantum Approximate Optimization Algorithm (QAOA). Here, the master problem is the optimization problem Set Partitioning or its decision version Exact Cover and can be expressed as finding the ground state of an Ising spin glass Hamiltonian. For Exact Cover, our numerical results indicate that the required algorithm depth decreases with the number of feasible solutions for a given success probability of finding a feasible solution. For Set Partitioning, on the other hand, we find that for a given success probability of finding the optimal solution, the required algorithm depth can increase with the number of feasible solutions if the Hamiltonian is balanced poorly, which in the worst case is exponential in the problem size. We therefore address the importance of properly balancing the objective and constraint parts of the Hamiltonian. We empirically find that the approach is viable with QAOA if polynomial algorithm depth can be realized on quantum devices.

I. INTRODUCTION

Large-scale Integer Linear Programs (ILPs) appear in the real world frequently as they model problems such as planning, scheduling and resource allocation. These problems are characterized by their large size, a linear cost function, affine inequality and/or equality constraints, as well as variables required to be integers.

Airline planning problems such as Crew Rostering, Crew Pairing [1, 2] and Tail Assignment [3, 4] fall into this category. These problems are made more difficult by very complex rules and regulations imposed by aviation authorities, airlines and unions [1]. These rules can even be hard to express in mathematical optimization models and the models can furthermore have objective functions that are nonlinear in some optimization formulations [5, 6].

One way to address these difficulties is to formulate the optimization problem with a very large number of variables and to separate the problem into a generation problem and a selection problem. With this formulation, standard approaches such as Branch-and-Bound or Branch-and-Cut [7] can not be used directly to solve these problems due to their large size, where even enumerating the legal decision variables can require exponential time and space, see Sec. A 2 for a more detailed explanation. Instead, by starting with an empty set of variables, the generation problem is responsible for generating new variables (aircraft routes in the Tail Assignment problem) to the selection problem (an ILP for the Tail Assignment problem). The task of the selection problem is to find the subset of the generated variables that in the most cost-effective way satisfy all the constraints in the ILP (in the most basic Tail Assignment problem, this corresponds to having each flight in the schedule covered by exactly one aircraft). This process is generalized in the Branch-and-Price algorithm [6] which combines Branch-and-Bound [8] and Column Generation [9, 10] and has generally been successful for large-scale ILPs with this type of structure. The benefit of separating the problem is that the complex rules only affect the generation problem, whereas the selection problem is often a pure Set Cover or Set Partitioning problem.

In the Column Generation algorithm, the generation and selection problems are solved iteratively until optimal conditions hold. In this context, the selection problem is called the Restricted Master Problem (RMP) and the generation problem is called the Pricing Problem (PP). The RMP, which only contains a subset of the decision variables of the original problem, is solved as a Linear Program (LP). Column Generation is generally insufficient to solve the original ILP since the solution is most likely fractional. To remedy this, Column Generation is combined with Branch-and-Bound for finding the integer solution. For readers unfamiliar with Branch-and-Price, details are given in Appx. A.

* e-mail: marika.svensson@jeppesen.com

With the results for factoring with Shor's algorithm [11] and unstructured database search with Grover's algorithm [12], providing subexponential and quadratic speed-up, respectively, it is natural to ask if quantum algorithms also can provide speed-up for ILPs even though superpolynomial speed-up for these problems is not expected. The adiabatic quantum algorithm [13] and quantum annealing [14] have subsequently been proposed. Other quantum algorithms for combinatorial optimization problems [15, 16] such as Grover's adaptive search algorithm [17] have also been proposed. In recent years, much interest has been given to the Quantum Approximate Optimization Algorithm (QAOA) [18] for solving combinatorial optimization problems, as it may be a suitable algorithm to run on near-term gate-based quantum computers and to demonstrate quantum advantage or quantum supremacy [19].

Experiments performed in [20] have reported to demonstrate quantum supremacy for a problem that is not related to optimization. Such devices can be classified as Noisy Intermediate-Scale Quantum (NISQ) computers, where qubits are controlled imperfectly and quantum error correction is generally not considered [21]. Moreover, QAOA was demonstrated in [22] for the Sherrington-Kirkpatrick model and MaxCut, where experiments agree well with simulations. Such results further motivate investigating QAOA for ILPs and distinctly large-scale ILPs.

Here, we address the open question of whether quantum algorithms can provide any advantage for large-scale ILPs, where we stress that these problems can require exponential time and space even to *generate* the full ILP or the continuous relaxation counterpart. The large number of decision variables therefore in practice rules out a direct application of any quantum algorithm capable of solving an ILP, as well as standard classical algorithms for ILPs and the continuous relaxation. Building instead on the above mentioned Branch-and-Price algorithm, we propose to augment it by using a quantum co-processor to find optimal or near-optimal solutions to RMP-instances. For real-world large-scale ILPs, optimality is often intractable and finding good bounds can be hard. The exact definition of a near-optimal or 'good-enough' solution will thus be a balance between running-time and quality of the solution. We can then ensure that the number of decision variables does not exceed the capabilities of the quantum processor when we generate the RMP. Here, we also would like to note that the proposed hybrid algorithm is a heuristic and the performance for real-world ILP problems cannot be properly evaluated until the hardware reaches between 10^3 to 10^4 qubits, corresponding to the same amount of decision variables in the RMP problem. At that point, we argue that this method has the potential to reduce the time to solution and improve solution quality. The detailed description of where a NISQ co-processor could first be used to test for possible speedup in finding near-optimal solutions to ILPs, is one of the main results of this paper. Furthermore, we investigate the method numerically by considering QAOA and the real-world problem Tail Assignment that generalizes Set Partitioning and its decision version Exact Cover, which are NP-hard and NP-complete problems [23]. Here, we are naturally limited to instances with up to 20 decision variables, but we explore these instances to learn how to balance the cost and constraint parts of the Hamiltonian to achieve a large probability of finding the optimal solution, when there are multiple feasible solutions. The results have been obtained by simulating ideal QAOA circuits applied to instances with one or more feasible solutions, extracted from a heuristic Branch-and-Price algorithm [3]. The numerical results expand on [24], where QAOA was applied to instances with a single feasible solution and mapped as an Exact Cover problem (the decision version of the optimization problem Set Partitioning), also extracted from Tail Assignment. The Exact Cover version of this problem has also been investigated experimentally on a quantum processor [25] and on a quantum annealer [26]. The current paper thus gives a broader ILP context and background for these works as well as a first extension to cases with multiple feasible solutions.

The paper is organized as follows. In Sec. II we introduce the Tail Assignment problem. We present the method for integrating a quantum algorithm with Branch-and-Price in Sec. III. In Sec. IV we review QAOA and the chosen mapping of Exact Cover and Set Partitioning to an Ising spin glass Hamiltonian. In Sec. V the extracted RMP instances are presented. We present and motivate the chosen optimization strategy for studying larger algorithm depths in Sec. VI. Results are given in Sec. VII first for Exact Cover and second for Set Partitioning. Last, in Sec. VIII we summarize the findings and discuss interesting open questions that are beyond the scope of this work.

II. TAIL ASSIGNMENT - AN EXAMPLE OF A REAL-WORLD LARGE-SCALE INTEGER LINEAR PROGRAM

Airlines regularly face several large NP-hard planning problems such as Fleet Assignment, Crew Pairing, Crew Rostering and Tail Assignment in the planning process [3, 27]. For Tail Assignment, the task is to determine, given a set of flights and a set of aircraft, what flights are operated by which individual aircraft and what order under the constraint that each flight is flown exactly once such that some objective is optimized. Operational constraints such as minimum connection times, airport curfews, maintenance, and preassigned activities must also be respected, and can be considered part of the input to Tail Assignment. A set of flights operated by an aircraft is referred to as a route, where the operational constraints distinguish legal routes from illegal routes. This means that a solution consists of a set of legal routes that cover all flights exactly once in the most cost-effective way. As

an example, an airline can encounter problems with one thousand flights per day with hundreds of aircraft, where the aircraft are of ten different types [3]. In the worst case, this means that the number of possible routes to determine if they are legal or illegal would be $2^{|F|}$, where F is the set of flights. By considering restrictions such as the arrival time must be less than the departure time of two flights following each other in a route the combinatorial explosion can be decreased. However, typically the number of legal routes will be very large and too large to solve without separating the problem into a selection problem and a generation problem.

Tail assignment can thus be classified as a large-scale ILP, where we refer the readers to [28] and [7] for a comprehensive view of established algorithms for solving ILPs and to [6, 10, 29, 30] for large-scale ILPs. The classical algorithm we consider here used to find optimal or near-optimal solutions to Tail Assignment in [3] is a heuristic Branch-and-Price. The heuristic Branch-and-Price can be understood as the Branch-and-Price algorithm where the branching step is replaced with a fixing step that is better suited for Tail Assignment by diving into a branch of the full search tree.

For consistency, we give the details of the algorithms Branch-and-Bound, Column Generation, Branch-and-Price and the heuristic Branch-and-Price in Appx. A.

A. The Set Partitioning problem and the Exact Cover problem

We define a simple path-based model of Tail Assignment as a Set Partitioning problem

$$\text{minimize} \sum_{r\in R} c_r x_r, \tag{1}$$

$$\text{subject to} \sum_{r\in R} a_{fr} x_r = 1 \ \forall f \in F, \tag{2}$$

$$x_r \in \{0,1\} \ \forall r \in R, \tag{3}$$

where F is the set of flights and R is the set of legal aircraft routes. In the linear objective function, Eq. (1), $c_r \in \mathbb{Z}$ corresponds to the cost of using route r. The entries $a_{fr} \in \{0,1\}$ are elements of a constraint matrix A indicating if flight f is part of route r. A column in the constraint matrix is therefore a route. Furthermore, Eq. (2) enforces the requirement that the set of routes in a solution should contain flight f exactly once. Finally, the decision variables $x_r \ \forall r \in R$ indicate which routes are used.

The Tail Assignment problem can, in practice, also be described by the decision problem Exact Cover, for cases where the objective is to find any feasible solution and not the optimal solution necessarily. The Exact Cover problem can be modeled as an ILP where the objective function in Eq. (1) is ignored and set to 0 for any assignment of the decision variables.

We now define the set S_{feasible} to be the set of feasible solutions to the Set Partitioning problem and the Exact Cover problem as

$$S_{\text{feasible}} = \left\{ \vec{x} \in \{0,1\}^{|R|} : \sum_{r\in R} a_{fr} x_r = 1 \ \forall f \in F \right\}. \tag{4}$$

If we consider a linear system of equations modulus 2

$$A\vec{x} = \vec{b} \bmod 2 \tag{5}$$

where the matrix A is of dimension $|F| \times |R|$, $\vec{x}$ is a column vector with $|R|$ unknown variables and $\vec{b}$ is a column vector with $|F|$ entries. The elements of A, $\vec{b}$ and $\vec{x}$ are either 0 or 1, respectively. The system of equations has

$$2^{|R|-\text{rank}(A)} \tag{6}$$

number of solutions as long as the linear system of equations in Eq. (5) has at least one solution [31]. For Set Partitioning $2^{|R|-\text{rank}(A)}$ constitutes an upper bound on the number of feasible solutions $|S_{\text{feasible}}|$ [32], since any feasible solution to Set Partitioning is also a solution modulus 2 to the system of equations in Eq. (5) where all entries in $\vec{b}$ is set to one. It is therefore possible that the number of feasible solutions is significantly smaller than the upper bound. Furthermore, as the counting version of Exact Cover and Set Partitioning is #P-complete [33], obtaining the actual number of feasible solutions for typical instances for Tail Assignment becomes intractable.

We have investigated the number of feasible solutions for generated RMP instances of Tail Assignment with CPLEX [34]. We find that the number of feasible solutions for two sets of generated instances can be larger than $5 \cdot 10^6$ for problems with 700-800 decision variables. We can therefore not rule out that the number of feasible solutions can be very large in practice, and the consequence is to investigate if a large feasible set is a limiting factor in the performance for QAOA.

III. INTEGRATING A QUANTUM ALGORITHM WITH BRANCH-AND-PRICE

In this section we present the method where the Branch-and-Price algorithm is augmented by integrating any quantum algorithm capable of finding optimal or near-optimal integer solutions to RMP instances. The integrated Branch-and-Price algorithm is depicted in Fig. 1 where Branch-and-Price is distinguished with green and blue colored boxes, and dotted and dashed borders. The green boxes with dotted borders highlight the Column Generation algorithm, and the blue boxes with dashed borders are distinctive for the Branch-and-Bound algorithm. The red boxes with solid borders give the integration of a quantum algorithm. This hybrid algorithm is one of the main results of this paper.

The integrated method utilizes a quantum algorithm for each Column Generation iteration if the RMP is

deemed promising. We remind the reader that since routes are generated dynamically by the Column Generation algorithm each iteration corresponds to a new ILP instance, which means that each iteration provides a possibility to find a new integer solution to the problem via a quantum (or classical) algorithm. For example, we might want to avoid using a quantum algorithm in the beginning of the Column Generation process as it will, in general, be more likely to find good integer solutions in later iterations. However, determining how often to use a quantum algorithm will be a trade-off that depends on if the RMP instance is expected to contain integer solutions with reasonable quality, the run-time of the algorithm for practical instances, the quality of solutions the quantum algorithm can find and its potential to be used in parallel with the Branch-and-Price algorithm. Additionally, prior to utilizing a quantum algorithm classical preprocessing techniques are applied to the RMP instance and the output of the quantum algorithm is used as input to classical postprocessing techniques. We note that the method is similar to those explored in [35] and shares similarities to the use of a quantum device for scheduling problems in [36]. However, our proposed method is the first to our knowledge that considers the hybrid classical and quantum approach for large-scale ILPs and is inspired by the integration of classical IP solvers for 0-1 integer programs into a generation and selection approach for large-scale ILPs in [5].

The addition of a quantum algorithm can improve the classical algorithm in several ways. Firstly, the quantum algorithm can provide a set of optimal or near-optimal integer solutions to RMP instances, which means that the quantum algorithm can be used as a primal heuristic in the Column Generation algorithm. This technique is sometimes referred to as the restricted master heuristic [37]. In the restricted master heuristic, a subset, which is a fixed number of columns and variables, is chosen from the RMP and the resulting problem is solved as a static Integer Program (IP). However, we do not wish to restrict the number of variables and columns to solve as a static IP. Instead, we propose to use the whole RMP instance unless we are required to leave out variables due to limitations in the size of a quantum device. Such heuristics can improve solution quality as observed in [38] by simply obtaining optimal or near-optimal solutions to RMP instances. Furthermore, as primal heuristics have been shown to be very important for solving mixed integer programs, heuristics that leverage a quantum algorithm seems to be a natural step for Branch-and-Price. Moreover, by finding a set of integer solutions, some flexibility is introduced as it is possible to compare the quality of several solutions with respect to more parameters than each solution's cost. This is mainly an advantage for a real-world problem, where buffers occurring in solutions can improve sensitivity to disruptions.

Secondly, the quantum algorithm can provide tighter upper bounds in the branching step, which can be utilized in pruning decisions directly without sacrificing optimality. When we have access to tighter upper bounds, these bounds are compared to the lower bounds found in the Column Generation algorithm. If the lower bound is greater or equal to the upper bound, we can discard the subproblem as we can prune by bound. If we do not have access to these tighter upper bounds, more subproblems are created and explored. This means that the upper bounds can reduce the search tree's size, which leads to a reduced running time of the algorithm. The upper bounds can also reduce the number of iterations required in the Column Generation algorithm as noted in [35] by computing the Lagrangian lower bound, where the stopping criteria is given when the Lagrangian lower bound is greater than the best known upper bound.

We can also consider introducing heuristic pruning rules that can reduce the running time of Branch-and-Price. We remark that finding a good solution fast can be preferable to finding the optimal solution for real-world problems. Heuristic pruning rules guided by optimal or near-optimal solutions to RMP instances can therefore be beneficial. However, as even optimal integer solutions to RMP instances do not guarantee an optimal solution to the subproblem in Branch-and-Price, the pruning decisions will be heuristic and do not guarantee an optimal solution. By introducing heuristic pruning rules, the goal is thus to obtain high quality solutions faster. The pruning decision can be determined by comparing the solution quality for different RMP instances by monitoring the iterative change in the objective and the LP lower bound gap. If the Branch-and-Price is based on variable fixing decisions, the solutions from a quantum algorithm can indicate if certain variables can be chosen to be fixed. The procedure of fixing a variable is such that if a variable x_i is set to 1 for a majority of the obtained solutions, the variable can be fixed to 1 and the Branch-and-Bound algorithm dives into this particular branch of the search tree. Further techniques as in RQAOA in [39] can also be utilized where it is possible to find relations between two decision variables $z_i = \sigma_{ij} z_j$ where $\sigma_{ij} = \text{sign}(\langle \vec{\gamma}^*, \vec{\beta}^* | \hat{\sigma}_i^z \hat{\sigma}_j^z | \vec{\gamma}^*, \vec{\beta}^* \rangle)$ and (i, j) is an edge in the graph $G = (V, E)$ such that $(i, j) = \text{argmax}_{(i',j') \in E} \{ | \langle \vec{\gamma}^*, \vec{\beta}^* | \hat{\sigma}_{i'}^z \hat{\sigma}_{j'}^z | \vec{\gamma}^*, \vec{\beta}^* \rangle | \}$ of an Ising model that encodes an ILP. Such heuristic pruning rules would be similar to the ones of diving heuristics (which can be greedy, random or based on rounding strategies) or local branching heuristics [40].

Finally, the quantum algorithm can reduce the running time if it finds some integer solution below a given threshold or sufficiently close to the lower bound of the original problem as the algorithm, in that case, stops even though the search tree of Branch-and-Price is not explored fully.

Since the method is heuristic, the running time can best be evaluated by executing it on real problems and quantum devices, which is currently intractable due to the current size of quantum computers. We expect that as quantum hardware matures, such experiments will be of interest. We can, on the other hand, note that the gen-

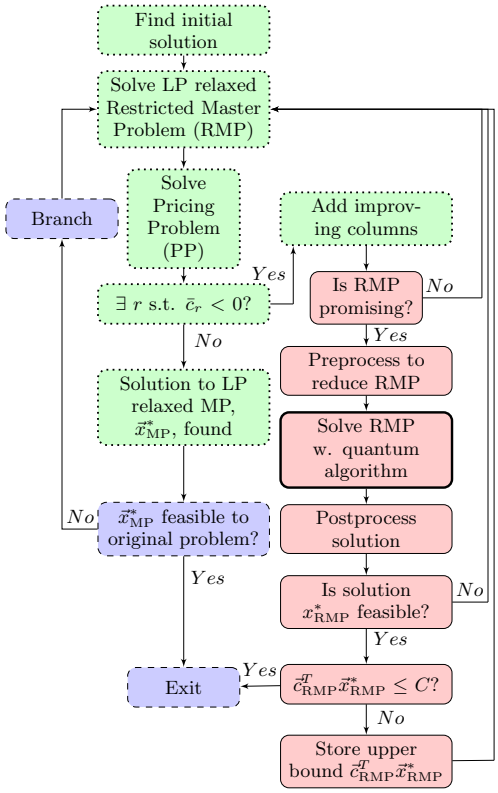


FIG. 1: High level depiction of the Branch-and-Price algorithm integrated with a quantum algorithm capable of finding solutions to ILPs. The variable $\bar{c}_r$ is the reduced cost of route r, $\vec{c}_{RMP}$ is the cost vector of an RMP instance, and its entries correspond to for Tail Assignment to the entries c_r for $r = 1, \ldots, |R|$ in Eq. (1)-(3). The solution provided by a quantum algorithm with postprocessing is the vector $\vec{x}^*_{RMP}$, and the constant C is a threshold for the accepted quality of a solution. This hybrid algorithm is one of the main results of this paper.

eral Branch-and-Bound algorithm has worst-case running time $\mathcal{O}(Mb^d)$ where b is branching factor, d is the search depth and M is the upper bound on the running time to explore a subproblem fully. If we can obtain optimal or near-optimal integer solutions to subproblems, the number of nodes we can prune is larger and thus reduces the algorithm's running time.

Furthermore, as the augmented algorithm is valid for any Branch-and-Price algorithm applied to problems with Master Problems (MPs) possible to solve by some quantum algorithm, the method can be suitable for a large class of ILPs. In particular, this framework can be employed for airline planning problems such as Tail Assignment, Crew Pairing and Crew Rostering but also other large-scale ILPs such as vehicle routing problems [41].

Whilst this approach prohibits applying a quantum algorithm to the Tail Assignment problem and other large-scale ILPs directly, it reduces the number of required decision variables and qubits. In particular, the MP for Tail Assignment, which is a Set Partitioning problem, is mapped such that the number of decision variables corresponds exactly to the required number of qubits (this is true also for other MPs that are 0-1 variable LPs with equality constraints). We could map the Tail Assignment problem directly to an Ising model using an arc-based formulation (see in [3] Eq. (4.1)-(4.7)), but this would require 10^7 qubits for a problem with 10^3 flights and 10 aircraft prior preprocessing. For typical RMP instances we instead expect to require around $10^3 - 10^4$ decision variables for the path-based formulation in Eq. (1)-(3). The proposed method is thus much more suitable for NISQ computers. The arc-based formulation has an additional disadvantage beyond the resource requirement of qubits for problems as Tail Assignment, which are the recursive maintenance requirements. These are non-trivial to map to an Ising model, and removing the constraints would likely result in infeasible solutions.

Moreover, we expect that if RMP instances can be solved approximately with sufficiently shallow circuit depth, the circuits can be realized on NISQ [42] computers.

As mentioned earlier in this section, we propose a preprocessing step using classical integer programming techniques [43, 44] in order to reduce the number of variables and constraints of the problem prior to utilizing a quantum computer. Reducing the number of variables (required qubits) and constraints (problem graph connectivity) is important for the limited NISQ computers to be able to address real-world problems. The level of sophistication can range from very basic to very advanced techniques and the level of sophistication used will be a trade-off between the computational time of the preprocessor and the size and performance of the quantum computer. We also consider classical postprocessing of the output from a quantum algorithm, where additional local searches can be done and we can combine good RMP solutions to obtain improved solutions with standard or specialized classical solvers. Infeasible solutions can additionally be attempted to be corrected to feasible solutions by heuristic classical algorithms.

We stress that the benefit in separating the original problem with the Branch-and-Price algorithm is that the master problem often is a pure Set Partitioning or Set Cover problem without any additional side constraints. The PP, on the other hand, is often a resource constrained shortest path problem that considers the complex rules. Thus, the method is not based on being more suitable for NISQ devices but is based on known successful methods for solving complex large-scale ILPs. Fur-

thermore, by simplifying a real-world problem to a pure Set Cover or Set Partitioning problem we also avoid tackling an ILP with potentially many complicated side constraints with quantum algorithms. This also means that the intricate task of balancing multiple constraint penalties is simplified.

If the method is favorable for large-scale ILPs depends on how complicated the constraints are and the resource requirements of various formulations. The method proposed here can be expected to provide constant speed-up and improve the quality of the solutions. However, it is unclear if the method can provide polynomial speed-up as the addition of a quantum algorithm provides no guarantee for a speed-up and is tied to the Column Generation algorithm, which limits the possible speed-up we can expect. If it is possible to use an alternate formulation that is not required to be separated into a generation problem and a selection problem, it might be beneficial to map the problem directly to an Ising spin glass Hamiltonian. However, as we have pointed out, this often requires significantly more decision variables and qubits to be applicable to real problems and will be more challenging for NISQ devices.

IV. THE QUANTUM APPROXIMATE OPTIMIZATION ALGORITHM

Farhi, Goldstone and Gutmann presented in [18] QAOA, which is a hybrid classical and quantum variational algorithm capable of finding approximate solutions to combinatorial optimization problems. The algorithm is inspired by the adiabatic quantum algorithm but is designed for gate-based quantum computers. Furthermore, evidence that a classical computer can not simulate a QAOA circuit without exponential overhead was presented later in [19]. The algorithm consists of alternating the operators $e^{-i\gamma_k \hat{H}_f}$ and $e^{-i\beta_k \hat{H}_M}$ for $k = 1, 2, \ldots, p$, where p is the depth of the algorithm. An ideal QAOA circuit applied to the initial state $|+\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle$ gives the QAOA state for depth p

$$|\vec{\gamma}, \vec{\beta}\rangle = e^{-i\beta_p \hat{H}_M} e^{-i\gamma_p \hat{H}_f} \ldots e^{-i\beta_1 \hat{H}_M} e^{-i\gamma_1 \hat{H}_f} |+\rangle$$

where $\hat{H}_M = \sum_{i=1}^{n} \hat{\sigma}_i^x$ is the mixing Hamiltonian and $\hat{H}_f = \sum_{\vec{x} \in \{0,1\}^n} f(\vec{x}) |\vec{x}\rangle \langle \vec{x}|$ is a diagonal cost Hamiltonian with respect to the computational basis. The cost Hamiltonian encodes an objective function $f(\vec{x})$ which represents a combinatorial optimization problem. With optimal angles $\vec{\gamma}^*$ and $\vec{\beta}^*$ and sufficiently large algorithm depth, the QAOA state should have a large proportion in states that are close to the ground state and equal to the ground state. By repeating the process of constructing the state and performing measurements in the computational basis, a solution that is equal or close to the ground state of the cost Hamiltonian can be found.

The total running time to execute the quantum circuit, as well as the implementation of the gates associated to the cost Hamiltonian is both graph and hardware architecture dependent. If we let the algorithm depth go to infinity and restrict the angles to be small, the algorithm becomes exact [18].

For an ILP problem $\hat{H}_f$ will consist of one partial Hamiltonian that corresponds to the objective function and another that corresponds to constraints, not unlike common penalty methods [45]. If $f(\vec{x})$ represents a minimization problem the optimal angles $\vec{\gamma}^*$ and $\vec{\beta}^*$ can be found by solving the classical optimization problem

$$\text{argmin } \langle \vec{\gamma}, \vec{\beta} | \hat{H}_f | \vec{\gamma}, \vec{\beta} \rangle , \tag{7}$$

$$\text{subject to } \gamma_i \in [0, 2\pi] \ \forall i = 1, .., p, \tag{8}$$

$$\beta_i \in [0, \pi] \ \forall i = 1, .., p \tag{9}$$

as $\langle \vec{\gamma}, \vec{\beta} | \hat{H}_f | \vec{\gamma}, \vec{\beta} \rangle = f(\vec{x}^*)$ if $|\vec{\gamma}, \vec{\beta}\rangle = |\vec{x}^*\rangle$ where $\vec{x}^*$ is the optimal solution to the problem $f(\vec{x})$ represents. The function in Eq. (7) is the expectation value function and can be referred to as the energy landscape. The domain in Eq. (8) and (9) holds for Hamiltonian $\hat{H}_f$ with integer eigenvalues [24].

As far as we know, instances extracted from the real-world problem Tail Assignment has previously only been studied for QAOA in the context of Exact Cover in [24] where success probabilities close to unity for instances up to 25 qubits with one feasible solution could be obtained for $p \leq 20$ for ideal QAOA circuits. Recently, the vehicle routing problem was also studied up to 20 qubits [46] where a clear dependency was established between the problem size and the performance of QAOA. On the other hand, real-world problems have been studied for quantum annealing, such as for flight gate assignment in [47], where the authors address the issue of bin packing the cost vector of the objective function. However, the complication of degenerate problem instances have not been discussed to a large extent in the context of QAOA, nor has much focus been given to how suitable weights are found to balance the constraints and the objective part of the Hamiltonian $\hat{H}_f$. In Sec. VII, we focus on the effect of choosing suitable weights on the required algorithm depth given a success probability and if having a large feasible space is a limiting factor for the performance.

A. Mapping Set Partitioning and Exact Cover

It is possible to map the Set Partitioning and Exact Cover problem to the Ising spin glass Hamiltonian with an underlying graph $G = (V, E)$ with nodes given by the set V and the edges given by the set E, where the Hamiltonian is $\hat{H} = \sum_{i=1}^{|V|} h_i \hat{\sigma}_i^z + \sum_{(i,j) \in E} J_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z$ as presented in [48]. In this case, the Hamiltonian has at most two spin interaction terms $\hat{\sigma}_i^z \otimes \hat{\sigma}_j^z$ and is 2-local [49], albeit this does not correspond to a geometric locality with respect to hardware architecture.

By introducing a quadratic penalty on the constraints in Eq. (2) a nonlinear integer optimization problem is ob-

tained. The quadratic penalty results in a Hamiltonian which has two parts (when ignoring a constant energy shift), a Hamiltonian which is related to the objective function and a Hamiltonian related to the constraints. These parts are weighted with constants μ_1 and μ_2 accordingly

$$\begin{aligned}\hat{H}^{\text{Set Partitioning}} = &\sum_{r\in R}\Big[\mu_1 \cdot h_r^{\text{Objective}} \\ &+ \mu_2 \cdot h_r^{\text{Exact Cover}}\Big]\hat{\sigma}_r^z \\ &+ \mu_2 \cdot \sum_{r'>r} J_{rr'}^{\text{Exact Cover}}\hat{\sigma}_r^z\hat{\sigma}_{r'}^z,\end{aligned}$$

where

$$h_r^{\text{Objective}} = \frac{c_r}{2},$$

$$h_r^{\text{Exact Cover}} = \sum_{f\in F} a_{fr}\left(\sum_{r'\in R}\frac{a_{fr'}}{2} - 1\right)$$

and

$$J_{rr'}^{\text{Exact Cover}} = \sum_{f\in F}\frac{a_{fr}a_{fr'}}{2}.$$

We observe that the terms $h_r^{\text{Objective}}$ are given by the objective function in Eq. (1) and therefore indicate the cost of an assignment of the decision variables $\vec{x} \in \{0,1\}^{|R|}$. The terms $h_r^{\text{Exact Cover}}$ and $J_{rr'}^{\text{Exact Cover}}$ are due to the constraints in Eq. (2), where $J_{rr'}^{\text{Exact Cover}}$ gives a penalty for each overlapping flight in route r and r' and with the terms $h_r^{\text{Exact Cover}}$ gives a penalty if the combination of routes in an assignment does not cover all flights.

The problem graph $G = (V, E)$ is given by the coefficients $h_r^{\text{Exact Cover}}, h_r^{\text{Objective}}$ and $J_{rr'}^{\text{Exact Cover}}$ in the Hamiltonian where the graph itself can be thought also as a conflict graph of the variables. Finally, the detailed mapping of Exact Cover to an Ising spin glass model was presented in [24] and further expanded for the mapping of the Set Partitioning problem in Appx. B. Mappings for other minimization problems common for large-scale ILPs such as Set Cover can also be found in [48].

V. PROBLEM INSTANCES

The instances [50] have been extracted from the real-world problem Tail Assignment by finding a set of different integer solutions when executing the heuristic Branch-and-Price algorithm. The different solutions are found by permuting the cost of routes randomly during the execution of the algorithm. From this set, 35 instances have been constructed with varying number of routes and number of feasible solutions by combining complete and partial solutions.

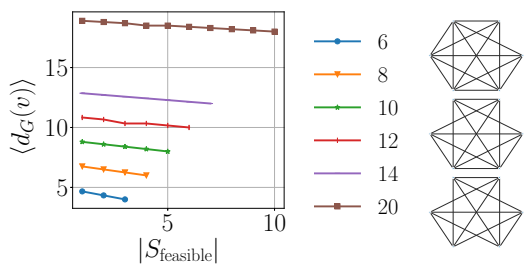
Typically, the instances have very large costs and can be as large as 10^6, making the energy landscape numerically hard to search. The objective function has therefore been further simplified to study qualitative differences in the performance of RMP instances for QAOA. The costs have been simplified such that the smallest cost $c_r^{\min}$ is set to 1, larger costs have been modified such that each cost c_r has a unique value and that the optimal solution is unique. For real instances this is not a proposed methodology, as it can disturb the order of the solutions with respect to quality significantly. An option for real instances is to either increase the weight for the penalty of the constraints, which results in a numerically challenging energy landscape to optimize or we can disturb the costs such that they are easier to handle but preserves the objective function with some accuracy.

We can modify the costs by subtracting all costs with a constant and dividing all costs with another constant, finally the costs are rounded to integers. There is a limit to how much we can disturb the costs such that the order of solutions with respect to cost is not changed significantly. One should choose to divide by a constant that separates the costs c_r by at least a constant integer, which results in a better preservation of the objective function compared to choosing a larger constant to divide the costs by. Here we have assumed a simple objective function to study the performance of QAOA.

The instances are identified by the number of decision variables $|R|$ and the number of feasible solutions $|S_{\text{feasible}}|$. The number of decision variables are 6, 8, 10, 12, 14 and 20. The number of feasible solutions vary from 1 to $|R|/2$. We denote a problem graph associated to an instance $G_{r=|R|}^{s=|S_{\text{feasible}}|}$ which gives the set of graphs as

$$\{G_r^s\}_{r=6-20}^{s=[r/2]}.$$

Additionally, in [24] it was observed that the average node degree of the problem graphs affects the performance of QAOA, in that obtaining near unity success probability require greater algorithm depth as the average node degree, $\langle d_G(v)\rangle$, of the problem graphs increases. The effect of the average node degree was found



(a) Average node degree of problem graphs for generated instances (b) Problem graphs

FIG. 2: Average node degree of the problem graphs are shown in Fig. 2a. Problem graphs $\{G_r^s\}_{r=6}^{s=1,2,3}$ are depicted in Fig. 2b

to dominate over the problem size such that for a given

success probability, the required algorithm depth was greater for instances with 15 qubits compared to instances with 25 qubits. We have extracted the average node degree of each problem graph, depicted in Fig. 2. It can be noted that the average node degree increases with the problem size and decreases as the number of feasible solutions increases. We further noted that the problem graphs are close to being complete graphs, i.e., each node's degree is $|R|-1$ or $|R|-2$. It is thus expected that such instances are hard for QAOA to solve with respect to problem size.

VI. OPTIMIZATION STRATEGY

Finding the solution to the optimization problem in Eq. (7)-(9) is NP-hard [51, 52] in itself. Furthermore, each query of the function in Eq. (7) requires either executing the QAOA circuit on a quantum device or a simulation on a classical computer. As we are currently prohibited from executing QAOA for the problem instances with sufficient algorithm depths on a quantum device the remaining option is to simulate the algorithm with a classical computer. Moreover, since simulating the quantum circuits is exponential in the number of qubits, the consequence is that a function evaluation is computationally expensive. Furthermore, in order to study the performance of QAOA more accurately, we wish to study intermediate to large algorithm depths, which makes the simulations even more expensive as the dimension of the expectation value function in Eq. (7) is 2 times the algorithm depth.

Compared to problems as MaxCut with uniform weights set to 1 or versions thereof [18, 53, 54] the Set Partitioning problem and Exact Cover problem have coefficients in the Hamiltonian h_i and J_{ij} that are governed by the constraint matrix and objective function that grow with the chosen weights. These coefficients are thus not constrained to 0,1 or -1 and can be large. The difference in coefficients results in complicated energy landscapes, that oscillate rapidly, to optimize with multiple local minima. Moreover, we can see this from the closed form expression of the energy landscape for $p=1$ for an Ising spin glass Hamiltonian, associated to a graph $G=(V,E)$ with edge weights J_{ij} and node weights h_i, which is given by

$$\begin{aligned}\langle\gamma\beta|\,\hat{H}\,|\gamma\beta\rangle &= \sum_{i=1}^{n} h_i \sin(2\beta)\sin(2\gamma h_i)\prod_{j:(i,j)\in E}\cos(2\gamma J_{ij}) \\ &+\sum_{(i,j)\in E}\frac{J_{ij}}{2}\Bigg(\sin^2(2\beta)\prod_{\substack{(i,k)\in E\\(j,k)\notin E}}\cos(2\gamma J_{ik})\prod_{\substack{(j,k)\in E\\(i,k)\notin E}}\cos(2\gamma J_{jk}) \\ &\times\Bigg[\cos(2\gamma(h_i-h_j))\prod_{\substack{(i,k)\in E\\(j,k)\in E}}\cos(2\gamma(J_{ik}-J_{jk})) \\ &-\cos(2\gamma(h_i+h_j))\prod_{\substack{(j,k)\in E\\(i,k)\in E}}\cos(2\gamma(J_{ik}+J_{jk}))\Bigg] \\ &+\sin(4\beta)\sin(2\gamma J_{ij})\Bigg[\cos(2\gamma h_i)\prod_{k\neq j:(i,k)\in E}\cos(2\gamma J_{ik}) \\ &+\cos(2\gamma h_j)\prod_{l\neq i:(j,l)\in E}\cos(2\gamma J_{jl})\Bigg]\Bigg), \quad (10)\end{aligned}$$

as presented in [55]. We derive the expression for consistency in Appx. C.

The complicated energy landscape underlies our motivation to focus on obtaining good locally optimal angles via the interpolation strategy presented by Zhou, in [56], in order to study the success probability for QAOA with intermediate to large algorithm depth p. The first step in the interpolation algorithm is to perform global optimization for algorithm depth $k=1$ and for algorithm depth $k>1$ locally optimal angles $(\vec{\gamma}^{L^*},\vec{\beta}^{L^*})$ angles are found by providing a good starting point $(\vec{\gamma}^{L},\vec{\beta}^{L})$ to a local search algorithm. The starting point for local search is determined by interpolating previously found locally optimal angles. The algorithm iterates for $k=2,\ldots,p$. The following definition gives the interpolation in each step

$$\eta^{L}_{k+1,i}=\begin{cases}\eta^{L^*}_{k,1} & \text{if } i=1\\ \frac{i-1}{k}\eta^{L^*}_{k,i-1}+\frac{k-i+1}{k}\eta^{L^*}_{k,i} & \text{if } i=2,\ldots,k\\ \eta^{L^*}_{k,k} & \text{if } i=k+1\end{cases}$$

where η is γ or β. The index i denotes the i:th element of locally optimal angles found for algorithm depth k and index k denotes the best found angles of algorithm depth k. The distinction between L and L^* is the separation of the starting point and angles found after a local search. In our case, the global optimization was performed with python's differential evolution routine. The local optimization was performed with L-BFGS-B, which is also a standard solver in python.

VII. NUMERICAL RESULTS FOR RESTRICTED MASTER PROBLEM INSTANCES

We present the numerical results obtained for ideal QAOA circuits where the variational parameters have been obtained via the interpolation strategy first for Exact Cover in Sec. VII A and second for Set Partitioning in Sec. VII B.

A. Solving the Exact Cover problem

For Exact Cover, we only require to obtain a feasible solution $\vec{x}_i \in S_{\text{feasible}}$. For such purpose, the most natural choice of mapping is by ignoring the objective part of the Hamiltonian, i.e., the cost Hamiltonian is expressed as

$$\hat{H}_f = \hat{H}^{\text{Exact Cover}}.$$

Furthermore, it is straightforward to define the success probability as the probability of obtaining any of the feasible solutions

$$P_{\text{success}}^{\text{Exact Cover}} = \sum_{\vec{x}_i \in S_{\text{feasible}}} |\langle \vec{x}_i | \vec{\gamma}^{L^*}, \vec{\beta}^{L^*} \rangle|^2.$$

The success probabilities for QAOA applied to the Exact Cover instances are plotted in Fig. 3. We remark

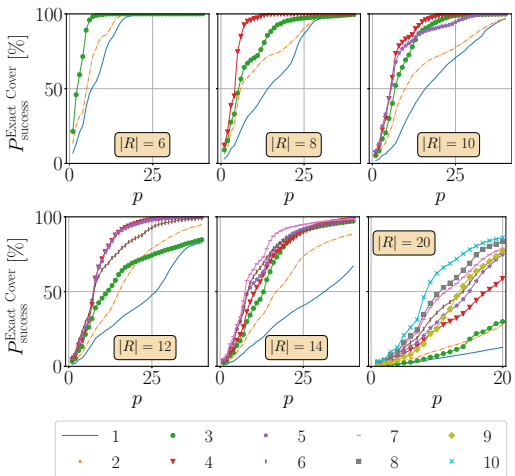


FIG. 3: Success probabilities for Exact Cover. The colors and markers indicate $|S_{\text{feasible}}|$

that the required algorithm depth decreases for a given success probability as the number of feasible solutions increases in general, albeit increases when the problem size increases. These results correspond to results found in [24], wherein Fig. 2 we presented the average node degree of the instances, that decreases with the number of feasible solutions whilst increasing more significantly as the problem size increases.

As the most challenging cases are those where the number of feasible solutions is small, we observe that obtaining a success probability above 50% can require an algorithm depth that is more than $|R|$ by at least a constant, where $|R|$ is the number of decision variables and qubits of the instance. Therefore, it is unknown how well QAOA can perform for instances with $10^3 - 10^4$ decision variables when executed on a NISQ device as decoherence is a limiting factor currently.

B. Solving the Set Partitioning problem

When we consider applying QAOA to the Set Partitioning problem, two additional aspects are of interest. The first aspect is how one should choose good weights that balance the objective part of the Hamiltonian and the Exact Cover (constraints given by a quadratic penalty) part of the Hamiltonian. The second aspect is a consequence of the first, namely how the chosen weights affect the required algorithm depth for a given success probability. The total cost Hamiltonian is a combination of the two partial Hamiltonians accordingly

$$\hat{H}_f = \hat{H}^{\text{Set Partitioning}} = \mu_1 \hat{H}^{\text{Objective}} + \mu_2 \hat{H}^{\text{Exact Cover}}.$$

We have chosen the weight $\mu_1 \in \{\mathbb{Z}^+ \cup \{0\}\}$ depending on a factor f

$$\mu_1 = \begin{cases} 0 & \text{if } f = \infty \\ 1 & \text{otherwise} \end{cases},$$

and $\mu_2 \in \mathbb{Z}^+$ depending on the largest eigenvalues of the partial objective and Exact Cover Hamiltonians, and factor f

$$\mu_2 = \begin{cases} 1 & \text{if } f = \infty \\ \left\lfloor f \cdot \frac{\lambda^{\max}_{\text{Objective}}}{\lambda^{\max}_{\text{Exact Cover}}} \right\rceil & \text{otherwise} \end{cases}.$$

By choosing the weights to be integers, the domain is preserved in the optimization problem defined in Eq. (7)-(9). Thus, $f{=}\infty$ corresponds to the mapping where $\hat{H}^{\text{Set Partitioning}} = \hat{H}^{\text{Exact Cover}}$. We then define the success probability as the probability of finding the optimal solution

$$P_{\text{success}}^{\text{Set Partitioning}} = |\langle \vec{x}^* | \vec{\gamma}^{L^*}, \vec{\beta}^{L^*} \rangle|^2$$

where $\vec{x}^*$ is the solution to the Set Partitioning problem, i.e the binary vector that corresponds the minimal value of Eq. (1) such that $\vec{x}^* \in S_{\text{feasible}}$.

The success probabilities of Set Partitioning are plotted in Fig. 4 for ideal QAOA circuits. Dashed lines distinguish the lines for factor $f = \infty$ and the best found factors f^* are distinguished by the solid lines. Furthermore, success probabilities are tabulated for additional

factors for a given algorithm depth in Appx. D, where the factors have been chosen to construct cost Hamiltonians with the constraint that the ground state corresponds to the optimal solution $\vec{x}^*$.

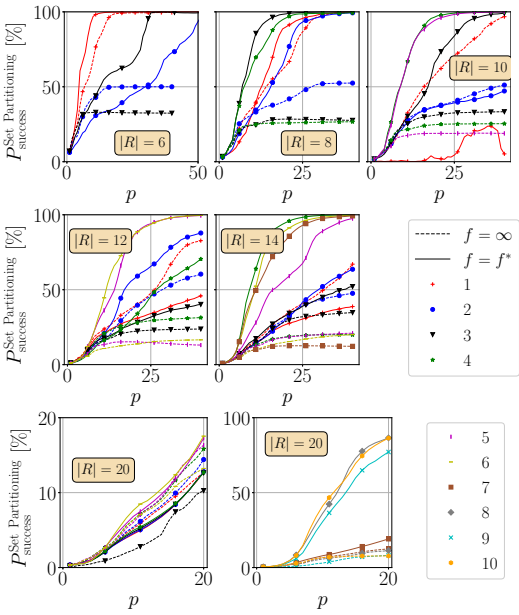


FIG. 4: Success probability for solving the Set Partitioning problem depending on the choice of weights μ_1 and μ_2, the dashed lines correspond to $f = \infty$ and solid lines correspond to the best found factors $f = f^*$. The colors and markers of the lines indicate the value of $|S_{\text{feasible}}|$

It is clear from the results of the numerical simulations in Fig. 4 and Table I that the success probability of solving Set Partitioning can be increased (and thus reducing the required algorithm depth) with a suitable choice of weights μ_1 and μ_2 for 22 instances of the 29 instances with more than one feasible solution. We also observe that a good choice of weights for instances with a single feasible solution corresponds to $f = \infty$ for all problem sizes. We observe that the success probability can decrease with the number of feasible solutions to $P^{\text{Set Partitioning}}_{\text{success}} \approx \frac{1}{|S_{\text{feasible}}|}$ if the weights are chosen poorly, which in the worst case is exponential in the problem size. To avoid requiring a considerable algorithm depth, finding good weights is thus required to solve the optimization problem with NISQ devices.

Moreover, the regret (the difference between the minimum expectation value found during the optimization procedure and the optimal solution) of the expectation value function is depicted in Fig. 5 for instances with 6 routes with varying weights. We observe for $f = \infty$ that the regret is reduced to near zero, whilst failing to increase the success probability significantly above $\frac{1}{|S_{\text{feasible}}|}$. For factors 10 and 100, the regret is greater compared to the best found factor for a given algorithm depth. The difference in regret corresponds to decreased required algorithm depth for the best found factor compared to factors 10 and 100 to achieve near unity success probability for Set Partitioning, see Table I.

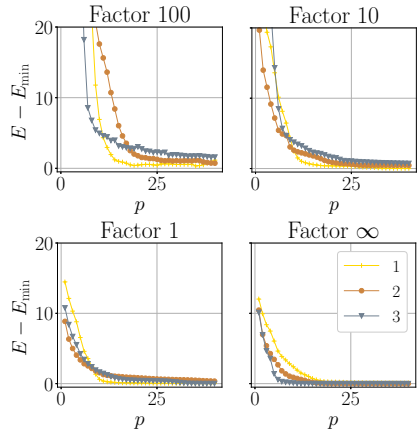


FIG. 5: Regret of the expectation value function for instances corresponding to problem graphs $G_6^{1,2,3}$ for factors 1, 10, 100 and ∞. The regret is defined as the difference between $E = \langle \vec{\gamma}^{L^*}, \vec{\beta}^{L^*} | \hat{H}_f | \vec{\gamma}^{L^*}, \vec{\beta}^{L^*} \rangle$ where $(\vec{\gamma}^{L^*}, \vec{\beta}^{L^*})$ are the locally optimal angles found by the interpolation strategy and $E_{\min} = \langle \vec{x}^* | \hat{H}_f | \vec{x}^* \rangle$

Since we observed that choosing a factor other than ∞ fails to increase the success probability for 7 instances with more than one feasible solution, we have extracted the smallest nonzero energy gap ratio with respect to the maximum eigenvalue. Fig. 6 shows the ratio for instances with 6 and 20 routes. The graphs show for instances with 6 routes that the ratio can be increased for G_6^1 but not for $G_6^{2,3}$ by choosing a factor that considers the cost function. The lack of increased ratio corresponds to the increased required algorithm depth to obtain near unity success probability for $G_6^{2,3}$ compared to G_6^1. Furthermore, the choice $f = 10$ compared to $f = \infty$ results for instances G_{20}^{1-3} in decreased ratios. Whereas the ratio is increased for G_{20}^{4-7} and more distinctly for $G_{20}^{8,9,10}$. We note that as the ratio increases for the choice of factor f, the required algorithm depth is decreased for a given success probability here as well. We conclude from these results that a suitable choice for weights is such that the nonzero energy gap is as large as possible as a ratio of the maximum eigenvalue of the cost Hamiltonian. Moreover, when we attempt to balance the objective and constraint parts of the cost Hamiltonian the smallest eigenvalues are

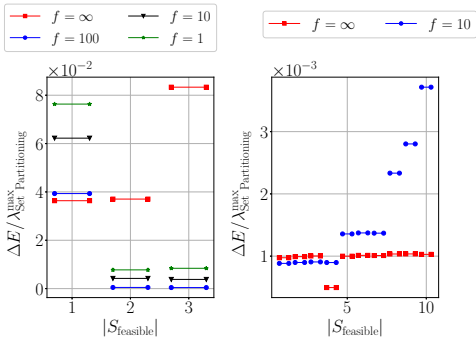


FIG. 6: Minimum nonzero energy gap ΔE for cost Hamiltonian $\hat{H}^{\text{Set Partitioning}}$ depending on the factor f, as a ratio of the maximum eigenvalue

not guaranteed to correspond to feasible solutions. This means that in the pursuit of finding the optimal solution, we can decrease the probability of finding a feasible solution. However, if we find weights such that the smallest eigenvalues correspond to feasible solutions, we do not sacrifice the probability of finding good feasible solutions for finding the optimal solution.

These results indicate that if NISQ devices are limited in algorithm depth, finding suitable weights will be crucial, requiring more computational effort. The task of finding suitable weights for Set Partitioning via the quadratic penalty method typically requires that several subproblems are solved, where each subproblem corresponds to a choice of weights. Typically, with the quadratic penalty method, the weight for the objective part is set to 1 and the weight for the quadratic penalty is set to be small initially. The weight of the quadratic penalty is then increased for a number of iterations or until convergence is reached. The quadratic penalty method could be executed with QAOA. We could also consider solving the problem with a classical computer, where the integer requirement could be relaxed to provide a good guess for the weights. An initial starting point for the weights can also be chosen as $\mu_1 = 1$ and $\mu_2 > \max_{(i,j)\in E}\{|c_i - c_j|\}$ if we assume that the smallest penalty is 1 for exchanging variables x_i and x_j or $\mu_2 > \sum_{r=1}^{|R|} |c_r|$ [57] where μ_2 is bounded from above. An alternative method to obtain suitable weights can be to initially attempt to solve Exact Cover where the weight is zero for the objective part and one for the constraint part. For the second iteration, QAOA with equal penalties set to one for the objective and constraint part of the Hamiltonian is then executed. If the solution degrades to be infeasible compared to the first solution we can assume that the objective part of the Hamiltonian dominates the constraint part. In that case, we need to increase the penalty for the constraint part for a number of iterations or until we reach a convergence. If we, on the other hand, find that we obtain a solution of similar cost as when we attempted to solve Exact Cover, we can increase the penalty for the objective part of the Hamiltonian for a number of iterations until we observe convergence for the solutions or until the solution degrades again such that it is infeasible. Since each choice of weights corresponds to a subproblem to be solved with QAOA it implies a computational overhead. However, if QAOA itself is executed in polynomial time the overhead should not change the overall complexity of the algorithm.

Finally, we conclude that the required algorithm depth of QAOA can be expected to grow with the problem size and increase as the number of feasible solutions decreases (assuming that we have identified suitable weights). Fig. 4 shows that we can expect to require at least $|R|$ in algorithm depth to achieve success probability above 50%.

VIII. CONCLUSIONS

We have proposed a method that can leverage quantum algorithms for large-scale ILPs and investigated the method by considering the quantum algorithm QAOA and the problem Tail Assignment. The method is useful for problems that are typically solved via Column Generation techniques, where a direct application to the problem (typically in a path-based formulation) requires in the worst case exponentially many qubits. The method can also be useful for NISQ devices as our method require less quantum resources compared to the arc-based formulations for problems as vehicle routing and Tail Assignment (defined in [3] as model TAS in Eq. (4.1)-(4.7)). For Crew Pairing and Crew Rostering, in particular, some constraints are not suited to be expressed in mathematical terms as noted in [5], utilizing a quantum algorithm in the Branch-and-Price framework for solving RMP instances can thus be the only viable option. Furthermore, for Tail Assignment, some constraints are recursive and non-trivial to express as an Ising model, limiting the potential to apply a quantum algorithm to the arc-based formulation directly.

The numerical results expand on the results in [24] by considering more diverse and realistic, albeit small instances. The results indicate that the required algorithm depth decreases for a given success probability as the number of feasible solutions increases for Exact Cover, where we find the opposite results for Set Partitioning if the cost Hamiltonian is weighted poorly. Moreover, the reduction in success probability for Set Partitioning can be significant as the number of feasible solutions can be very large. However, we also found that it is possible for most instances to find a suitable choice of weights such that the algorithm depth is significantly reduced to obtain a success probability above 50%, in particular for instances where the number of solutions is larger. Even with suitable weights, we expect that instances can require an algorithm depth that grows with the problem

size and node degree, where harder instances are those with few feasible solutions for QAOA with respect to both Set Partitioning and Exact Cover. Especially hard Set Partitioning instances for QAOA are expected to be those where the minimum nonzero energy gap is small with respect to the largest eigenvalue for any weights we choose and where the minimum eigenvalues no longer correspond to the feasible solutions (whilst the ground state is still the optimal solution). These instances are more difficult because the probability of finding a feasible solution degrades in these cases whilst favoring the optimal solution.

Moreover, we have chosen to follow the mapping for both problems as presented in [48]. Since there exists no evidence that suggests that this particular mapping, although obvious, is optimal there can exist some other more suitable mapping. Since it was observed that the node degree of the graphs affects the required algorithm depth, there might exist some more suitable mapping to be explored where the average node degree of the problem graphs can be reduced. However, exploring alternative mappings for Exact Cover and Set Partitioning has been omitted in this work and left as a potential future challenge to consider.

It can further be observed that common sizes of RMP instances of Tail Assignment require approximately 10^3-10^4 for practical problems. As NISQ computers were suggested to typically have 50-100 qubits initially, we would like to address this discrepancy. We remark that the quantum hardware is improving and new promises of NISQ devises with 1000 qubits by companies as IBM in 2023 [58] implies that the method will become applicable on NISQ devices in the near future. For future work, it would therefore be interesting to run QAOA on such devices for larger instances. Instances of interest to consider are generated RMP instances in Branch-and-Price frameworks for real-world problems and other hard ILP instances publicly available in operational research and mathematical optimization libraries. A remaining challenge for NISQ devices will be to realize QAOA circuits with the desired number of qubits for polynomial algorithm depths.

For future work, it could also be interesting to study if it is possible to reduce the RMP instance to be better suited for NISQ devices. For example, one could attempt to choose a subset of decision variables in RMP instances to construct smaller RMPs. However, such a reduction corresponds to options with a combinatorial behavior. Reducing the size of RMP instances can therefore require more advanced preprocessing techniques. Further techniques as those explored in [59] can also be valuable to consider.

We note that whilst our method provides a possibility to leverage quantum algorithms to an advantage for large-scale ILPs, any quantum algorithm under consideration must be capable of either providing significant speed-up in finding solutions of similar quality as the best classical solvers or capable of finding solutions of improved quality compared to classical solvers during the same execution time. The numerical experiments we have considered in this paper for QAOA can not answer these open questions fully. However, it should be observed that as the average node degree of the generated instances are large, we can therefore consider that the results in Sec. VII A-VII B to correspond to hard instances for QAOA with respect to problem size. Larger instances that are sparse can therefore have a reduced requirement on the algorithm depth, which further motivates studying instances with lower node degrees by both numerical simulations and executions on quantum devices.

Finally, we conclude that it is possible to integrate QAOA with a Branch-and-Price algorithm, where we achieve reasonably high success probabilities for RMP instances with a polynomial algorithm depth. In obtaining high quality integer solutions to RMP instances, the run-time of the general and heuristic Branch-and-Price algorithms can therefore be reduced and improve solution quality.

ACKNOWLEDGMENTS

This work was supported from the Knut and Alice Wallenberg Foundation through the Wallenberg Center for Quantum Technology (WACQT). G. F. acknowledges financial support from the Swedish Research Council through the VR project QUACVA

[1] F. Quesnel, G. Desaulniers, and F. Soumis, A branch-and-price heuristic for the crew pairing problem with language constraints, European Journal of Operational Research **283**, 1040 (2020).

[2] M. Deveci and N. Çetin Demirel, A survey of the literature on airline crew scheduling, Engineering Applications of Artificial Intelligence **74**, 54 (2018).

[3] M. Grönkvist, *The Tail Assignment Problem*, Ph.D. thesis, Chalmers University of Technology and Göteborg University (2005).

[4] M. Grönkvist and J. Kjerrström, *Tail Assignment in Practice*. (Springer Berlin Heidelberg, 2005).

[5] D. Wedelin, An algorithm for large scale 0-1 integer programming with application to airline crew scheduling., Annals of Operations Research **57**, 283 (1995).

[6] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance, Branch-and-price: Column generation for solving huge integer programs, Operations Research **46** (1970).

[7] M. Conforti, G. Cornuéjols, and G. Zambelli, *Integer Programming*, Graduate Texts in Mathematics (Springer In-

ternational Publishing, 2014).
[8] A. H. Land and A. G. Doig, An automatic method for solving discrete programming problems, ECONOMETRICA **28**, 497 (1960).
[9] G. B. Dantzig and P. Wolfe, Decomposition principle for linear programs, Operations Research **8**, 101 (1960), https://doi.org/10.1287/opre.8.1.101.
[10] M. E. Lübbecke and J. Desrosiers, Selected topics in column generation, Oper. Res. **53**, 1007–1023 (2005).
[11] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Journal on Computing **26**, 1484–1509 (1997).
[12] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96 (Association for Computing Machinery, New York, NY, USA, 1996) p. 212–219.
[13] 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 **292**, 472–475 (2001).
[14] T. Kadowaki and H. Nishimori, Quantum annealing in the transverse ising model, Physical Review E **58**, 5355–5363 (1998).
[15] E. Zahedinejad and A. Zaribafiyan, Combinatorial optimization on gate model quantum computers: A survey (2017), arXiv:1708.05294 [quant-ph].
[16] A. Montanaro, Quantum speedup of branch-and-bound algorithms, Physical Review Research **2**, 10.1103/physrevresearch.2.013056 (2020).
[17] A. Gilliam, S. Woerner, and C. Gonciulea, Grover adaptive search for constrained polynomial binary optimization, Quantum **5**, 428 (2021).
[18] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm (2014), arXiv:1411.4028.
[19] E. Farhi and A. W. Harrow, Quantum supremacy through the quantum approximate optimization algorithm (2019), arXiv:1602.07674 [quant-ph].
[20] F. Arute, K. Arya, R. Babbush, D. Bacon, J. Bardin, R. Barends, R. Biswas, S. Boixo, F. Brandao, D. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, and J. Martinis, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).
[21] J. Preskill, Quantum computing in the nisq era and beyond, Quantum **2**, 79 (2018).
[22] M. P. Harrigan, K. J. Sung, M. Neeley, K. J. Satzinger, F. Arute, K. Arya, J. Atalaya, J. C. Bardin, R. Barends, S. Boixo, and et al., Quantum approximate optimization of non-planar graph problems on a planar superconducting processor, Nature Physics **17**, 332–336 (2021).
[23] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness* (W. H. Freeman & Co., USA, 1990).
[24] P. Vikstål, M. Grönkvist, M. Svensson, M. Andersson, G. Johansson, and G. Ferrini, Applying the quantum approximate optimization algorithm to the tail-assignment problem, Physical Review Applied **14**, 10.1103/physrevapplied.14.034009 (2020).
[25] A. Bengtsson, P. Vikstål, C. Warren, M. Svensson, X. Gu, A. Frisk Kockum, P. Krantz, C. Križan, D. Shiri, I.-M. Svensson, G. Tancredi, G. Johansson, P. Delsing, G. Ferrini, and J. Bylander, Improved success probability with greater circuit depth for the quantum approximate optimization algorithm, Phys. Rev. Applied **14**, 034010 (2020).
[26] D. Willsch, M. Willsch, C. D. Gonzalez Calaza, F. Jin, H. De Raedt, M. Svensson, and K. Michielsen, Benchmarking advantage and d-wave 2000q quantum annealers with exact cover problems, Quantum Information Processing **21**, 141 (2022).
[27] P. Belobaba, A. Odoni, and C. Barnhart, *The Global Airline Industry*, Aerospace Series (Wiley, 2009) Chap. 6-7, pp. 153–210.
[28] G. L. Nemhauser and L. A. Wolsey, eds., *Integer and Combinatorial Optimization* (John Wiley & Sons, Inc., 1988).
[29] G. Desaulniers, J. Desrosiers, and M. M. Solomon, *Column Generation. [electronic resource].* (Springer US, 2005).
[30] L. S. Lasdon, *Optimization theory for large systems.* (Dover Publications, 2002).
[31] M. Mezard and A. Montanari, Information, physics, and computation (Oxford University Press, Inc., USA, 2009) Chap. 18, pp. 403–427.
[32] A. V. Seliverstov, *Binary Solutions to Some Systems of Linear Equations.* (Springer International Publishing, 2018).
[33] N. Livne, A note on #p-completeness of np-witnessing relations, Inf. Process. Lett. **109**, 259–261 (2009).
[34] Cplex, IBM ILOG, V12. 1: User's manual for cplex, International Business Machines Corporation **46**, 157 (2009).
[35] E. Danna and C. Le Pape, Branch-and-price heuristics: A case study on the vehicle routing problem with time windows, in *Column Generation*, edited by G. Desaulniers, J. Desrosiers, and M. M. Solomon (Springer US, Boston, MA, 2005) pp. 99–129.
[36] T. Tran, M. Do, E. Rieffel, J. Frank, Z. Wang, B. O'Gorman, D. Venturelli, and J. Beck, A hybrid quantum-classical approach to solving scheduling problems, in *SOCS* (2016).
[37] C. Joncour, S. Michel, R. Sadykov, D. Sverdlov, and F. Vanderbeck, Column generation based primal heuristics, Electronic Notes in Discrete Mathematics **36**, 695 (2010).
[38] J. Núñez Ares, H. de Vries, and D. Huisman, A column generation approach for locating roadside clinics in africa based on effectiveness and equity, European Journal of Operational Research **254**, 1002 (2016).
[39] S. Bravyi, A. Kliesch, R. Koenig, and E. Tang, Obstacles to variational quantum optimization from symmetry protection, Physical Review Letters **125**, 10.1103/physrevlett.125.260505 (2020).
[40] M. Fischetti and A. Lodi, Local branching, Mathematical Programming **98**, 23 (2003).
[41] D. Feillet, A tutorial on column generation and branch-and-price for vehicle routing problems, 4OR **8**, 407 (2010).
[42] F. Leymann and J. Barzen, The bitter truth about gate-based quantum algorithms in the nisq era, Quantum Science and Technology **5**, 044007 (2020).
[43] M.-T. Kong and N. Shah, Preprocessing rules for integer programming solutions to the generalised assignment problem, The Journal of the Operational Research Society **52**, 567 (2001).

[44] T. Achterberg, R. Bixby, Z. Gu, E. Rothberg, and D. Weninger, Presolve reductions in mixed integer programming, INFORMS Journal on Computing **32** (2019).
[45] J. Nocedal and S. Wright, *Numerical Optimization. [electronic resource].*, Springer Series in Operations Research and Financial Engineering (Springer New York, 2006) Chap. 17, pp. 497–528.
[46] Utkarsh, B. K. Behera, and P. K. Panigrahi, Solving vehicle routing problem using quantum approximate optimization algorithm (2020), arXiv:2002.01351 [quant-ph].
[47] T. Stollenwerk, E. Lobe, and M. Jung, Flight gate assignment with a quantum annealer, in *Quantum Technology and Optimization Problems*, edited by S. Feld and C. Linnhoff-Popien (Springer International Publishing, Cham, 2019) pp. 99–110.
[48] A. Lucas, Ising formulations of many np problems, Frontiers in Physics **2**, 5 (2014).
[49] S. Gharibian, Y. Huang, Z. Landau, and S. W. Shin, Quantum hamiltonian complexity, Foundations and Trends® in Theoretical Computer Science **10**, 159–282 (2015).
[50] M. Svensson, Extracted-datainstances-for-tailassignment, https://github.com/marikasvenssonjeppesen/Extracted-datainstances-for-Tailassignment (2021).
[51] E. G. Rieffel, S. Hadfield, T. Hogg, S. Mandrà, J. Marshall, G. Mossi, B. O'Gorman, E. Plamadeala, N. M. Tubman, D. Venturelli, W. Vinci, Z. Wang, M. Wilson, F. Wudarski, and R. Biswas, From ansätze to z-gates: a nasa view of quantum computing (2019), arXiv:1905.02860 [quant-ph].
[52] L. Bittel and M. Kliesch, Training variational quantum algorithms is np-hard – even for logarithmically many qubits and free fermionic systems (2021), arXiv:2101.07267 [quant-ph].
[53] M. Willsch, D. Willsch, F. Jin, H. De Raedt, and K. Michielsen, Benchmarking the quantum approximate optimization algorithm, Quantum Information Processing **19**, 10.1007/s11128-020-02692-8 (2020).
[54] Z. Wang, S. Hadfield, Z. Jiang, and E. G. Rieffel, Quantum approximate optimization algorithm for maxcut: A fermionic view, Physical Review A **97**, 10.1103/physreva.97.022304 (2018).
[55] A. Ozaeta, W. van Dam, and P. L. McMahon, Expectation values from the single-layer quantum approximate optimization algorithm on ising problems (2020), arXiv:2012.03421 [quant-ph].
[56] 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, Physical Review X **10**, 10.1103/physrevx.10.021067 (2020).
[57] S. Harwood, C. Gambella, D. Trenev, A. Simonetto, D. Neira, and D. Greenberg, Formulating and solving routing problems on quantum computers, IEEE Transactions on Quantum Engineering **PP**, 1 (2021).
[58] A. Cho, Ibm promises 1000-qubit quantum computer—a milestone—by 2023, Science 10.1126/science.abe8122 (2020).
[59] A. W. Harrow, Small quantum computers and large classical data sets (2020), arXiv:2004.00026 [quant-ph].
[60] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning, Discrete Optimization **19**, 79 (2016).
[61] C. E. Lemke, The dual method of solving the linear programming problem, Naval Research Logistics Quarterly **1**, 36 (1954).
[62] G. B. Dantzig, The general simplex method for minimizing a linear form under inequality constraints, Pacific Journal of Mathematics **5** (1955).
[63] M. Desrochers and F. Soumis, A generalized permanent labeling algorithm for the shortest path problem with time windows, Information Systems Research - ISR **26** (1988).

Appendix A: The heuristic Branch-and-Price algorithm for solving Tail Assignment

The Branch-and-Price algorithm is designed to solve large-scale Integer Linear Programs (ILPs) and combines the algorithms Column Generation and Branch-and-Bound. In this section, we review first Branch-and-Bound and second the Column Generation algorithm. Last, we review the Branch-and-Price algorithm and the fixing heuristic presented in [3] subject to be integrated with a quantum algorithm.

1. Branch-and-Bound

The Branch-and-Bound algorithm, given in [8] and surveyed in [60] more recently, provides a framework for finding the optimal solution to ILPs. As the feasible region is restricted to integer points and not convex, algorithms applicable for Linear Programs (LPs) can not solve ILPs generally. The distinction here is that LPs can be solved efficiently, whereas ILPs are NP-hard problems.

The algorithm, given in pseudo code in Alg. 1, decomposes the original ILP into subproblems recursively that can be visualized with a tree structure. Exhaustive search is avoided by pruning nodes of the tree giving more acceptable running times in practice. Each node in the tree represents a subproblem which is the original ILP with a reduced feasible space. Each subproblem can be relaxed, i.e., the decision variables are not discrete but continuous, yielding either a lower bound (if a minimization problem), an integer solution, or that the subproblem is infeasible.

Consider here that we are applying Branch-and-Bound to an integer linear program

$$\text{ILP} = \min\left\{\sum_{i=1}^{n} c_i x_i : \vec{x} \in S\right\}$$

where $S = \left\{\vec{x} \in \mathbb{Z}_+^n : \sum_{i=1}^{n} a_{ji} x_i \geq b_j \ \forall j = 1, \ldots, m\right\}$. The Linear Programming (LP) relaxation of the ILP is

$$\text{LP} = \min\left\{\sum_{i=1}^{n} c_i x_i : \vec{x} \in P\right\}$$

where $P = \{\vec{x} \in \mathbb{R}^n_+ : \sum_{i=1}^n a_{ji}x_i \geq b_j \ \forall j = 1, \ldots, m\}$. We know from linear programming theory that the LP relaxation of an ILP gives the relation LP $\leq$ ILP. A partition of the ILPs feasible space S yields two subproblems

$$\text{ILP}_1 = \min\left\{\sum_{i=1}^n c_i x_i : \vec{x} \in S_1\right\},$$
$$\text{ILP}_2 = \min\left\{\sum_{i=1}^n c_i x_i : \vec{x} \in S_2\right\}$$

where S_1 and S_2 are disjoint sets that partition S by a constraint on variable x_j such that $S_1 = \{\vec{x} \in S : x_j \leq \lfloor x_j^0 \rfloor\}$ and $S_2 = \{\vec{x} \in S : x_j \geq \lceil x_j^0 \rceil\}$. The variable $x_j^0 \in \vec{x}^0$ has some fractional value and $\vec{x}^0$ is an optimal solution to LP. We further know from linear programming theory that either ILP_1 or ILP_2 has the optimal solution to ILP. Similarly, the two subproblems can be related to new problems that correspond to the LP relaxation of ILP_1 and ILP_2 which provides lower bounds, can show that there exists no feasible integer point or can find an optimal integer solution. The three problems ILP, ILP_1 and ILP_2 can be visualized as a tree with a parent node and two child nodes, see Fig. 7. Clearly, ILP_1 and ILP_2 can be partitioned further into subproblems giving the tree structure rooted in a node representing the original ILP. If an LP relaxed subproblem is found to be infeasible, the node is pruned, i.e., the branch is not explored further and we say that the node is *pruned by infeasibility*.

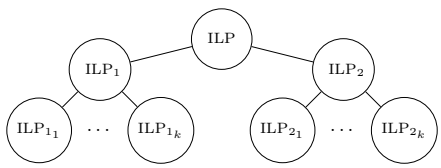


FIG. 7: Conceptual search tree of Branch-and-Bound

When the algorithm recursively explores subproblems, an incumbent, z^*, is maintained which is the current best feasible solution found to the ILP. Whenever a subproblem yields a solution greater or equal to the incumbent, this region cannot contain any integer solutions that would improve upon the one we already have and this particular node is pruned. We say that the node is *pruned by bound*.

If we find that a solution to a subproblem is integral, we also prune this node as we have found an optimal partial solution or candidate incumbent z_i for this specific region. We say that the node is *pruned by integrality*. If $z_i < z^*$ the incumbent is updated.

Finally, if a subproblem can not be pruned by infeasibility, bound or integrality the subproblem is partitioned into $k \geq 2$ nodes representing k subproblems, which are children to the current subproblem we are exploring in the tree. The k subproblems are then added to a list of unexplored subproblems and a new subproblem is chosen to be explored. When there are no unexplored subproblems left the algorithm terminates and returns the incumbent solution and the corresponding assignment.

Algorithm 1 Branch-and-Bound(S)

$\vec{x}^* \leftarrow \emptyset$
$z^* \leftarrow \infty$
$z_U^* \leftarrow \infty$
$\mathcal{L} \leftarrow \{S\}$
while $|\mathcal{L}| > 0$ **do**
  $S_i \leftarrow chooseSubProblem(\mathcal{L})$
  $\mathcal{L} \leftarrow \mathcal{L} \backslash \{S_i\}$
  if S_i has feasible solution to LP relaxation **then**
    $(z_L, \vec{x}_L) \leftarrow solveLPrelaxation(S_i)$
    if $z_L < z_U^*$ **then**
      if $\vec{x}_L$ feasible to ILP **then**
        ▷ Prune by integrality
        if $z_L < z^*$ **then**
          $\vec{x}^* \leftarrow \vec{x}_L$
          $z^* \leftarrow z_L$
          $z_U^* \leftarrow z^*$
        end if
      else
        $(z_U, \vec{x}_U) \leftarrow getFeasibleSolution(S_i)$
        $z_U^* \leftarrow \min(z_U^*, z_U)$
        $\{S_{i_1}, \ldots, S_{i_k}\} \leftarrow partition(S_i)$
        $\mathcal{L} \leftarrow \mathcal{L} \cup \{S_{i_1}, \ldots, S_{i_k}\}$
      end if
    else
      ▷ Prune by bound
    end if
  else
    ▷ Prune by infeasibility
  end if
end while
return $(z^*, \vec{x}^*)$

2. Column Generation

In the previous section we mentioned that the LP relaxation of an ILP could be efficiently solved. However, consider the case where the number of variables is exponentially large so that even generating the LP would take exponential time and space. This is exactly the case for large-scale ILPs as the Tail Assignment formulation in [3], which has an exponential number of possible routes in the worst case.

The Column Generation algorithm [10], depicted with green colored boxes with dotted borders in Fig. 1 and presented in pseudo code in Alg. 2, is based on well known duality concepts from linear programming theory. It has been proved successful for both linear programs and ILPs, particularly when the number of decision variables is very large. Instead of attempting to construct and solve the complete problem it is decomposed into a

Master Problem (MP)

$$z^*_{\text{MP}} = \text{minimize} \sum_{j \in J} c_j x_j,$$
$$\text{subject to} \sum_{j \in J} a_{ij} x_j \geq b_i \ \forall i \in I$$
$$x_j \geq 0 \ \forall j \in J$$

and a Pricing Problem (PP)

$$\text{argmin} \left\{ \bar{c}_j = c_j - \sum_{i \in I} a_{ij} \pi_i : j \in J \right\},$$

here π_i are the dual variables that correspond to the primal variables, x_j, found by solving the MP. The PP often encapsulates most of the problem specific details and difficult constraints and generates new columns, also referred to as entering variables.

Since the number of decision variables is very large, the MP is further reduced to a restricted version, denoted the Restricted Master Problem (RMP), meaning that the number of decision variables is smaller, often much smaller, than the original problem. The reduced size of the RMP is tractable to solve with some LP solver such as the dual simplex [61] or primal simplex [62] algorithm, compared to the MP.

The decomposition results in an iterative algorithm where the RMP and the PP are solved for a number of iterations or until optimal conditions hold. For each iteration, we attempt to find entering and exiting variables where the exiting variables are removed from the RMP and the entering variables are added to the RMP, resulting in new RMP and PP instances.

The PP is thus some problem that when solved can generate improving columns and decision variables to the RMP, based on given input of the dual variables from the RMP, such that the cost of the new solution, which at this point is not guaranteed to be integral, is improved. Improving columns are identified by having a negative reduced cost $\bar{c}_j$ and optimal conditions hold when no variables with negative reduced cost can be found, which is the same condition as in the simplex algorithm.

If the original problem is an ILP, the MP is the LP relaxation of the ILP. In the case of Tail Assignment the RMP corresponds to a restricted and LP relaxed Set Partitioning or Exact Cover problem, see Sec. II A, where the decision variables are continuous real variables. The PP can thus be defined as

$$\text{argmin} \left\{ \bar{c}_r = c_r - \sum_{f \in F} a_{fr} \pi_f : r \in R \right\}. \tag{A1}$$

for Tail Assignment, where π_f is the dual variable of flight f obtained when solving the RMP.

To be noted, the first step of Column Generation is to construct an initial RMP, which for Tail Assignment can be $A = \mathbb{1}_{|F| \times |F|}$ where the costs c_r are set to some large number and thus unlikely to be part of a solution. Variables can be chosen as exiting variables when the value of the reduced cost is above a given threshold, however, removing variables from the RMP does not necessarily as improve convergence as removing variables also removes dual information. Further investigations in deleting columns can be found in [3], in Sec. 6.4.

Furthermore, solving the PP at first glance appears intractable as the number of reduced costs can be exponentially large. By formulating the problem as a Resource Constrained Shortest Path Problem (RCSPP) we avoid to explicitly construct all routes. The RCSPP is described by a connection network, depicted in Fig. 8 with a unique sink vertex and other vertices representing flights with edges that represent legal connections where the nodes are associated with a flight cost c_f and a dual variable π_f found by solving the RMP. The problem depicted in Fig. 8 is a shortest path problem, where the objective is given by Eq. (A1) and additional costs for each edge, i.e., flight connection. The problem becomes an RCSPP problem when we introduce cumulative constraints as resources, where a resource is a value accumulated throughout the route and is required to not go above some limit, hence giving a resource constraint. Resources and subsequently resource constraints are introduced for each maintenance type where a requirement can be given by the maximum flying hours that are allowed prior to a check or the maximum number of landings an aircraft is allowed to make prior to a check.

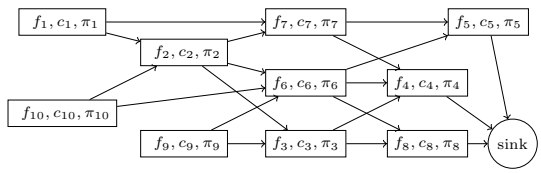


FIG. 8: Pricing problem

The task is then to find the shortest path with respect to the reduced costs in the network and the resource constraints. The PP is NP-hard where for example a label-setting algorithm [63] can be applied to solve instances.

Algorithm 2 ColumnGeneration (F, T)

```
RMP ← FindInitialSolution(F,T)
π⃗, x⃗ ← SolveRestrictedMasterProblem(RMP)
negativeReducedCosts ← solvePricingProblem(π⃗)
exitingVariables ← findExitingVariables(RMP)
while negativeReducedCosts ≠ ∅ do
    RMP ← RMP ∪ {negativeReducedCosts}
    RMP ← RMP\{exitingVariables}
    π⃗, x⃗ ← SolveRestrictedMasterProblem(RMP)
    negativeReducedCosts ← solvePricingProblem(π⃗)
    exitingVariables ← findExitingVariables(RMP)
end while
return z = c⃗^T x⃗, x⃗
```

3. Branch-and-Price and fixing heuristics

Since only integral solutions are accepted for ILPs (and the original formulation of Tail Assignment), the Column Generation algorithm is typically augmented to Branch-and-Price [6], by combining Column Generation and Branch-and-Bound. In Branch-and-Price, we add an additional branching step, which occurs when no columns with a negative reduced cost can be found via solving the PP and the optimal solution is not integral. The fractional solution from the Column Generation provides a lower bound, if we are considering a minimization problem, as the algorithm solves the LP relaxed subproblem in Branch-and-Bound. In the branching step the search space is partitioned, where the Column Generation algorithm is executed for each subproblem created. Therefore, we point out that Branch-and-Price can be thought of as Branch-and-Bound where Column Generation is utilized as a subroutine to compute bounds, show infeasibility or find an integer solution.

Moreover, Grönkvist [3] noticed that Branch-and-Price might be unnecessarily slow when applied to Tail Assignment and introduced a fixing heuristic where the branching step is replaced. The fixing heuristic finds the variable x_i closest to 1 and fixes it to 1, which forces the corresponding route to be part of the solution. It can be noted that the difference between the fixing heuristic and the typical branching is that the search space is restricted and not partitioned, meaning that the fixing heuristic is a dive into a specific branch of the search tree. Additional backtracking methods are utilized but are beyond this section's scope where such further information can be found in [3]. We denote the modified Branch-and-Price algorithm as the heuristic Branch-and-Price and depict the algorithm with the blue and green colored boxes with dotted and dashed borders in Fig. 1 subject to be integrated with a quantum algorithm in Sec. III.

Appendix B: Mapping problems to the Ising spin glass model

If we consider the Set Partitioning problem in Eq. (1)-(3) and apply a quadratic penalty on the constraints we obtain a nonlinear integer optimization problem. If we further assume constants $\mu_1 \in \{\mathbb{Z}^+ \cup \{0\}\}$, $\mu_2 \in \mathbb{Z}^+$ that balance the objective function and the constraints we obtain a new optimization problem

$$\text{min. } \mu_1 \sum_{r\in R} c_r x_r + \mu_2 \sum_{f\in F} \left(\left[\sum_{r\in R} a_{fr} x_r \right] - 1 \right)^2, \quad \text{(B1)}$$

$$\text{s.t.} \qquad x_r \in \{0,1\} \ \forall r \in R. \quad \text{(B2)}$$

The new optimization problem in Eq. (B1)-(B2) can subsequently be modified to have variables $s_r \in \{-1,1\}$ by replacing the variables $x_r = \frac{1+s_r}{2}$, as presented by Lucas for several combinatorial optimization problems [48]. The variable change results in the following classical Hamiltonian

$$\begin{aligned} H(s_1,\dots,s_{|R|}) &= \mu_1 \cdot \sum_{r\in R} c_r \frac{1+s_r}{2} \\ &\quad + \mu_2 \cdot \sum_{f\in F} \left(\left[\sum_{r\in R} a_{fr} \frac{1+s_r}{2} \right] - 1 \right)^2 \\ &= \mu_1 H^{\text{Objective}}(s_1,\dots,s_{|R|}) \\ &\quad + \mu_2 H^{\text{Exact Cover}}(s_1,\dots,s_{|R|}) \end{aligned}$$

which we expand separately for the objective Hamiltonian and the Exact Cover Hamiltonian, where the Exact Cover Hamiltonian can be referred to as the constraint Hamiltonian. For the objective part we obtain

$$\begin{aligned} H^{\text{Objective}}(s_1,\dots,s_{|R|}) &= \\ &\sum_{r\in R} h_r^{\text{Objective}} s_r + \sum_{r'>r} J_{rr'}^{\text{Objective}} s_r s_{r'} \\ &= \sum_{r\in R} \frac{c_r}{2} s_r + \sum_{r\in R} \frac{c_r}{2} = \sum_{r\in R} \frac{c_r}{2} s_r, \end{aligned}$$

by ignoring the constant energy shift. Thus

$$\begin{aligned} h_r^{\text{Objective}} &= \frac{c_r}{2}, \\ J_{rr'}^{\text{Objective}} &= 0. \end{aligned}$$

For the constraints, i.e., the Exact Cover Hamiltonian, it was showed in [24] that the classical Hamiltonian takes the form

$$\begin{aligned} H^{\text{Exact Cover}}(s_1,\dots,s_{|R|}) &= \sum_{r\in R} h_r^{\text{Exact Cover}} s_r \\ &\quad + \sum_{r'>r} J_{rr'}^{\text{Exact Cover}} s_r s_{r'} \end{aligned}$$

where

$$\begin{aligned} h_r^{\text{Exact Cover}} &= \sum_{f\in F} a_{fr} \left(\sum_{r'\in R} \frac{a_{fr'}}{2} - 1 \right), \\ J_{rr'}^{\text{Exact Cover}} &= \sum_{f\in F} \frac{a_{fr} a_{fr'}}{2}. \end{aligned}$$

For the Set Partitioning problem we then obtain the following Hamiltonian

$$\begin{aligned} H^{\text{Set Partitioning}}(s_1,\dots,s_{|R|}) &= \\ &\sum_{r\in R} \left[\mu_1 \cdot h_r^{\text{Objective}} + \mu_2 \cdot h_r^{\text{Exact Cover}} \right] s_r + \\ &\mu_2 \cdot \sum_{r'>r} J_{rr'}^{\text{Exact Cover}} s_r s_{r'}. \end{aligned}$$

Finally, the quantum Hamiltonian is obtained by promoting s_r to $\hat{\sigma}_r^z$

$$\hat{H}^{\text{Set Partitioning}}(\hat{\sigma}_1^z,\ldots,\hat{\sigma}_{|R|}^z) = \sum_{r\in R}[\mu_1\cdot h_r^{\text{Objective}} + \mu_2\cdot h_r^{\text{Exact Cover}}]\hat{\sigma}_r^z + \mu_2\cdot\sum_{r'>r} J_{rr'}^{\text{Exact Cover}}\hat{\sigma}_r^z\hat{\sigma}_{r'}^z.$$

It can be noted that the mapping holds for any ILP of the form

$$\begin{aligned}\text{minimize}\quad & \sum_{r\in R} c_r x_r,\\ \text{subject to}\quad & \sum_{r\in R} a_{fr}x_r = b_f\ \forall f\in F,\\ & x_r\in\{0,1\}\ \forall r\in R,\end{aligned}$$

if $h_r^{\text{Exact Cover}}$ is modified to

$$h_r^{\text{Exact Cover}} = \sum_{f\in F} a_{fr}\left(\sum_{r'\in R}\frac{a_{fr'}}{2} - b_f\right).$$

Appendix C: Expectation value for algorithm depth one

In this section we derive the expression of the expectation value in Eq. (10) for algorithm depth $p=1$ of QAOA. The expectation value of a general Ising spin glass Hamiltonian $\hat{H}=\sum_{i=1}^n h_i\hat{\sigma}_i^z+\sum_{(i,j)\in E}J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z$ associated to an undirected graph $G=(V,E)$ with $n=|V|$ nodes and $|E|$ edges can be computed accordingly

$$\langle E\rangle = \text{Tr}[\rho\hat{H}] = \sum_{i=1}^n h_i\text{Tr}[\rho\hat{\sigma}_i^z] + \sum_{(i,j)\in E}J_{ij}\text{Tr}[\rho\hat{\sigma}_i^z\hat{\sigma}_j^z].$$

The undirected graph G has no self loops, which means that no edge (i,i) is present in the graph. We furthermore consider the edge (i,j) as identical to edge (j,i) and the sum over edges thus include the edge between node i and j exactly once. In other words, the edges are unordered pairs that connect the two nodes without a particular direction, hence in graph G that we consider (j,i) is simply another way of referring to edge (i,j) which means that $J_{ij}=J_{ji}$. The density matrix in the expression for the expectation value is $\rho = U_M(\beta)U_c(\gamma)\left|+\right\rangle\left\langle+\right|U_c^\dagger(\gamma)U_M^\dagger(\beta)$, where QAOA operators are defined as

$$U_M(\beta) = \prod_{i=1}^n e^{-i\beta\hat{\sigma}_i^x}$$

$$U_c(\gamma) = \prod_{i=1}^n e^{-i\gamma h_i\hat{\sigma}_i^z}\prod_{(i,j)\in E}e^{-i\gamma J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z} = U_c^1(\gamma)U_c^2(\gamma).$$

We can rewrite the expectation value as

$$\begin{aligned}\langle E\rangle &= \sum_{i=1}^n h_i\text{Tr}\left[\left|+\right\rangle\left\langle+\right|U_c^\dagger(\gamma)U_M^\dagger(\beta)\hat{\sigma}_i^z U_M(\beta)U_c(\gamma)\right]\\ &+\sum_{(i,j)\in E}J_{ij}\text{Tr}\left[\left|+\right\rangle\left\langle+\right|U_c^\dagger(\gamma)U_M^\dagger(\beta)\hat{\sigma}_i^z\hat{\sigma}_j^z U_M(\beta)U_c(\gamma)\right]\\ &= \sum_{i=1}^n h_i\langle E_i\rangle + \sum_{(i,j)\in E}J_{ij}\langle E_{ij}\rangle\end{aligned}$$

by the cyclic property of the trace. We remark that partial terms $\text{Tr}[\left|+\right\rangle\left\langle+\right|\hat{a}]$ of the expectation value contribute if $\hat{a}$ is a combination of $\hat{\sigma}^x$ and/or $\mathbb{1}$. The resulting value for terms $\langle E_i\rangle$ and $\langle E_{ij}\rangle$ have been derived for triangle free graphs in [53]. However, the resulting value for a graph with triangles was shown via Mathematica in [55]. In this section we show the same general form of $\langle E_{ij}\rangle$ by analytical means.

We begin by considering some edge (i,j), clearly all terms in $U_M(\beta)$ commute with $\hat{\sigma}_i^z\hat{\sigma}_j^z$ except for $e^{-i\beta\hat{\sigma}_i^x}$ and $e^{-i\beta\hat{\sigma}_j^x}$. We use the following relation

$$\begin{aligned}F(\hat{a},\eta\hat{b}) &= e^{i\eta\hat{b}}\hat{a}e^{-i\eta\hat{b}}\\ &= c_\eta^2\hat{a} + s_\eta^2\hat{b}\hat{a}\hat{b} + i\frac{s_{2\eta}}{2}[\hat{b},\hat{a}]\end{aligned}\tag{C1}$$

where $c_x^y=\cos^y(x)$ and $s_x^y=\sin^y(x)$ for convenience. The terms resulting from the mixing operator $U_M(\gamma)$ are therefore

$$\begin{aligned}U_M^\dagger(\beta)\hat{\sigma}_i^z\hat{\sigma}_j^z U_M(\beta) &= F(\hat{\sigma}_i^z,\beta\hat{\sigma}_i^x)F(\hat{\sigma}_j^z,\beta\hat{\sigma}_j^x)\\ &= c_{2\beta}^2\hat{\sigma}_i^z\hat{\sigma}_j^z\\ &+ c_{2\beta}s_{2\beta}[\hat{\sigma}_i^z\hat{\sigma}_j^y + \hat{\sigma}_i^y\hat{\sigma}_j^z]\\ &+ s_{2\beta}^2\hat{\sigma}_i^y\hat{\sigma}_j^y\end{aligned}$$

by evaluating Eq. (C1). For the ease of future derivitions, we separate these parts as $\langle E_{ij}\rangle = \langle E_{ij}^{zz}\rangle + \langle E_{ij}^{zy}\rangle + \langle E_{ij}^{yz}\rangle + \langle E_{ij}^{yy}\rangle$ where the sinus and cosinus terms are temporarily ignored. To clarify, here we defined $\langle E_{ij}^{ab}\rangle = \text{Tr}\left[\left|+\right\rangle\left\langle+\right|U_c^\dagger(\gamma)\hat{\sigma}_i^a\hat{\sigma}_j^b U_c(\gamma)\right]$.

We note that $[U_c(\gamma),\hat{\sigma}_i^z\hat{\sigma}_j^z]=0$ and hence $\langle E_{ij}^{zz}\rangle$ does not contribute to the overall expectation value $\langle E\rangle$. However, for $\hat{\sigma}_i^y\hat{\sigma}_j^z$ all terms in $U_c^2(\gamma)$ with operators corresponding to edges with node i contribute, i.e., $[e^{-i\gamma J_{kp}\hat{\sigma}_k^z\hat{\sigma}_p^z},\hat{\sigma}_i^y\hat{\sigma}_j^z]\neq 0\ \forall p:(k=i,p)\in E$ and only $[e^{-i\gamma h_i\hat{\sigma}_i^z},\hat{\sigma}_i^y\hat{\sigma}_j^z]\neq 0$ of all operator terms corresponding to nodes in $U_c^1(\gamma)$. Similarly for $\hat{\sigma}_i^z\hat{\sigma}_j^y$, all operator terms in $U_c^2(\gamma)$ corresponding to edges that include node j and the term $e^{-i\gamma h_j\hat{\sigma}_j^z}$ in $U_c^1(\gamma)$ contribute. For $\hat{\sigma}_i^y\hat{\sigma}_j^y$ we note that all operator terms in $U_c^2(\gamma)$ for edges that include node i or j contributes as $[e^{-i\gamma J_{kp}\hat{\sigma}_k^z\hat{\sigma}_p^z},\hat{\sigma}_i^y\hat{\sigma}_j^y]\neq 0$ $\forall p\neq i:(k=j,p)\in E$ and $\forall p\neq j:(k=i,p)\in E$. Furthermore, both terms in $U_c^1(\gamma)$ that correspond to node i and j contribute to the expectation value as well.

We now wish to evaluate the terms $\langle E_{ij}^{yy}\rangle, \langle E_{ij}^{zy}\rangle$ and $\langle E_{ij}^{yz}\rangle$. We begin with the most complex case, $\langle E_{ij}^{yy}\rangle$,

which is the only term that changes if triangles are present in the graph compared to the expression given in [53]. Since the only terms in $U_c^1(\gamma)$ that do not commute with $\hat{\sigma}_i^y\hat{\sigma}_j^y$ are $e^{-i\gamma h_i\hat{\sigma}_i^z}$ and $e^{-i\gamma h_j\hat{\sigma}_j^z}$, the terms that we obtain from $U_c^1(\gamma)$ are thus

$$\begin{aligned}U_c^{1\dagger}(\gamma)\hat{\sigma}_i^y\hat{\sigma}_j^yU_c^1(\gamma) &= F(\hat{\sigma}_i^y,\gamma h_i\hat{\sigma}_i^z)F(\hat{\sigma}_j^y,\gamma h_j\hat{\sigma}_j^z)\\ &= c_{2h_i\gamma}c_{2h_j\gamma}\hat{\sigma}_i^y\hat{\sigma}_j^y + s_{2h_i\gamma}c_{2h_j\gamma}\hat{\sigma}_i^x\hat{\sigma}_j^y\\ &+ c_{2h_i\gamma}s_{2h_j\gamma}\hat{\sigma}_i^y\hat{\sigma}_j^x + s_{2h_i\gamma}s_{2h_j\gamma}\hat{\sigma}_i^x\hat{\sigma}_j^x\end{aligned}$$

which gives us four terms to consider. We will now use the following relation

$$\begin{aligned}G(\hat{c},\eta_a\hat{a},\eta_b\hat{b}) &= e^{i\eta_a\hat{a}}e^{i\eta_b\hat{b}}\hat{c}e^{-i\eta_b\hat{b}}e^{-i\eta_a\hat{a}}\\ &= e^{i\eta_a\hat{a}}(c_{\eta_b}^2\hat{c} + s_{\eta_b}^2\hat{b}\hat{c}\hat{b} + i\frac{s_{2\eta_b}}{2}[\hat{b},\hat{c}])e^{-i\eta_a\hat{a}}\\ &= c_{\eta_b}^2[c_{\eta_a}^2\hat{c} + s_{\eta_a}^2\hat{a}\hat{c}\hat{a} + i\frac{s_{2\eta_a}}{2}[\hat{a},\hat{c}]]\\ &+ s_{\eta_b}^2[c_{\eta_a}^2\hat{b}\hat{c}\hat{b} + s_{\eta_a}^2\hat{a}(\hat{b}\hat{c}\hat{b})\hat{a} + i\frac{s_{2\eta_a}}{2}[\hat{a},\hat{b}\hat{c}\hat{b}]]\\ &+i\frac{s_{2\eta_b}}{2}[c_{\eta_a}^2[\hat{b},\hat{c}] + s_{\eta_a}^2\hat{a}[\hat{b},\hat{c}]\hat{a} + i\frac{s_{2\eta_a}}{2}[\hat{a},[\hat{b},\hat{c}]]]\end{aligned} \tag{C2}$$

to evaluate the contributing terms when $U_c^2(\gamma)$ is applied. It then becomes clear that the operators $\hat{\sigma}_i^x\hat{\sigma}_j^y$ and $\hat{\sigma}_i^y\hat{\sigma}_j^x$ do not contribute to $\langle E_{ij}^{yy}\rangle$ as there are no terms that can result in a pure $\hat{\sigma}^x$ and/or $\mathbb{1}$ combination when evaluating Eq. (C2). We explicitly give the expressions for all terms below for the case of $\hat{\sigma}_i^x\hat{\sigma}_j^y$ (which by symmetry also allow us to throw away $\hat{\sigma}_i^y\hat{\sigma}_j^x$):

$$\hat{a} = \hat{\sigma}_i^z\hat{\sigma}_k^z, \quad \hat{b} = \hat{\sigma}_j^z\hat{\sigma}_p^z, \quad \hat{c} = \hat{\sigma}_i^x\hat{\sigma}_j^y,$$

$$\hat{a}\hat{c}\hat{a} \propto \hat{\sigma}_i^x\hat{\sigma}_j^y, \quad [\hat{a},\hat{c}] \propto \hat{\sigma}_i^y\hat{\sigma}_j^y\hat{\sigma}_k^z, \quad \hat{b}\hat{c}\hat{b} \propto \hat{\sigma}_i^x\hat{\sigma}_j^y,$$

$$\hat{a}(\hat{b}\hat{c}\hat{b})\hat{a} \propto \hat{\sigma}_i^x\hat{\sigma}_j^y, \quad [\hat{a},\hat{b}\hat{c}\hat{b}] \propto \hat{\sigma}_i^y\hat{\sigma}_j^y\hat{\sigma}_k^z, \quad [\hat{b},\hat{c}] \propto \hat{\sigma}_i^x\hat{\sigma}_j^x\hat{\sigma}_p^z,$$

$$\hat{a}[\hat{b},\hat{c}]\hat{a} \propto \begin{cases}\hat{\sigma}_i^x\hat{\sigma}_j^x\hat{\sigma}_k^z & \text{if } k=p\\ \hat{\sigma}_i^x\hat{\sigma}_j^x\hat{\sigma}_p^z & \text{else}\end{cases},$$

$$[\hat{a},[\hat{b},\hat{c}]] \propto \begin{cases}\hat{\sigma}_i^y\hat{\sigma}_j^x & \text{if } k=p\\ \hat{\sigma}_i^y\hat{\sigma}_j^x\hat{\sigma}_k^z\hat{\sigma}_p^z & \text{else}\end{cases}.$$

This means that we only need to consider $\hat{\sigma}_i^y\hat{\sigma}_j^y$ and $\hat{\sigma}_i^x\hat{\sigma}_j^x$. We can rewrite the contributing terms as

$$\frac{1}{2}\left(c_{2(h_i-h_j)\gamma}[\hat{\sigma}_i^y\hat{\sigma}_j^y + \hat{\sigma}_i^x\hat{\sigma}_j^x] - c_{2(h_i+h_j)\gamma}[\hat{\sigma}_i^x\hat{\sigma}_j^x - \hat{\sigma}_i^y\hat{\sigma}_j^y]\right)$$

by using the relations

$$s_{2h_i\gamma}s_{2h_j\gamma} = \frac{1}{2}\left[c_{2(h_i-h_j)\gamma} - c_{2(h_i+h_j)\gamma}\right], \tag{C3}$$

$$c_{2h_i\gamma}c_{2h_j\gamma} = \frac{1}{2}\left[c_{2(h_i-h_j)\gamma} + c_{2(h_i+h_j)\gamma}\right]. \tag{C4}$$

For the contributing terms we now consider how terms in $U_c^2(\gamma)$ corresponding to a triangle (i,j,p) act on $\hat{\sigma}_i^y\hat{\sigma}_j^y$ and $\hat{\sigma}_i^x\hat{\sigma}_j^x$. By evaluating the function in Eq. (C2) for $\hat{\sigma}_i^x\hat{\sigma}_j^x$ we get the following contributing terms

$$\begin{aligned}G(\hat{\sigma}_i^x\hat{\sigma}_j^x,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z,\gamma J_{jp}\hat{\sigma}_j^z\hat{\sigma}_p^z) &= c_{2\gamma J_{jp}}c_{2\gamma J_{ip}}\hat{\sigma}_i^x\hat{\sigma}_j^x\\ &+ s_{2\gamma J_{jp}}s_{2\gamma J_{ip}}\hat{\sigma}_i^y\hat{\sigma}_j^y\end{aligned}$$

since

$$\hat{a} = \hat{\sigma}_i^z\hat{\sigma}_p^z, \quad \hat{b} = \hat{\sigma}_j^z\hat{\sigma}_p^z, \quad \hat{c} = \hat{\sigma}_i^x\hat{\sigma}_j^x,$$

$$\hat{a}\hat{c}\hat{a} = -\hat{\sigma}_i^x\hat{\sigma}_j^x, \quad [\hat{a},\hat{c}] = 2i\hat{\sigma}_i^y\hat{\sigma}_j^x\hat{\sigma}_p^z, \quad \hat{b}\hat{c}\hat{b} = -\hat{\sigma}_i^x\hat{\sigma}_j^x,$$

$$\hat{a}(\hat{b}\hat{c}\hat{b})\hat{a} = \hat{\sigma}_i^x\hat{\sigma}_j^x, \ [\hat{a},\hat{b}\hat{c}\hat{b}] = -2i\hat{\sigma}_i^y\hat{\sigma}_j^x\hat{\sigma}_p^z, \ [\hat{b},\hat{c}] = 2i\hat{\sigma}_i^x\hat{\sigma}_j^y\hat{\sigma}_p^z,$$

$$\hat{a}[\hat{b},\hat{c}]\hat{a} = -2i\hat{\sigma}_i^x\hat{\sigma}_j^y\hat{\sigma}_p^z, \quad [\hat{a},[\hat{b},\hat{c}]] = -4\hat{\sigma}_i^y\hat{\sigma}_j^y.$$

The final expression for $\hat{\sigma}_i^x\hat{\sigma}_i^x$ is found by the relation $c_{2a} = (c_a^2 - s_a^2)$. For $\hat{\sigma}_i^y\hat{\sigma}_j^y$ we get the following contributing terms

$$\begin{aligned}G(\hat{\sigma}_i^y\hat{\sigma}_j^y,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z,\gamma J_{jp}\hat{\sigma}_j^z\hat{\sigma}_p^z) &= c_{2\gamma J_{jp}}c_{2\gamma J_{ip}}\hat{\sigma}_i^y\hat{\sigma}_j^y\\ &+ s_{2\gamma J_{jp}}s_{2\gamma J_{ip}}\hat{\sigma}_i^x\hat{\sigma}_j^x.\end{aligned}$$

by inspecting the partial terms we obtain by considering Eq. (C2):

$$\hat{a} = \hat{\sigma}_i^z\hat{\sigma}_p^z, \quad \hat{b} = \hat{\sigma}_j^z\hat{\sigma}_p^z, \quad \hat{c} = \hat{\sigma}_i^y\hat{\sigma}_j^y,$$

$$\hat{a}\hat{c}\hat{a} = -\hat{\sigma}_i^y\hat{\sigma}_j^y, \quad [\hat{a},\hat{c}] = -2i\hat{\sigma}_i^x\hat{\sigma}_j^y\hat{\sigma}_p^z, \quad \hat{b}\hat{c}\hat{b} = -\hat{\sigma}_i^y\hat{\sigma}_j^y,$$

$$\hat{a}(\hat{b}\hat{c}\hat{b})\hat{a} = \hat{\sigma}_i^y\hat{\sigma}_j^y, \ [\hat{a},\hat{b}\hat{c}\hat{b}] = 2i\hat{\sigma}_i^x\hat{\sigma}_j^y\hat{\sigma}_p^z, \ [\hat{b},\hat{c}] = -2i\hat{\sigma}_i^y\hat{\sigma}_j^x\hat{\sigma}_p^z,$$

$$\hat{a}[\hat{b},\hat{c}]\hat{a} = 2i\hat{\sigma}_i^y\hat{\sigma}_j^x\hat{\sigma}_p^z, \quad [\hat{a},[\hat{b},\hat{c}]] = -4\hat{\sigma}_i^x\hat{\sigma}_j^x.$$

By using trigonometric relations

$$c_{a+b} = c_ac_b - s_as_b, \tag{C5}$$

$$c_{a-b} = c_ac_b + s_as_b. \tag{C6}$$

it is clear that the operators corresponding to a triangle (i,j,p) act on $\hat{\sigma}_i^y\hat{\sigma}_j^y + \hat{\sigma}_i^x\hat{\sigma}_j^x$ and $\hat{\sigma}_i^x\hat{\sigma}_j^x - \hat{\sigma}_i^y\hat{\sigma}_j^y$ as

$$\begin{aligned}&G(\hat{\sigma}_i^y\hat{\sigma}_j^y + \hat{\sigma}_i^x\hat{\sigma}_j^x,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z,\gamma J_{jp}\hat{\sigma}_j^z\hat{\sigma}_p^z) =\\ &\qquad c_{2\gamma(J_{jp}-J_{ip})}[\hat{\sigma}_i^y\hat{\sigma}_j^y + \hat{\sigma}_i^x\hat{\sigma}_j^x],\\ &G(\hat{\sigma}_i^x\hat{\sigma}_j^x - \hat{\sigma}_i^y\hat{\sigma}_j^y,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z,\gamma J_{jp}\hat{\sigma}_j^z\hat{\sigma}_p^z) =\\ &\qquad c_{2\gamma(J_{jp}+J_{ip})}[\hat{\sigma}_i^x\hat{\sigma}_j^x - \hat{\sigma}_i^y\hat{\sigma}_j^y].\end{aligned}$$

Furthermore, other terms in $U_c^2(\gamma)$ corresponding to edges (i,k) and (j,l) that are not part of a triangle will give rise to the following expressions for $\hat{\sigma}_i^x\hat{\sigma}_j^x$

$$\begin{aligned}F(\hat{\sigma}_i^x\hat{\sigma}_j^x,\gamma J_{ik}\hat{\sigma}_i^z\hat{\sigma}_k^z) &= c_{2\gamma J_{ik}}\hat{\sigma}_i^x\hat{\sigma}_j^x - s_{2\gamma J_{ik}}\hat{\sigma}_i^y\hat{\sigma}_j^y\hat{\sigma}_k^z,\\ F(\hat{\sigma}_i^x\hat{\sigma}_j^x,\gamma J_{jl}\hat{\sigma}_j^z\hat{\sigma}_l^z) &= c_{2\gamma J_{jl}}\hat{\sigma}_i^x\hat{\sigma}_j^x - s_{2\gamma J_{jl}}\hat{\sigma}_i^x\hat{\sigma}_j^y\hat{\sigma}_l^z\end{aligned}$$

and for $\hat{\sigma}_i^y\hat{\sigma}_j^y$ we get

$$F(\hat{\sigma}_i^y\hat{\sigma}_j^y,\gamma J_{ik}\hat{\sigma}_i^z\hat{\sigma}_k^z) = c_{2\gamma J_{ik}}\hat{\sigma}_i^y\hat{\sigma}_j^y + s_{2\gamma J_{ik}}\hat{\sigma}_i^x\hat{\sigma}_j^y\hat{\sigma}_k^z,$$
$$F(\hat{\sigma}_i^y\hat{\sigma}_j^y,\gamma J_{jl}\hat{\sigma}_j^z\hat{\sigma}_l^z) = c_{2\gamma J_{jl}}\hat{\sigma}_i^y\hat{\sigma}_j^y + s_{2\gamma J_{jl}}\hat{\sigma}_i^y\hat{\sigma}_j^x\hat{\sigma}_l^z.$$

Thus, for $\langle E_{ij}^{yy}\rangle$ we get the following contributing parts to the expectation value

$$\langle E_{ij}^{yy}\rangle = \frac{1}{2}s_{2\beta}^2 \prod_{\substack{(i,k)\in E\\(j,k)\notin E}} c_{2\gamma J_{ik}} \prod_{\substack{(j,l)\in E\\(i,l)\notin E}} c_{2\gamma J_{jl}} \times \Bigg[c_{2(h_i-h_j)\gamma}\prod_{\substack{(i,p)\in E\\(j,p)\in E}} c_{2(J_{ip}-J_{jp})\gamma} - c_{2(h_i+h_j)\gamma}\prod_{\substack{(j,p)\in E\\(i,p)\in E}} c_{2(J_{ip}+J_{jp})\gamma}\Bigg] \tag{C7}$$

since only the $\hat{\sigma}_i^x\hat{\sigma}_j^x$ terms are nonzero when the trace is taken.

For $\langle E_{ij}^{yz}\rangle$ on the other hand, the only non commuting term of $U_c^1(\gamma)$ with $\hat{\sigma}_i^y\hat{\sigma}_j^z$ is $e^{-i\gamma h_i\hat{\sigma}_i^z}$, which results in

$$F(\hat{\sigma}_i^y\hat{\sigma}_j^z,\gamma h_i\hat{\sigma}_i^z) = c_{2\gamma h_i}\hat{\sigma}_i^y\hat{\sigma}_j^z + s_{2\gamma h_i}\hat{\sigma}_i^x\hat{\sigma}_j^z.$$

For the operator, $e^{-iJ_{ij}\gamma\hat{\sigma}_i^z\hat{\sigma}_j^z}$, corresponding to edge (i,j), we have that

$$F(\hat{\sigma}_i^y\hat{\sigma}_j^z,\gamma J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z) = c_{2\gamma J_{ij}}\hat{\sigma}_i^y\hat{\sigma}_j^z + s_{2\gamma J_{ij}}\hat{\sigma}_i^x,$$
$$F(\hat{\sigma}_i^x\hat{\sigma}_j^z,\gamma J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z) = c_{2\gamma J_{ij}}\hat{\sigma}_i^x\hat{\sigma}_j^z - s_{2\gamma J_{ij}}\hat{\sigma}_i^y.$$

Other operators of $U_c^2(\gamma)$ corresponding to edges that include node i further gives the expressions

$$F(\hat{\sigma}_i^y\hat{\sigma}_j^z,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z) = c_{2\gamma J_{ip}}\hat{\sigma}_i^y\hat{\sigma}_j^z + s_{2\gamma J_{ip}}\hat{\sigma}_i^x\hat{\sigma}_j^z\hat{\sigma}_p^z,$$
$$F(\hat{\sigma}_i^x,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z) = c_{2\gamma J_{ip}}\hat{\sigma}_i^x - s_{2\gamma J_{ip}}\hat{\sigma}_i^y\hat{\sigma}_p^z,$$
$$F(\hat{\sigma}_i^x\hat{\sigma}_j^z,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z) = c_{2\gamma J_{ip}}\hat{\sigma}_i^x\hat{\sigma}_j^z - s_{2\gamma J_{ip}}\hat{\sigma}_i^y\hat{\sigma}_j^z\hat{\sigma}_p^z,$$
$$F(\hat{\sigma}_i^y,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z) = c_{2\gamma J_{ip}}\hat{\sigma}_i^y + s_{2\gamma J_{ip}}\hat{\sigma}_i^x\hat{\sigma}_p^z.$$

We can again conclude that since only the $\hat{\sigma}_i^x$ term contributes here, the final contribution to the expectation value is

$$\langle E_{ij}^{yz}\rangle = \frac{s_{4\beta}}{2}c_{2\gamma h_i}s_{2\gamma J_{ij}}\prod_{p\neq j:(i,p)\in E} c_{2\gamma J_{ip}} \tag{C8}$$

and

$$\langle E_{ij}^{zy}\rangle = \frac{s_{4\beta}}{2}c_{2\gamma h_j}s_{2\gamma J_{ij}}\prod_{p\neq i:(j,p)\in E} c_{2\gamma J_{jp}} \tag{C9}$$

by exchanging index i and j.

Finally, for $\langle E_i\rangle$ the only non commuting term of $U_M(\beta)$ is $e^{-i\beta\hat{\sigma}_i^x}$, which results in

$$F(\hat{\sigma}_i^z, e^{-i\beta\hat{\sigma}_i^x}) = c_{2\beta}\hat{\sigma}_i^z + s_{2\beta}\hat{\sigma}_i^y.$$

We can exclude $\hat{\sigma}_i^z$ as it commutes with $U_c(\gamma)$. On the other hand $\hat{\sigma}_i^y$ does not commute with $e^{-i\gamma h_i\hat{\sigma}_i^z}$. The operator $U_c^1(\gamma)$ therefore gives rise to the following expression

$$F(\hat{\sigma}_i^y,\gamma h_i\hat{\sigma}_i^z) = c_{2\gamma h_i}\hat{\sigma}_i^y + s_{2\gamma h_i}\hat{\sigma}_i^x.$$

When we act with $e^{-i\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z}$ for an edge $p:(i,p)\in E$ we get

$$F(\hat{\sigma}_i^y,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z) = c_{2\gamma J_{ip}}\hat{\sigma}_i^y + s_{2\gamma J_{ip}}\hat{\sigma}_i^x\hat{\sigma}_p^z,$$
$$F(\hat{\sigma}_i^x,\gamma J_{ip}\hat{\sigma}_i^z\hat{\sigma}_p^z) = c_{2\gamma J_{ip}}\hat{\sigma}_i^x - s_{2\gamma J_{ip}}\hat{\sigma}_i^y\hat{\sigma}_p^z$$

which only contributes with $c_{2\gamma J_{ip}}\hat{\sigma}_i^x$. The resulting contribution of $\langle E_i\rangle$ to the overall expectation value is therefore

$$\langle E_i\rangle = s_{2\beta}s_{2\gamma h_i}\prod_{p:(i,p)\in E} c_{2\gamma J_{ip}}. \tag{C10}$$

We conclude by noting that if we add all the terms in Eq. (C10), (C9), (C8) and (C7) with their coefficients h_i and J_{ij} we get the expression in Eq. (10).

Appendix D: Success probabilities of Set Partitioning

In this section, a summary is given of the results of ideal simulations of QAOA circuits for all instances applied to the Set Partitioning problem. Table I shows success probabilities for Hamiltonians constructed for factors $f = \infty$, f^* and intermediate choices. As a shorthand, P^f denotes $P_{\text{success}}^{\text{Set Partitioning}}$ given a Set Partitioning Hamiltonian with weights μ_1 and μ_2 for a factor f.

TABLE I: Success probabilities for QAOA applied to Set Partitioning for problem sizes 6-20, given algorithm depth p for multiple choices of factor f

$\|R\|$	p	$\|S_{\text{feasible}}\|$	$P^{f=\infty}$	$P^{f=100}$	$P^{f=10}$	$P^{f=1}$	
6	40	1	99.55	99.52	99.96	99.98	
		2	50.	51.36	59.93	72.67	
		3	32.39	36.	67.43	99.71	
$\|R\|$	p	$\|S_{\text{feasible}}\|$	$P^{f=\infty}$	$P^{f=100}$	$P^{f=10}$	$P^{f=1}$	
8	40	1	99.68	99.54	99.83	99.83	
		2	52.47	57.29	95.46	99.33	
		3	27.62	38.48	99.22	99.99	
		4	26.69	96.68	99.75	99.4	
$\|R\|$	p	$\|S_{\text{feasible}}\|$	$P^{f=\infty}$	$P^{f=100}$	$P^{f=10}$	$P^{f=33.33}$	
10	40	1	96.75	5.19			
		2	51.15	0.28		47.18	
		3	33.37	38.33	99.03		
		4	25.38	39.52	99.95		
		5	18.89	89.3	99.91		
$\|R\|$	p	$\|S_{\text{feasible}}\|$	$P^{f=\infty}$	$P^{f=100}$	$P^{f=10}$	$P^{f=20}$	$P^{f=25}$
12	40	1	82.72	45.85			
		2	60.41	64.52	87.83		
		3	23.78	27.19		40.21	
		4	31.35	37.62			70.46
		5	13.06	20.15	99.5		
		6	16.54	24.31	99.53		
$\|R\|$	p	$\|S_{\text{feasible}}\|$	$P^{f=\infty}$	$P^{f=100}$	$P^{f=10}$		
14	40	1	67.07	38.76			
		2	47.59	63.63	21.06		
		3	34.73	43.78	52.09		
		4	19.94	42.91	99.92		
		5	20.99	31.98	97.46		
		6	20.06	36.09	99.63		
		7	12.25	25.68	99.22		
$\|R\|$	p	$\|S_{\text{feasible}}\|$	$P^{f=\infty}$	$P^{f=10}$			
20	20	1	12.71	12.79			
		2	14.42	12.8			
		3	10.27	12.64			
		4	15.84	12.68			
		5	16.34	17.26			
		6	13.23	17.53			
		7	12.44	19.09			
		8	11.32	86.52			
		9	8.19	77.11			
		10	7.75	86.39			

PAPER B

Benchmarking Advantage and D-Wave 2000Q quantum annealers with exact cover problems

Dennis Willsch, Madita Willsch, Carlos D. Gonzalez Calaza, Fengping Jin, Hans De Raedt, **Marika Svensson**, and Kristel Michielsen

Quantum Inf. Process 21, 141 (2022)

Benchmarking Advantage and D-Wave 2000Q quantum annealers with exact cover problems

Dennis Willsch,[1,*] Madita Willsch,[1,2] Carlos D. Gonzalez Calaza,[1] Fengping Jin,[1] Hans De Raedt,[1,3] Marika Svensson,[4,5] and Kristel Michielsen[1,6]

[1] *Institute for Advanced Simulation, Jülich Supercomputing Centre, Forschungszentrum Jülich, 52425 Jülich, Germany*
[2] *AIDAS, 52425 Jülich, Germany*
[3] *Zernike Institute for Advanced Materials, University of Groningen, Nijenborgh 4, 9747 AG Groningen, The Netherlands*
[4] *Jeppesen, 411 03 Gothenburg, Sweden*
[5] *Department of Computer Science and Engineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden*
[6] *RWTH Aachen University, 52056 Aachen, Germany*

(Dated: May 6, 2021)

We benchmark the quantum processing units of the largest quantum annealers to date, the 5000+ qubit quantum annealer Advantage and its 2000+ qubit predecessor D-Wave 2000Q, using tail assignment and exact cover problems from aircraft scheduling scenarios. The benchmark set contains small, intermediate, and large problems with both sparsely connected and almost fully connected instances. We find that Advantage outperforms D-Wave 2000Q for almost all problems, with a notable increase in success rate and problem size. In particular, Advantage is also able to solve the largest problems with 120 logical qubits that D-Wave 2000Q cannot solve anymore. Furthermore, problems that can still be solved by D-Wave 2000Q are solved faster by Advantage. We find that D-Wave 2000Q can only achieve better success rates for a few very sparsely connected problems.

Keywords: Quantum Computing, Quantum Annealing, Optimization Problems, Benchmarking

I. INTRODUCTION

Quantum annealing is a quantum computing paradigm that relies on quantum fluctuations to solve optimization problems [1–9]. In September 2020, D-Wave Systems has released a quantum annealer with a 5000+ qubit quantum processing unit (QPU) called Advantage [10]. This system has more than twice as many qubits as its predecessor D-Wave 2000Q and an increase in qubit connectivity from 6 to 15 by using the Pegasus topology [11–14]. High expectations have been placed on its computational power, and first independent studies have become available [15–21]. For such a rapidly developing technology, it is an important task for independent researchers to study progress and test new developments.

Conceptually, there are three classes of benchmarks for quantum annealers:

(1) Comparison with detailed real-time simulations of quantum annealing systems based on solving the time-dependent Schrödinger equation [22, 23] or the time-dependent master equation [5, 6, 24–28].

(2) Direct QPU benchmarks (including comparison with other quantum annealing systems and optimization problem solvers) for problems of intermediate size that may or may not need embeddings and solve either real-world or artificial problems [10, 29–38].

(3) Benchmarks of hybrid solvers that use a combination of QPUs and CPUs or GPUs to solve large-scale application problems [11, 17, 39].

The experiments reported in this paper focus on benchmarking the bare QPU performance, thus belonging to benchmarking class (2).

We assess the progress in quantum annealing technology by benchmarking both Advantage and D-Wave 2000Q with exact cover problems. The exact cover problem is an NP-complete problem [40] that has become a prominent application to study quantum annealing [41–45] and gate-based quantum computing [46–50]. In our case, the exact cover problems are derived from the tail assignment problem [51] (see [49] for more information) and represent simplified aircraft scheduling scenarios. Related aircraft assignment problems on quantum annealers have been studied in [52–54].

The essence of the present exact cover problems is shown in Fig. 1(a). In this case, we are given 40 flight routes. Each route contains several out of 472 flights. The task is to find a selection of flight routes such that all 472 flights are covered *exactly* once. On the quantum annealer, each route is represented by a qubit. If a route is to be selected, the corresponding qubit ends up in the state $|1\rangle$ after the measurement. Eventually, each selected route shall be assigned to one airplane. We remark that solving the problem with a given set of routes is a simplification of the general case.

The difficulty of the problem for a quantum computer can be seen by the following counting argument: For 40 routes (i.e. 40 qubits), the number of possible selections

* Corresponding author: Dennis Willsch; d.willsch@fz-juelich.de

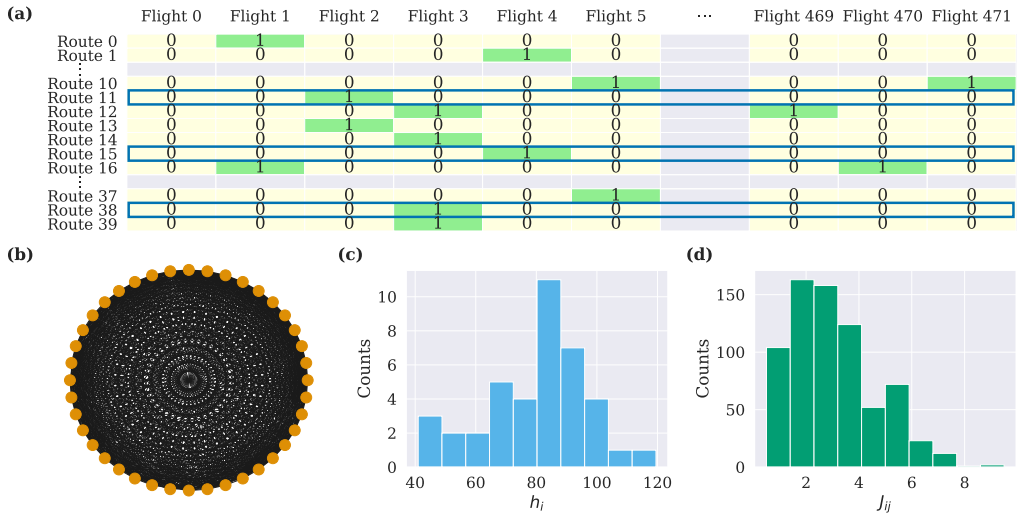


FIG. 1. **Visualization of the exact cover problem with 40 logical qubits (instance 0). (a)** Boolean matrix A defining the exact cover problem instance (see Eq. (13)). For each of the 40 routes, the matrix A indicates all flights that are covered by this route. The exact cover problem is to find a selection of routes (i.e., a subset of rows of the matrix A) such that all 472 flights are covered exactly once (meaning that the sum of the selected rows contains only ones). For this problem, rows belonging to this solution are indicated with blue boxes. The full solution is given by the ground state $|0100001010010101000000101000100000000000\rangle$ (the rightmost qubit corresponds to route 0). **(b)** Coupler graph of the Ising formulation of this problem (cf. Eq. (2)), where each non-zero J_{ij} corresponds to a black line between the 40 qubits. With 711 out of all $\binom{40}{2} = 780$ couplers being non-zero, this problem is almost fully connected (see also Appendix B). A distribution of the values of the Ising parameters of this problem is shown in **(c)** for the qubit biases h_i given by Eq. (10) and **(d)** for the couplers J_{ij} given by Eq. (11)).

is $2^{40} \approx 10^{12}$. For 120 routes (i.e. 120 qubits), which are the largest problems that are solved in the present benchmark set, the number of selections already grows to 10^{36}. Hence, exact cover problems are well suited for benchmarking the Advantage and D-Wave 2000Q QPUs.

We find that Advantage outperforms D-Wave 2000Q on almost all problems in the present benchmark set up to 120 logical qubits. Advantage can embed and solve larger problems. Furthermore, the time-to-solution on Advantage is at least roughly a factor of two shorter. Advantage scores better success rates for all problems with almost all-to-all connectivity. Only some problems with a very sparse qubit connectivity have sometimes higher success rates on D-Wave 2000Q.

The remainder of this paper is structured as follows. In Sec. II, we describe the mathematical details associated with the exact cover problems under investigation. In Sec. III, we present and discuss the results that Advantage and D-Wave 2000Q have produced for both small-scale and large-scale exact cover problems. Section IV contains our conclusions.

II. METHODS

This section presents the mathematical details behind the problems under investigation and how they are solved on the quantum annealers. We first outline the type of optimization problems that Advantage and D-Wave 2000Q can solve, including the important distinction between *physical* and *logical* qubits. Then, we describe the tail assignment and exact cover problems under investigation, along with their formulation on the D-Wave 2000Q and Advantage QPUs.

A. QUBO and Ising problems

The QPUs produced by D-Wave Systems are designed to solve binary quadratic models (BQMs), i.e., quadratic optimization problems over discrete variables that can each take two different values. BQMs are typically formulated as quadratic unconstrained binary optimization

(QUBO) models or Ising models:

$$\text{QUBO}: \quad \min_{x_i=0,1} \left(\sum_{i \le j} x_i Q_{ij} x_j + C_1 \right), \tag{1}$$

$$\text{Ising}: \quad \min_{s_i=\pm 1} \left(\sum_i h_i s_i + \sum_{i<j} J_{ij} s_i s_j + C_2 \right), \tag{2}$$

where the indices i and j range over all qubits. In the QUBO model in Eq. (1), the problem is defined by the QUBO matrix Q with values $Q_{ij} \in \mathbb{R}$, and the binary problem variables are $x_i = 0, 1$. In the Ising model in Eq. (2), the problem is defined by the biases $h_i \in \mathbb{R}$ and the couplers $J_{ij} \in \mathbb{R}$, and the problem variables are $s_i = \pm 1$. An example distribution of h_i and J_{ij} of the problems under investigation is shown in Figs. 1(c) and (d), respectively.

It is worth mentioning that on the D-Wave QPUs, all problems are internally converted into Ising models and (if the autoscaling feature is on) rescaled by a constant factor such that $h_i \in [-2, 2]$ and $J_{ij} \in [-1, 1]$ [55] (note that these ranges might be different for future QPUs). To convert between QUBO and Ising formulation, we use the quantum annealing convention (see also [56])

$$x_i = \frac{1 + s_i}{2}, \tag{3}$$

so that $x_i = 0$ ($x_i = 1$) maps to $s_i = -1$ ($s_i = 1$). Note that in the literature on gate-based quantum computing, also the alternative $x_i = (1 - s_i)/2$ is often used [57] (which would result in a change of sign for the qubit biases, $h_i \mapsto -h_i$)

The constants C_1 and C_2 in Eqs. (1) and (2) do not affect the solution of the problem. However, they can be used to shift the energy (i.e., the value of the objective function at the solution). We use it to shift the energy of the ground state to zero so that we can conveniently determine the success rate by counting all solutions with energy zero.

B. Physical and logical qubits

Many problems may require non-zero couplers Q_{ij} or J_{ij} between different qubits i and j that do not physically exist on the QPUs. For instance, the 40 qubit problem sketched in Fig. 1(b) has almost all-to-all connectivity. In this case, solving the problem on a QPU requires the concept of *embedding* the problem on a QPU.

Conventionally, the qubits that physically exist on a QPU are called *physical* qubits. On a D-Wave 2000Q QPU, the 2000+ physical qubits are connected in a *Chimera* topology [7]. This means that each physical qubit is connected to 6 other physical qubits on average. On an Advantage QPU, the Chimera topology has been upgraded to the *Pegasus* topology [10]. This means that nearly all of the 5000+ physical qubits are connected to 15 other physical qubits, increasing the connectivity by a factor of 2.5. Note that the Pegasus topology is an extension of the Chimera topology, so that a Chimera graph can be natively embedded in a Pegasus graph (see also [17]).

To increase the effective connectivity between qubits, several physical qubits can be combined into a *logical* qubit. The QUBO and Ising models in Eqs. (1) and (2) are typically formulated in terms of such logical qubits, and not the underlying physical qubits. The physical qubits that form a logical qubit are called a *chain*. To ensure that physical qubits within a chain function as a single logical qubit, the couplers J_{ij} between them are set to a reasonably large, negative value called `chain_strength`. If a chain between two qubits *breaks* (i.e., if the product of the Ising variables is $s_i s_j = -1$), a penalty of $2 \times$ `chain_strength` is added to the energy.

We define the `chain_strength` in terms of the *relative chain strength* RCS $\in [0, 1]$ according to

$$\texttt{chain\_strength} = \text{RCS} \times \texttt{max\_strength}, \tag{4}$$

where $\texttt{max\_strength} = \max\left(\{|h_i|\} \cup \{|J_{ij}|\}\right)$ is the maximum absolute value of all h_i and J_{ij}. For instance, for the problem sketched in Fig. 1, `max_strength` would be 119.5.

An *embedding* is a mapping from each logical qubit to a chain of physical qubits. An important property of embeddings are the chain lengths. In our experience, embeddings with too large chains may result in a poor quality of the solutions produced by a QPU. In Appendix B, we provide more details on the specific chains encountered in the embeddings (see Fig. 6).

The D-Wave Ocean SDK [58] provides algorithms to automatically generate qubit embeddings with a given value for `chain_strength`. Still, when using a QPU, finding and characterizing embeddings is an important step and may considerably affect the quality of the solution. For this reason, as part of the present benchmark, we systematically investigate different embeddings and relative chain strengths in Sec. III A.

C. Tail assignment problem

The problem instances considered in this work are derived from the tail assignment problem [51]. The tail assignment problem is a fundamental component of aircraft assignment problems, i.e., the problem of assigning flights to individual airplanes, identified by their tail number. The objective is to minimize the overall cost subject to certain constraints such as minimum connection times, airport curfews, maintenance, and preassigned activities. The general role of tail assignment problems in aircraft scheduling and their relation to the column generation technique [59] and the branch-and-price algorithm [60] is described in detail in [49].

We consider a simple form of the tail assignment problem given by

$$\text{minimize} \quad \sum_{r=0}^{R-1} c_r x_r, \tag{5}$$

$$\text{subject to} \quad \sum_{r=0}^{R-1} A_{rf} x_r = 1 \quad \forall f = 0, \ldots, F-1, \tag{6}$$

where $r = 0, \ldots, R-1$ enumerates all flight routes, $f = 0, \ldots, F-1$ enumerates the flights, $x_r \in \{0, 1\}$ are the Boolean problem variables with $x_r = 1$ if route r is to be selected, c_r is the cost of selecting route r, and $A \in \{0, 1\}^{R \times F}$ is the Boolean problem matrix with $A_{rf} = 1$ if flight f is contained in route r (see Fig. 1(a) for an example of the matrix A). Further models for the tail assignment problem can be found in [47, 49, 51, 54].

A BQM version of the tail assignment problem given by Eqs. (5) and (6) is

$$\min_{\vec{x} \in \{0,1\}^N} \left(\lambda \vec{c}^T \vec{x} + \left(A^T \vec{x} - \vec{b} \right)^2 \right), \tag{7}$$

where the number of qubits $N = R$ is given by the number of flight routes, $\vec{c} = (c_0, \ldots, c_{R-1})^T$ contains the costs, $\vec{b} = (1, \ldots, 1)^T$ is an F-dimensional vector of ones. Note that the scaling factor λ in Eq. (7) is the inverse of the penalty multiplier that would determine the scale of the constraint $A\vec{x} = \vec{b}$ in Eq. (6). It was put in front of the cost function $\vec{c}^T \vec{x}$ so that the exact cover version of the problem corresponds to $\lambda = 0$ (see Sec. II D).

We obtain the QUBO formulation of the tail assignment problem by multiplying out the square in Eq. (7) and collecting all terms into the general QUBO model Eq. (1). An outline of the calculation is given in Appendix A. After doing this, we can read off the entries of the QUBO matrix as

$$Q_{ij} = \begin{cases} (2AA^T)_{ij} & (i < j) \\ (AA^T)_{ii} - (2A\vec{b})_i + \lambda c_i & (i = j) \end{cases}, \tag{8}$$

$$C_1 = \vec{b}^T \vec{b}. \tag{9}$$

Note that the qubit indices $i, j \in \{0, \ldots, N-1\}$ correspond to the previous route index $r \in \{0, \ldots, R-1\}$.

Finally, the Ising formulation of the tail assignment problem is found by using Eq. (3) to replace the qubit variables x_i by spin variables s_i in the QUBO model Eq. (1). After collecting linear, quadratic, and constant terms, we find the expressions for the coefficients of the Ising model Eq. (2),

$$h_i = \sum_j \frac{1}{2}(AA^T)_{ij} - (A\vec{b})_i + \frac{1}{2}\lambda c_i, \tag{10}$$

$$J_{ij} = \frac{1}{2}(AA^T)_{ij}, \tag{11}$$

$$C_2 = C_1 + \sum_{i<j} \frac{1}{2}(AA^T)_{ij} + \sum_i \frac{1}{2}((AA^T)_{ii} - (2A\vec{b})_i + \lambda c_i). \tag{12}$$

These expressions hold for general values of A and $\vec{b}$ (details on the calculation can also be found in Appendix A). A characteristic distribution of the biases h_i and the couplers J_{ij} is shown in Figs. 1(c) and (d), respectively.

D. Exact cover problem

The exact cover problem is an NP-complete set partitioning problem [40]. In matrix form, it can be written as

$$\min_{x_r = 0,1} \sum_{f=0}^{F-1} \left(\sum_{r=0}^{R-1} A_{rf} x_r - 1 \right)^2, \tag{13}$$

and its purpose can be directly understood from Fig. 1(a): The selection of routes (i.e., rows) with $x_r = 1$ has to be such that each flight f in the problem matrix A is covered exactly once.

The exact cover problem corresponds to the feasibility version of the tail assignment problem given by the sole constraints in Eq. (6), without the cost function in Eq. (5). Formally, the exact cover problems are obtained by setting $\lambda = 0$ in Eq. (7), which yields Eq. (13).

Hence, the QUBO (Ising) coefficients of the exact cover problem are given by Eq. (8) (Eqs. (10) and (11)) for $\lambda = 0$ (see also [43]). Note that the explicit expressions for the constants in Eqs. (9) and (12) are not relevant for the solution of the problem on the quantum annealer. However, it is convenient to add them to the resulting energies to ensure that the energy minimum is zero, because then we can determine the success rate by counting the occurrences of samples with energy zero.

Further details on the the exact cover problems, including the number of logical couplers and physical qubits used in the generated embeddings, can be found in Appendix B.

III. RESULTS

In this section, we present the benchmark results for Advantage and D-Wave 2000Q. We first consider small and intermediate exact cover problems with 30–40 logical

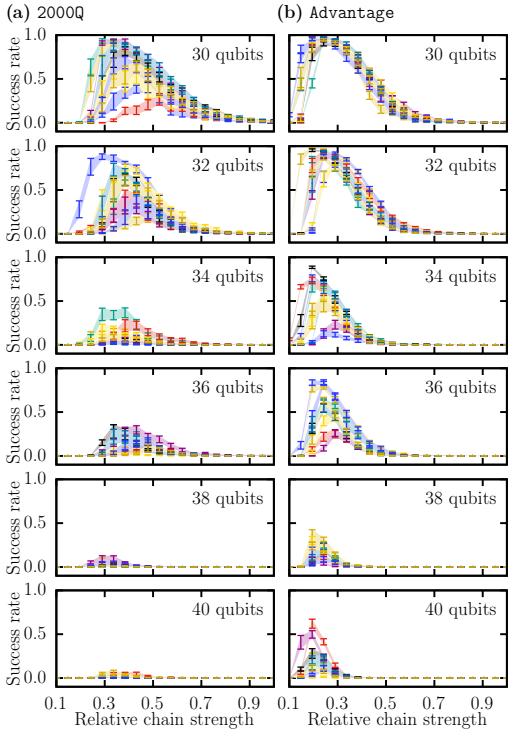


FIG. 2. **Success rates for exact cover problems with 30–40 qubits (instance 0) as a function of the relative chain strength (bottom axis) on (a) D-Wave 2000Q and (b) Advantage with default annealing time** $20\,\mu$s**.** Each run is repeated for 10 different embeddings (represented by different colors) and with 10 repetitions to gather statistics. Markers indicate the corresponding standard deviation above and below the mean. Filled areas between the markers are guides to the eye. The curves for the success rates as a function of the RCS are representative of the other problem instances 1, 2, and 3 characterized in Appendix B.

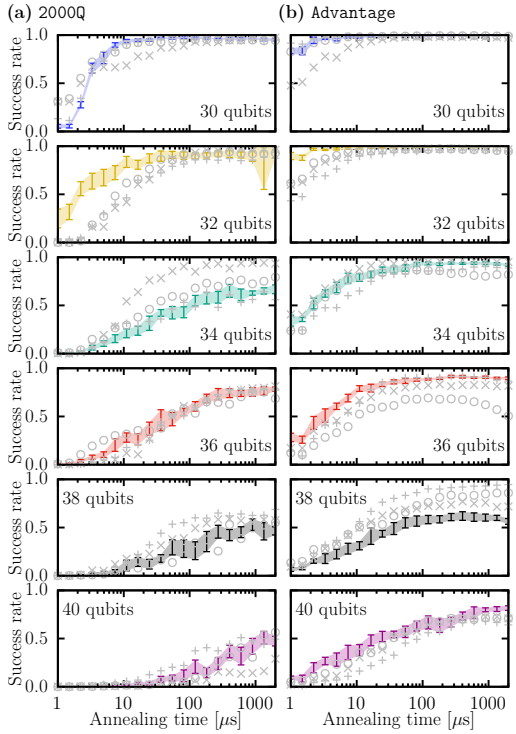


FIG. 3. **Success rates as a function of the annealing time on (a) D-Wave 2000Q and (b) Advantage, averaged over 10 repetitions.** Different markers indicate different problem instances: 0 (markers with error bars), 1 (pluses), 2 (crosses), 3 (circles). For each instance, the runs in each panel are performed with the same embeddings and relative chain strengths, characterized in Fig. 6 in App. B. For instance 0 (highlighted in color), these parameters correspond to the best configuration of the corresponding panel in Fig. 2. Additionally, for instance 0, the standard deviation from the 10 repetitions is indicated by the filled areas between the error bars.

qubits and almost full connectivity, with a focus on comparing different embeddings and annealing times. Then, we proceed to large exact cover problems with up to 120 logical qubits, with a focus on the success rate and the time that it takes the QPUs to solve the problems. For the sake of completeness, we also present results for tail assignment problems with $\lambda \neq 0$ (cf. Eq. (7)) in Appendix C.

The experiments reported in this section were performed in August and September 2020 using the solver `DW_2000Q_VFYC_6` for D-Wave 2000Q and the solvers `Advantage_beta` and `Advantage_system1.1` for Advantage.

A. Densely connected problems with 30–40 qubits

The problems with $N = 30, 32, 34, 36, 38, 40$ qubits and almost full connectivity (cf. Appendix B) are exact cover problems from aircraft scheduling scenarios with 472 flights. Each qubit represents a flight route that contains some of the 472 flights (cf. Fig. 1). We remark that by construction, the ground state of each problem instance is unique and contains 9 qubits in state $|1\rangle$. We obtain the success rate by counting the number of samples with energy zero (note that the value of the constant Eq. (12) can be used to shift the energy accordingly).

1. *Characterizing embeddings and chain strengths*

For each problem with $N = 30, 32, 34, 36, 38, 40$ qubits, we generate 10 different embeddings on both D-Wave 2000Q and Advantage. For each embedding, we scan the RCS (cf. Eq. (4)). The results are shown in Fig. 2.

We see that the results on Advantage (Fig. 2(b)) are generally much better than on D-Wave 2000Q, especially for larger problems. The reason for this is that the 30–40 qubit problems are almost fully connected (cf. Figs. 1(b) and 5). Hence, in such cases, the user can clearly profit from the much larger connectivity between qubits on the Pegasus topology.

This observation is in line with the fact that the chains on Advantage for the same problem are much shorter (see Fig. 6(b)). We also see that for increasing problem size, generating multiple embeddings and tuning the chain strengths is crucial to obtain good results when using the bare QPUs.

2. *Scanning the annealing time*

Next, we study the influence of the annealing time on the success rate. For this, we first select the best embedding and relative chain strength for each problem, based on the results from the previous section. For problem instance 0, for example, this configuration corresponds to the position of the peaks in each panel in Fig. 2. For the selected configuration, we then replace the default annealing time of $20\,\mu$s by 20 different, logarithmically spaced annealing times in the QPU annealing time range $[1\,\mu\text{s}, 2000\,\mu\text{s}]$. The results for the 30–40 qubit problems and all four problem instances are shown in Fig. 3.

Comparing D-Wave 2000Q and Advantage, we can make the following observations: First, Advantage reaches higher maximum success rates, typically for the longest annealing time. Second, the success rates on Advantage are already reasonable for short annealing times. This means that Advantage is typically faster than D-Wave 2000Q (see also the comparison of QPU access times in Fig. 4(b) the following section). Finally, the fluctuations over 10 repetitions are smaller on Advantage (see the filled areas in Fig. 3). We note that this observation also holds for the three instances whose statistics are not indicated in Fig. 3 (gray markers). Thus we conclude that Advantage shows a demonstrable advantage over D-Wave 2000Q.

B. Large problems with 50–120 qubits

In this section, we scale up the problem size to exact cover problems with 50 to 120 logical qubits. The goal is to assess the scaling potential of the QPUs. For each problem size, we consider six problem instances. Each corresponds to an aircraft scheduling problem of the type sketched in Fig. 1(a) with 535 flights. As before, the ground state of these exact cover problems is unique and known. It has 40 qubits in state $|1\rangle$.

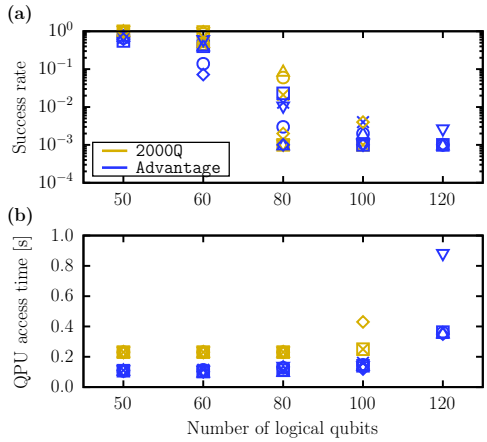


FIG. 4. **Solution of exact cover problems with 50–120 logical qubits on both D-Wave 2000Q (yellow) and Advantage (blue).** Shown are **(a)** the success rate as a function of the number of logical qubits and **(b)** the QPU access time (which mainly consists of programming, annealing, and readout times [61]) . Different markers indicate different problem instances: 0 (crosses), 1 (squares), 2 (circles), 3 (up-pointing triangles), 4 (down-pointing triangles), 5 (diamonds). The embedding and the relative chain strength for these runs was selected using the same procedure as in Sec. III A.

These large problems require many more physical qubits so that their embedded versions can occupy a large part of the QPUs. For this reason, the success rates may be smaller, and especially D-Wave 2000Q may be not able to solve the largest problems anymore.

Indeed, as Fig. 4(a) shows, only Advantage can solve five out of the six 120 qubit instances. The success rates for the 80 and 100 qubit instances are comparable for D-Wave 2000Q and Advantage. Only for the 50 and 60 qubit instances, we see that D-Wave 2000Q yields sometimes better success rates than Advantage.

The reason for this can be understood by studying the number of couplers required for these problems: As shown in Fig. 5 in Appendix B, the 50 and 60 qubit problems need comparably little couplers, so they have a sparser connectivity. Therefore, many of the physical couplers present in the Pegasus topology on Advantage are not required. In other words, the problems can already be embedded well enough on the Chimera topology. We suspect that in this case, the additional unused connections between qubits on Advantage may disturb the annealing process, because even if they are not used, they still exist physically. This observation is in line with a similar observation for Advantage reported in [17]. It can also be observed for superconducting qubits that im-

plement the gate-based quantum computer model; see e.g. the crosstalk experiment in [62].

Finally, the large problems may require more time on the QPUs, so that they are well suited to compare QPU run times between D-Wave 2000Q and Advantage. Unless mentioned otherwise, the same number of reads and annealing times have been used for D-Wave 2000Q and Advantage to ensure a fair comparison. The timing results are shown in Fig. 4(b).

For 50–80 qubits, the annealing time is less than or equal to $20\,\mu$s, so for 1000 reads it does not make up a significant fraction of the QPU access time. Under these conditions, we see that D-Wave 2000Q still needs at least twice as long to solve the problem. Thus, we infer that it is a speedup in programming and readout times [61] that make Advantage faster than D-Wave 2000Q.

For problems with 100 and 120, the annealing time needs to be significantly increased to find the ground state, which is visible in the QPU access times. For instance, problem instance 5 for 100 qubits on D-Wave 2000Q (the single yellow diamond at 100 qubits in Fig. 4(b)) corresponds to an annealing time of $200\,\mu$s. The same annealing time is required for problem instances 1, 2, 3, and 5 for 120 qubits on Advantage (the blue cluster at 120 qubits in Fig. 4(b)). Still, Advantage solves these problems faster than D-Wave 2000Q solves the corresponding 100 qubit problem. Only instance 4 for 120 qubits stands out (the blue down-pointing triangle in Fig. 4(b)): Here, an annealing time of $2000\,\mu$s was required to find a solution. In order not to exceed the maximum run time on Advantage, the number of reads was reduced to 400. Thus, $400 \times 2000\,\mu\text{s} = 0.8\,\text{s}$ makes up the largest part of the QPU access time in this case.

IV. CONCLUSION

In this paper, we have benchmarked the performance of the 2000+ qubit quantum annealer D-Wave 2000Q and the 5000+ qubit quantum annealer Advantage. The benchmark suite consists of intermediate and large exact cover problems from aircraft scheduling scenarios with both sparse and dense logical qubit connectivity.

We observed a considerable increase in performance on Advantage. First, Advantage was able to solve exact cover problems with up to 120 logical qubits that were unsolvable on D-Wave 2000Q. Second, the success rates produced by Advantage were almost always higher. Third, Advantage is approximately twice as fast as D-Wave 2000Q in terms of both programming and readout times. Additionally, the required annealing times to solve a problem on Advantage are often shorter. Finally, the fluctuations in success rates over several repetitions on Advantage were smaller.

A large part of the increase in performance can be attributed not only to the larger number of physical qubits, but rather to the increase in qubit connectivity: Every qubit in the Pegasus topology is connected to 15 other qubits, as compared to 6 other qubits in the Chimera topology used in D-Wave 2000Q. We observed chain lengths in the embeddings that were roughly smaller by a factor of two. We could only observe slightly better performance on D-Wave 2000Q for problems with very sparse qubit connectivity. These problems could already be well embedded on the Chimera topology. We conjecture that for these problems, the presence of the additional couplers in the Pegasus topology may slightly disturb the results (see also [17]).

When using the bare QPUs, it is essential to scan several embeddings and chain strengths to find optimal results. Furthermore, it is important to tune the annealing time. We note that besides the bare QPUs, we have also submitted all exact cover problems of our benchmark set to D-Wave's hybrid solver services, which use a combination of QPUs and classical solvers to solve much larger problems [11]. All exact cover problems could be solved by the hybrid solvers `hybrid_v1` (using D-Wave 2000Q) and `hybrid_binary_quadratic_model_version2` (using Advantage) on September 14, 2020. See [17] for more detailed benchmarks of the hybrid solvers with problems of up to 12000 variables.

Our benchmark study confirms the consistent increase in both size and performance of quantum annealers over the past years. For the future, it is an interesting question whether D-Wave Systems will prove capable of keeping up the steep progress of doubling qubit numbers and increasing performance and qubit connectivity at the same time.

ACKNOWLEDGMENTS

We thank Manpreet Jattana for presenting parts of the results reported in this paper on Qubits 2020. We thank D-Wave Systems for early access to an Advantage$^{\text{TM}}$ system and the new hybrid solver service in Leap during the Advantage beta program. We gratefully acknowledge the Jülich Supercomputing Centre for funding this project by providing additional computing time through the Jülich UNified Infrastructure for Quantum computing (JUNIQ) on the D-Wave quantum annealer. D.W. and M.W. acknowledge support from the project JUNIQ that has received funding from the Federal Ministry of Education and Research (BMBF) and the Ministry of Culture and Science of the State of North Rhine-Westphalia. C.G. acknowledges support from the project OpenSuperQ (820363) of the EU Quantum Flagship. M.S. acknowledges funding from the Knut and Alice Wallenberg foundation (KAW) through the Wallenberg Centre for Quantum Technology (WACQT).

[1] B. Apolloni, C. Carvalho, and D. de Falco, *Quantum stochastic optimization,* Stoch. Process. Their Appl. **33**, 233 (1989).

[2] A. Finnila, M. Gomez, C. Sebenik, C. Stenson, and J. Doll, *Quantum annealing: A new method for minimizing multidimensional functions,* Chem. Phys. Lett. **219**, 343 (1994).

[3] T. Kadowaki and H. Nishimori, *Quantum annealing in the transverse Ising model,* Phys. Rev. E **58**, 5355 (1998).

[4] J. Brooke, D. Bitko, T. F. Rosenbaum, and G. Aeppli, *Quantum Annealing of a Disordered Magnet,* Science **284**, 779 (1999).

[5] R. Harris, M. W. Johnson, T. Lanting, A. J. Berkley, J. Johansson, P. Bunyk, E. Tolkacheva, E. Ladizinsky, N. Ladizinsky, T. Oh, F. Cioata, I. Perminov, P. Spear, C. Enderud, C. Rich, S. Uchaikin, M. C. Thom, E. M. Chapple, J. Wang, B. Wilson, M. H. S. Amin, N. Dickson, K. Karimi, B. Macready, C. J. S. Truncik, and G. Rose, *Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor,* Phys. Rev. B **82**, 024511 (2010).

[6] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, E. M. Chapple, C. Enderud, J. P. Hilton, K. Karimi, E. Ladizinsky, N. Ladizinsky, T. Oh, I. Perminov, C. Rich, M. C. Thom, E. Tolkacheva, C. J. S. Truncik, S. Uchaikin, J. Wang, B. Wilson, and G. Rose, *Quantum annealing with manufactured spins,* Nature **473**, 194 (2011).

[7] P. I. Bunyk, E. M. Hoskinson, M. W. Johnson, E. Tolkacheva, F. Altomare, A. J. Berkley, R. Harris, J. P. Hilton, T. Lanting, A. J. Przybysz, and J. Whittaker, *Architectural Considerations in the Design of a Superconducting Quantum Annealing Processor,* IEEE Transactions on Applied Superconductivity **24**, 1 (2014).

[8] J. Job and D. Lidar, *Test-driving 1000 qubits,* Quantum Sci. Technol. **3**, 030501 (2018).

[9] P. Hauke, H. G. Katzgraber, W. Lechner, H. Nishimori, and W. D. Oliver, *Perspectives of quantum annealing: methods and implementations,* Rep. Prog. Phys. **83**, 054401 (2020).

[10] C. McGeoch and P. Farré, *The D-Wave Advantage System: An Overview*, Tech. Rep. (D-Wave Systems Inc, Burnaby, BC, Canada, 2020) D-Wave Technical Report Series 14-1049A-A.

[11] C. McGeoch, P. Farré, and W. Bernoudy, *D-Wave Hybrid Solver Service + Advantage: Technology Update*, Tech. Rep. (D-Wave Systems Inc, Burnaby, BC, Canada, 2020) D-Wave User Manual 09-1109A-V.

[12] D-Wave Systems, *Technical Description of the D-Wave Quantum Processing Unit*, Tech. Rep. (D-Wave Systems Inc., Burnaby, BC, Canada, 2020) D-Wave User Manual 09-1109A-V.

[13] K. Boothby, P. Bunyk, J. Raymond, and A. Roy, *Next-Generation Topology of D-Wave Quantum Processors,* arXiv:2003.00133 [quant-ph] (2020).

[14] A. D. King and W. Bernoudy, *Performance benefits of increased qubit connectivity in quantum annealing 3-dimensional spin glasses,* arXiv:2009.12479 [quant-ph] (2020).

[15] J. Cohen and C. Alexander, *Picking Efficient Portfolios from 3,171 US Common Stocks with New Quantum and Classical Solvers,* arXiv:2011.01308 [quant-ph] (2020).

[16] M. Kuramata, R. Katsuki, and K. Nakata, *Larger Sparse Quadratic Assignment Problem Optimization Using Quantum Annealing and a Bit-Flip Heuristic Algorithm,* arXiv:2012.10135 [quant-ph] (2020).

[17] C. D. G. Calaza, D. Willsch, and K. Michielsen, *Garden optimization problems for benchmarking quantum annealers,* arXiv:2101.10827 [quant-ph] (2021).

[18] T. Birdal, V. Golyanik, C. Theobalt, and L. Guibas, *Quantum Permutation Synchronization,* arXiv:2101.07755 [quant-ph] (2021).

[19] H. S. Bhatia and F. Phillipson, *Performance Analysis of Support Vector Machine Implementations on the D-Wave Quantum Annealer*, in *International Conference on Computational Science* (Springer, 2021) pp. 1–14.

[20] D. M. Fox, K. M. Branson, and R. C. Walker, *mRNA codon optimization on quantum computers,* 10.1101/2021.02.19.431999 (2021).

[21] S. A. Rahman, R. Lewis, E. Mendicelli, and S. Powell, *SU(2) lattice gauge theory on a quantum annealer,* arXiv:2103.08661 [hep-lat] (2021).

[22] M. Willsch, D. Willsch, F. Jin, H. De Raedt, and K. Michielsen, *Real-time simulation of flux qubits used for quantum annealing,* Phys. Rev. A **101**, 012327 (2020).

[23] M. Willsch, *Study of quantum annealing by simulating the time evolution of flux qubits*, Ph.D. thesis, RWTH Aachen University, Aachen (2020).

[24] S. Boixo, T. Albash, F. M. Spedalieri, N. Chancellor, and D. A. Lidar, *Experimental signature of programmable quantum annealing,* Nat. Commun. **4**, 2067 (2013).

[25] T. Albash, T. Rønnow, M. Troyer, and D. Lidar, *Reexamining classical and quantum models for the D-Wave One processor,* Eur. Phys. J. Spec. Top. **224**, 111 (2015).

[26] T. Albash, W. Vinci, A. Mishra, P. A. Warburton, and D. A. Lidar, *Consistency tests of classical and quantum models for a quantum annealer,* Phys. Rev. A **91**, 042314 (2015).

[27] S. Boixo, V. N. Smelyanskiy, A. Shabani, S. V. Isakov, M. Dykman, V. S. Denchev, M. H. Amin, A. Y. Smirnov, M. Mohseni, and H. Neven, *Computational multiqubit tunnelling in programmable quantum annealers,* Nat. Commun. **7**, 10327 (2016).

[28] J. Marshall, D. Venturelli, I. Hen, and E. G. Rieffel, *Power of Pausing: Advancing Understanding of Thermalization in Experimental Quantum Annealers,* Phys. Rev. Applied **11**, 044083 (2019).

[29] S. Boixo, T. F. Rønnow, S. V. Isakov, Z. Wang, D. Wecker, D. A. Lidar, J. M. Martinis, and M. Troyer, *Evidence for quantum annealing with more than one hundred qubits,* Nat. Phys. **10**, 218 (2014).

[30] T. F. Rønnow, Z. Wang, J. Job, S. Boixo, S. V. Isakov, D. Wecker, J. M. Martinis, D. A. Lidar, and M. Troyer, *Defining and detecting quantum speedup,* Science **345**, 420 (2014).

[31] J. Hall, M. Novotny, T. Neuhaus, and K. Michielsen, *A Study of Spanning Trees on a D-Wave Quantum Computer,* Physics Procedia **68**, 56 (2015), proceedings of the 28th Workshop on Computer Simulation Studies in Condensed Matter Physics (CSP2015).

[32] I. Hen, J. Job, T. Albash, T. F. Rønnow, M. Troyer, and

D. A. Lidar, *Probing for quantum speedup in spin-glass problems with planted solutions,* Phys. Rev. A **92**, 042325 (2015).

[33] C. C. McGeoch, *Benchmarking D-Wave quantum annealing systems: some challenges,* in *Electro-Optical and Infrared Systems: Technology and Applications XII; and Quantum Information Science and Technology*, Vol. 9648, edited by D. A. Huckridge, R. Ebert, M. T. Gruneisen, M. Dusek, and J. G. Rarity, International Society for Optics and Photonics (SPIE, 2015) pp. 264 – 273.

[34] M. Novotny, Q. L. Hobl, J. Hall, and K. Michielsen, *Spanning Tree Calculations on D-Wave 2 Machines,* J. Phys. Conf. Ser. **681**, 012005 (2016).

[35] R. Y. Li, R. Di Felice, R. Rohs, and D. A. Lidar, *Quantum annealing versus classical machine learning applied to a simplified computational biology problem,* npj Quantum Inf. **4**, 14 (2018).

[36] M. Willsch, D. Willsch, F. Jin, H. De Raedt, and K. Michielsen, *Benchmarking the quantum approximate optimization algorithm,* Quantum Inf. Process. **19**, 197 (2020).

[37] D. Willsch, M. Willsch, H. De Raedt, and K. Michielsen, *Support vector machines on the D-Wave quantum annealer,* Comput. Phys. Commun. **248**, 107006 (2020).

[38] G. Cavallaro, D. Willsch, M. Willsch, K. Michielsen, and M. Riedel, *Approaching Remote Sensing Image Classification with Ensembles of Support Vector Machines on the D-Wave Quantum Annealer,* in *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium* (2020) pp. 1973–1976.

[39] S. Mugel, C. Kuchkovsky, E. Sanchez, S. Fernandez-Lorenzo, J. Luis-Hita, E. Lizaso, and R. Orus, *Dynamic Portfolio Optimization with Real Datasets Using Quantum Processors and Quantum-Inspired Tensor Networks,* arXiv:2007.00017 [quant-ph] (2020).

[40] R. M. Karp, *Reducibility among Combinatorial Problems,* in *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department*, edited by R. E. Miller, J. W. Thatcher, and J. D. Bohlinger (Springer US, Boston, MA, 1972) pp. 85–103.

[41] 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 **292**, 472 (2001).

[42] V. Choi, *Adiabatic Quantum Algorithms for the NP-Complete Maximum-Weight Independent Set, Exact Cover and 3SAT Problems,* arXiv:1004.2226 [quant-ph] (2010).

[43] A. Lucas, *Ising formulations of many NP problems,* Front. Phys. **2**, 5 (2014).

[44] Y. Cao, S. Jiang, D. Perouli, and S. Kais, *Solving Set Cover with Pairs Problem using Quantum Annealing,* Sci. Rep. **6**, 33957 (2016).

[45] I. Sax, S. Feld, S. Zielinski, T. Gabor, C. Linnhoff-Popien, and W. Mauerer, *Approximate Approximation on a Quantum Annealer,* in *Proceedings of the 17th ACM International Conference on Computing Frontiers*, CF '20 (Association for Computing Machinery, New York, NY, USA, 2020) pp. 108–117.

[46] E. Farhi, J. Goldstone, and S. Gutmann, *A Quantum Approximate Optimization Algorithm,* arXiv:1411.4028 [quant-ph] (2014).

[47] P. Vikstål, M. Grönkvist, M. Svensson, M. Andersson, G. Johansson, and G. Ferrini, *Applying the Quantum Approximate Optimization Algorithm to the Tail-Assignment Problem,* Phys. Rev. Applied **14**, 034009 (2020).

[48] A. Bengtsson, P. Vikstål, C. Warren, M. Svensson, X. Gu, A. F. Kockum, P. Krantz, C. Križan, D. Shiri, I.-M. Svensson, G. Tancredi, G. Johansson, P. Delsing, G. Ferrini, and J. Bylander, *Improved Success Probability with Greater Circuit Depth for the Quantum Approximate Optimization Algorithm,* Phys. Rev. Applied **14**, 034010 (2020).

[49] M. Svensson, M. Andersson, M. Grönkvist, P. Vikstål, D. Dubhashi, G. Ferrini, and G. Johansson, *A Heuristic Method to solve large-scale Integer Linear Programs by combining Branch-and-Price with a Quantum Algorithm,* arXiv:2103.15433 [quant-ph] (2021).

[50] D. Willsch, M. Willsch, F. Jin, K. Michielsen, and H. De Raedt, *GPU-accelerated simulations of quantum annealing and the quantum approximate optimization algorithm,* arXiv:2104.03293 [quant-ph] (2021).

[51] M. Grönkvist, *The Tail Assignment Problem*, Ph.D. thesis, Chalmers University of Technology and Göteborg University (2005).

[52] T. Stollenwerk, E. Lobe, and M. Jung, *Flight Gate Assignment with a Quantum Annealer,* in *Quantum Technology and Optimization Problems* (Springer International Publishing, 2019) pp. 99–110.

[53] T. Stollenwerk, B. O'Gorman, D. Venturelli, S. Mandrà, O. Rodionova, H. Ng, B. Sridhar, E. G. Rieffel, and R. Biswas, *Quantum Annealing Applied to De-Conflicting Optimal Trajectories for Air Traffic Management,* IEEE Transactions on Intelligent Transportation Systems **21**, 285 (2020).

[54] L. Martins, *Applying Quantum Annealing to the Tail Assignment Problem*, Ph.D. thesis, University of Porto (2020).

[55] D-Wave Systems, *D-Wave Solver Properties and Parameters*, Tech. Rep. (D-Wave Systems Inc., Burnaby, BC, Canada, 2021) D-Wave User Manual 09-1169A-S.

[56] D-Wave Systems, *D-Wave Problem-Solving Handbook*, Tech. Rep. (D-Wave Systems Inc., Burnaby, BC, Canada, 2020) D-Wave User Manual 09-1171A-G.

[57] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, New York, 2010).

[58] D-Wave Systems, *D-Wave Ocean SDK,* (2020), `https://github.com/dwavesystems/dwave-ocean-sdk`, release 2.5.0.

[59] J. Desrosiers and M. E. Lübbecke, *A Primer in Column Generation,* in *Column Generation*, edited by G. Desaulniers, J. Desrosiers, and M. M. Solomon (Springer US, Boston, MA, 2005) pp. 1–32.

[60] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, *Branch-and-Price: Column Generation for Solving Huge Integer Programs,* Oper. Res. **46**, 316 (1998).

[61] D-Wave Systems, *Solver Computation Time*, Tech. Rep. (D-Wave Systems Inc., Burnaby, BC, Canada, 2021) D-Wave User Manual 09-1107B-A.

[62] D. Willsch, *Supercomputer simulations of transmon quantum computers*, Ph.D. thesis, RWTH Aachen University, Aachen (2020).

Appendix A: Derivation of QUBO and Ising models

In this appendix, we outline the derivation of the QUBO and Ising coefficients from the BQM in Eq. (7) for the tail assignment and exact cover problems under investigation.

To obtain the QUBO formulation, we first multiply out the square in Eq. (7),

$$\left(A^T\vec{x}-\vec{b}\right)^2 = \left(A^T\vec{x}-\vec{b}\right)^T\left(A^T\vec{x}-\vec{b}\right) \tag{A1}$$

$$= \vec{x}^T AA^T\vec{x} - \vec{b}^T A^T\vec{x} - \vec{x}^T A\vec{b} + \vec{b}^T\vec{b} \tag{A2}$$

$$= \sum_{ij} x_i (AA^T)_{ij} x_j - \sum_i (2A\vec{b})_i x_i + \vec{b}^T\vec{b}. \tag{A3}$$

After splitting the first sum into three parts, $\sum_{ij} = \sum_{i<j} + \sum_{i>j} + \sum_i$, and exchanging $i \leftrightarrow j$ in the second part (since the matrix AA^T is symmetric), we obtain the upper triangular coefficients of the QUBO matrix, $Q_{ij} = (2AA^T)_{ij}$ for $i < j$, and the first part of the diagonal coefficients $(AA^T)_{ii}$. The second sum in Eq. (A3) yields, after using $x_i = x_i x_i$, the remaining part of the diagonal coefficients. The last term yields the constant contribution to the QUBO model. Combining this with the linear term $\lambda\vec{c}^T\vec{x}$ in Eq. (7), we obtain all coefficients of the QUBO model $\sum_{i\le j} x_i Q_{ij} x_j + C_1$,

$$Q_{ij} = \begin{cases} (2AA^T)_{ij} & (i<j) \\ (AA^T)_{ii} - (2A\vec{b})_i + \lambda c_i & (i=j) \end{cases}, \tag{A4}$$

$$C_1 = \vec{b}^T\vec{b} = F. \tag{A5}$$

To obtain the coefficients h_i and J_{ij} of the corresponding Ising model, we replace the qubit variables by spin variables, $x_i = (1+s_i)/2$ (cf. Eq. (3)),

$$\sum_{i\le j} x_i Q_{ij} x_j = \sum_{i\le j} \frac{(1+s_i)}{2} Q_{ij} \frac{(1+s_j)}{2} \tag{A6}$$

$$= \sum_{i<j} \frac{(1+s_i)}{2} (2AA^T)_{ij} \frac{(1+s_j)}{2} + \sum_i \frac{(1+s_i)}{2} ((AA^T)_{ii} - (2A\vec{b})_i + \lambda c_i) \frac{(1+s_i)}{2} \tag{A7}$$

$$= \sum_{i<j} \frac{1}{2}(AA^T)_{ij} s_i + \sum_{i>j} \frac{1}{2}(AA^T)_{ij} s_i + \sum_i \frac{1}{2}(AA^T)_{ii} s_i - \sum_i (A\vec{b})_i s_i + \sum_i \frac{1}{2}\lambda c_i s_i \tag{A8}$$

$$+ \sum_{i<j} \frac{1}{2}(AA^T)_{ij} s_i s_j \tag{A9}$$

$$+ \sum_{i<j} \frac{1}{2}(AA^T)_{ij} + \sum_i \frac{1}{2}((AA^T)_{ii} - (2A\vec{b})_i + \lambda c_i), \tag{A10}$$

where we used that the matrix AA^T is symmetric and that $s_i^2 = 1$. Note that this calculation does not make use of prior knowledge about the values of A and $\vec{b}$.

We can then identify the coefficients of the Ising model $\sum_i h_i s_i + \sum_{i<j} J_{ij} s_i s_j + C_2$ as

$$h_i = \sum_j \frac{1}{2}(AA^T)_{ij} - (A\vec{b})_i + \frac{1}{2}\lambda c_i, \tag{A11}$$

$$J_{ij} = \frac{1}{2}(AA^T)_{ij}, \tag{A12}$$

$$C_2 = C_1 + \sum_{i<j} \frac{1}{2}(AA^T)_{ij} + \sum_i \frac{1}{2}((AA^T)_{ii} - (2A\vec{b})_i + \lambda c_i). \tag{A13}$$

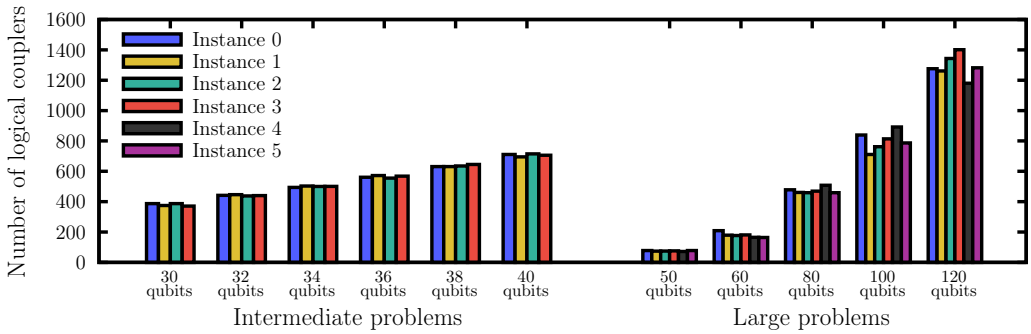


FIG. 5. **Number of logical couplers required for the exact cover problems in the present benchmark set.** The intermediate problems with 30–40 qubits contain 4 different problem instances for each problem size (left). The large problems with 50–120 qubits contain 6 different problem instances (right). Note that after the embedding, the numbers of physical qubits and couplers required on the QPUs may be much larger (cf. Fig. 6(a)).

Appendix B: Exact cover problem details

In this appendix, we provide details on the exact cover problems used in the present benchmark set. For each problem instance, Fig. 5 shows the number of couplers required between the qubits. Note that the 30–40 qubit problems require almost all-to-all connectivity.

In Fig. 6, we provide details on the generated embeddings for the intermediate 30–40 qubit problems. In Fig. 6(a), we list the number of physical qubits required in the embeddings on D-Wave 2000Q and Advantage. As these problems require almost full connectivity, the number of physical qubits is much larger than the number of logical qubits, especially on D-Wave 2000Q. We see that D-Wave 2000Q needs more than twice as many physical qubits, but also the slope as a function of the logical qubits is steeper. This is reasonable as also the number of logical couplers increases (cf. Fig. 5), and with sparser connectivity on the Chimera topology, more physical qubits need to be chained into a logical qubit.

This trend is also visible when looking at the chain lengths shown in Fig. 6(b): While the chains on Advantage stay almost constant for increasing problem sizes, the chains on D-Wave 2000Q grow longer on average. Especially for 40 qubits, chains on D-Wave 2000Q can be up to 19 physical qubits long. Such chains are almost always broken and may lead to a wrong value for the logical qubit that they represent (cf. Sec. II B).

In Fig. 6(c), we plot the relative chain strengths that produced the best results. For each $N = 30, 32, 34, 36, 38, 40$, the first point (instance 0) corresponds to the peak with the optimal success rate in the corresponding panel in Fig. 2.

Appendix C: Tail assignment problems with $\lambda \neq 0$

The tail assignment problem introduced in Sec. II C contains an objective function that represents the cost associated with each flight route (cf. Eq. (5)). Depending on the magnitude of these cost terms, the multiplier λ in the BQM version of the problem (see Eq. (7)) has to be adjusted to put a reasonable weight on the cost function with respect to the constraints. Therefore, we test several values of λ for a 25 qubit problem. For each λ, we generate 10 embeddings (cf. Sec. II B) and evaluate the success rate on D-Wave 2000Q and Advantage. The unique ground state was found using both a linear program solver and exact enumeration of all 2^{25} states on a GPU. We remark that the 25 qubit tail assignment problem solved in this section is the same problem that was investigated as the largest problem instance in [47].

The results as a function of the embedding are shown in Fig. 7. We see that for relatively large λ, the success rates for D-Wave 2000Q fluctuate strongly as a function of the embedding. In such cases, it may become possible to violate some of the constraints to obtain a better value of the cost function (cf. Eq. (7)). However, as λ approaches zero (Figs. 7(b)–(d)) and the problem approaches its exact cover version, most embeddings yield the optimal solution with unit probability, especially on Advantage (unfilled markers).

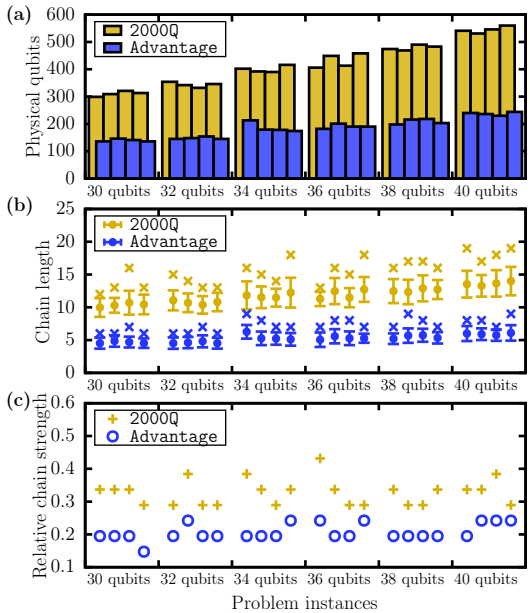


FIG. 6. **Details on the optimal embeddings found for the intermediate exact cover problems with $N = 30, 32, 34, 36, 38, 40$ logical qubits on D-Wave 2000Q (yellow) and Advantage (blue).** For each N, the four problem instances 0, 1, 2, and 3 are shown from left to right on the bottom axes. **(a)** Number of physical qubits needed for the best embeddings; **(b)** lengths of the corresponding physical qubit chains (cf. Sec. II B), where the markers with error bars indicate the mean and the standard deviation, and crosses indicate the maximum chain lengths; **(c)** optimal relative chain strengths, taken from the positions of the corresponding peaks in Fig. 2.

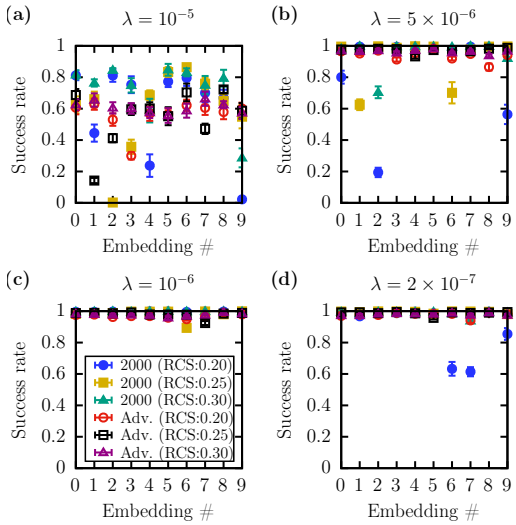


FIG. 7. **Success rates for the tail assignment problem on D-Wave 2000Q (filled markers) and Advantage (unfilled markers) as a function of the relative chain strength RCS given by Eq. (4) (see legend).** Each run is performed for 10 different embeddings (bottom axis). For a given embedding, the markers indicate the mean and the standard deviation for 10 repetitions using the same embedding. The formulation of the tail assignment problem given by Eq. (7) for 25 qubits is solved for decreasing values of the scaling factor **(a)** $\lambda = 10^{-5}$, **(b)** $\lambda = 5 \times 10^{-6}$, **(c)** $\lambda = 10^{-6}$, **(d)** $\lambda = 2 \times 10^{-7}$.

PAPER C

Applying the Quantum Approximate Optimization Algorithm to the Tail Assignment Problem

Pontus Vikstål, Mattias Grönkvist, **Marika Svensson**, Martin Andersson, Göran Johansson, and Giulia Ferrini

Phys. Rev. Applied 14, 034009 (2020)

Applying the Quantum Approximate Optimization Algorithm to the Tail Assignment Problem

Pontus Vikstål,[1,*] Mattias Grönkvist,[2] Marika Svensson,[1,2]
Martin Andersson,[2] Göran Johansson,[1] and Giulia Ferrini[1]

[1] *Wallenberg Centre for Quantum Technology, Department of Microtechnology and Nanoscience, Chalmers University of Technology, 412 96 Gothenburg, Sweden*
[2] *Jeppesen Systems AB, 411 03 Gothenburg, Sweden.*

(Dated: May 21, 2020)

Airlines today are faced with a number of large scale scheduling problems. One such problem is the Tail Assignment problem, which is the task of assigning individual aircraft to a given set of flights, minimizing the overall cost. Each aircraft is identified by the registration number on its tail fin. In this article, we simulate the Quantum Approximate Optimization Algorithm (QAOA) applied to instances of this problem derived from real world data. The QAOA is a variational hybrid quantum-classical algorithm recently introduced and likely to run on near-term quantum devices. The instances are reduced to fit on quantum devices with 8, 15 and 25 qubits. The reduction procedure leaves only one feasible solution per instance, which allows us to map the Tail Assignment problem onto the Exact Cover problem. We find that repeated runs of the QAOA identify the feasible solution with close to unit probability for all instances. Furthermore, we observe patterns in the variational parameters such that an interpolation strategy can be employed which significantly simplifies the classical optimization part of the QAOA. Finally, we empirically find a relation between the connectivity of the problem graph and the single-shot success probability of the algorithm.

I. INTRODUCTION

Real world planning and scheduling problems typically require heuristic algorithms, which is also the case for the Tail Assignment problem. The problem is to assign a set of flights to a set of aircraft in order to create a feasible flight schedule for an airline, while minimizing the overall cost [1].

Recently, quantum computing hardware has reached the regime where it is possible to run quantum algorithms which are hard to simulate on classical hardware, even considering the world's largest supercomputer [2]. This motivates the search for a heuristic quantum algorithm for solving the Tail Assignment problem. A promising approach for this is the Quantum Approximate Optimization Algorithm (QAOA) [3], which is a heuristic hybrid quantum-classical algorithm designed for solving combinatorial optimization problems. Since the algorithm was first proposed by Farhi et al. [3] it has been an active area of research interest [4–10], mainly because of its promising possibility to run on a near term Noisy Intermediate Scale Quantum (NISQ) device. An important open question is whether a quantum computer in general can provide advantages with regards to such classically hard combinatorial optimization problems. Recent studies have indicated that QAOA can have a quadratic Grover type speed up for state transfer and unstructured search problems [11, 12]. Although these results are promising, the performance is largely unknown for QAOA with respect to real world optimization problems.

Here we present, to our knowledge, the first results for QAOA when applied to a real world aircraft assignment problem. We perform numerical simulations of an ideal quantum computer to investigate the performance of QAOA for solving the simplified case of the Tail Assignment problem where all costs are equal to *zero*. This simplified case can be mapped onto the Exact Cover problem [13]. In this context, we note that the solution of random instances of the Exact Cover and of its restricted version Exact Cover by 3-sets on a quantum annealer has been considered in Refs. [14–19]. QAOA for Exact Cover has recently been executed on a 2-qubit quantum computer in a proof-of-principle experiment by some of the authors of the present paper, and collaborators [20].

The paper is organized as follows. In Sec. II, we introduce the Tail Assignment problem, and we explain how we extract the Exact Cover instances that we analyze in this work. In Sec. III, we review the QAOA and explain how it can be utilized to solve the Exact Cover problem. Then, in Sec. IV we present numerical results of the performance of QAOA with respect to the Tail Assignment problem-extracted instances of Exact Cover for three different problem sizes. Specifically, we look at the dependence of the success probability as a function of the algorithm iteration level p and of the problem size. Finally, in Sec. V we present what implications these results might have for solving the Tail Assignment problem.

II. THE TAIL ASSIGNMENT PROBLEM

Airlines are daily confronted with several complicated large-scale planning problems involving many different

* e-mail: vikstal@chalmers.se

resource types such as passengers, crew, aircraft, maintenance and ground staff. The typical airline planning process [21] is a sequential process which starts with the construction of a timetable, followed by a number of aircraft and crew planning steps. These steps are all large scale optimization problems and have different objectives, but the overall goal is to maximize profit, safety and crew satisfaction while minimizing the potential for disruptions. At the same time a large number of complex regulatory, operational and quality constraints must be satisfied.

The Tail Assignment problem [1] is one of the fleet planning problems where the goal is to decide which individual aircraft (or tail, from the aircraft tail identification number) should operate each flight. A set of flights operated in sequence by the same aircraft is called a *route*. In order for a route to be considered legal to operate, it needs to satisfy a number of constraints. For example, the buffer time between the arrival of a flight and the departure of the next flight in the route (the *turn time*) must be above a certain threshold, called the minimum turn time. The minimum turn time can depend on the type of flights involved (domestic/international), the airport, the time of day and possibly even the individual aircraft characteristics. Another type of constraint is a destination restriction, which prohibits specific aircraft from visiting certain airports, for example due to limited engine thrust combined with short runways. Curfew restrictions are timed restrictions, typically limiting noisy aircraft from operating during night hours at centrally placed airports. Finally, routes must satisfy a number of long and short term maintenance constraints. This typically means that the aircraft must regularly visit some airport with a maintenance facility for long enough to perform maintenance.

Now, let F denote the set of flights, T the set of tails and R the set of all legal routes. Denote by c_r the cost of route $r \in R$ and by C_f the cost of leaving flight f unassigned. The route cost can for example indicate how robust the route is with respect to disruptions, what the fuel cost is for the route, or a combination of several different criteria. Let a_{fr} be 1 if flight f is covered by route r and 0 otherwise, and let b_{tr} be 1 if route r uses tail t and 0 otherwise. The decision variable x_r is 1 if route r should be used in the solution, and 0 otherwise. The variables u_f and v_t are 1 if flight f is left unassigned or tail t is unused, respectively, and 0 otherwise. The Tail Assignment problem can now be formulated as

$$\text{minimize} \quad \sum_{r\in R} c_r x_r + \sum_{f\in F} C_f u_f, \tag{1}$$

$$\text{subject to} \quad \sum_{r\in R} a_{fr} x_r + u_f = 1, \quad \forall f \in F, \tag{2}$$

$$\sum_{r\in R} b_{tr} x_r + v_t = 1, \quad \forall t \in T, \tag{3}$$

$$x_r, u_f, v_t \in \{0, 1\} \tag{4}$$

The objective (1) is to minimize the total cost of the selected routes, subject to constraints (2) ensuring that each flight is assigned to exactly one route and constraints (3) ensuring that each tail is used at most once. Flights can be left unassigned at a cost C_f, but that cost is typically very high compared to the route costs. Not using an aircraft does not come with any penalty cost. The model is an example of a Set Partitioning problem, which is NP-hard [22].

A. Solving the Tail Assignment Problem

Clearly, the number of legal routes for a Tail Assignment instance increases exponentially with the number of flights. Since the model presented above requires all the legal routes to be enumerated, it only works for small instances. The solution method traditionally used for these types of models is *column generation* [1]. Column generation starts from some initial solution and uses information from the linear programming dual problem to dynamically generate new variables (columns in the constraint matrix) which are known to potentially improve the current solution. In the Tail Assignment case, the problem of generating improving variables turns out to be a resource constrained shortest path problem. Given mild conditions on the variable generation step, the column generation process can be shown to guarantee optimality for the LP relaxation of the problem, i.e. without the integrality conditions Eq. (4). To find an optimal solution for the full problem including the integrality conditions, column generation must be combined with tree search. The combination of tree search and column generation is often called *branch-and-price* [23].

B. Instances extraction

For the purposes of this article, given the current capability of quantum computers, we will focus on Tail Assignment instances where we have artificially limited the number of routes. The instances have originally been solved using a branch-and-price heuristic, and we have randomly selected a number of routes from the set of all generated routes to create instances of specific sizes. The solution found by the branch-and-price heuristic is always included, so we know that all instances have a solution with all flights assigned. This means that we can skip the u_f variables in the model. We also have uniquely assigned start flights for each aircraft, which means that constraints Eq. (3) can be omitted. Finally, in the remainder of this article we will focus on the decision version of the Tail Assignment problem where the goal is to find any solution satisfying all the constraints, disregarding the costs c_r. This decision version of the Set Partitioning problem is called the Exact Cover problem, it is know to be NP-complete [24], and can be expressed

as the following optimization problem:

$$\begin{aligned} &\text{minimize} && 0 && (5)\\ &\text{subject to} && \sum_{r\in R} a_{fr}x_r = 1, \quad \forall f \in F, && (6)\\ & && x_r \in \{0,1\}, && (7)\end{aligned}$$

where the minimization on 0 is left to recall that this formulation stems from the Tail Assignment problem, where we neglect the costs in Eq.(1). Despite the simplification introduced, the Exact Cover problem is still very relevant for the study of Tail Assignment as many airlines, including for example Air France, consider the Tail Assignment problem to be a pure feasibility problem [25].

III. QAOA APPLIED TO THE TAIL ASSIGNMENT PROBLEM

A large class of NP-complete optimization problems including the Exact Cover (and even many NP-hard problems) can naturally be expressed as the problem of finding the ground state, or minimum energy configuration, of a quantum Ising Hamiltonian [26]

$$\hat{H}_C = \sum_{i<j} J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z + \sum_{i=1}^{n} h_i\hat{\sigma}_i^z. \tag{8}$$

We will refer to this quantum Ising Hamiltonian as a cost Hamiltonian. In this section, we derive explicitly the cost Hamiltonian corresponding to the Exact Cover problem expressed by Eq. (6) and (7). Later, we recall the QAOA algorithm, and in particular how it makes use of the cost Hamiltonian for finding its minimum energy configuration.

A. Ising formulation of the Exact Cover problem

Consider the formulation of the Exact Cover problem presented in Eq. (6) and (7). By subtracting 1 from both sides of Eq. (6) and squaring the expression an energy function formulation is obtained:

$$\mathcal{E}(s_1,\ldots,s_{|R|}) = \sum_{f=1}^{|F|}\left(\sum_{r=1}^{|R|} a_{fr}x_r - 1\right)^2. \tag{9}$$

Here $|R|$ and $|F|$ denote the cardinality of R and F, respectively. We see that all constraints are satisfied if the energy (9) is equal to *zero*.

By replacing the binary variables $x_r \in \{0,1\}$ with spin variables $s_r \in \{-1,1\}$ as

$$x_r = \frac{s_r+1}{2}, \tag{10}$$

and expanding the square of Eq. (9) we obtain the *Ising energy function* for the Exact Cover problem

$$\begin{aligned}\mathcal{E}(s_1,\ldots,s_{|R|}) = \sum_{f=1}^{|F|}\left(\sum_{r=1}^{|R|} a_{fr}\frac{s_r+1}{2} - 1\right)^2 = \\ +\frac{1}{4}\sum_{f=1}^{|F|}\sum_{r=1}^{|R|}\sum_{r'=1}^{|R|} a_{fr}a_{fr'}s_rs_{r'} \\ +\frac{1}{2}\sum_{f=1}^{|F|}\sum_{r=1}^{|R|} a_{fr}s_r\left(\sum_{r'=1}^{|R|} a_{fr'} - 2\right) \\ +\frac{1}{4}\sum_{f=1}^{|F|}\left(\sum_{r=1}^{|R|} a_{fr} - 2\right)^2. \end{aligned} \tag{11}$$

By defining $J_{rr'}$ as

$$J_{rr'} \equiv \frac{1}{2}\sum_{f=1}^{|F|} a_{fr}a_{fr'}, \tag{12}$$

and h_r as

$$h_r \equiv \frac{1}{2}\sum_{f=1}^{|F|} a_{fr}\left(\sum_{r'=1}^{|R|} a_{fr'} - 2\right), \tag{13}$$

the Ising energy function becomes

$$\frac{1}{2}\sum_{r=1}^{|R|}\sum_{r'=1}^{|R|} J_{rr'}s_rs_{r'} + \sum_{r=1}^{|R|} h_rs_r + \text{const.} \tag{14}$$

where the constant is equal to $\frac{1}{4}\sum_{f=1}^{|F|}\left(\sum_{r=1}^{|R|} a_{fr} - 2\right)^2$. The sum of all the diagonal terms ($i=j$) in the first sum is equal to $\mathrm{Tr}(J)$ since $s_i^2 = 1$; because J_{ij} is symmetric i.e. $J_{ij} = J_{ji}$, we can further simplify the expression and write the Ising energy function as

$$\mathcal{E}(s_1,\ldots,s_{|R|}) = \sum_{r<r'} J_{rr'}s_rs_{r'} + \sum_{r=1}^{|R|} h_rs_r + \text{const}, \tag{15}$$

where we have absorbed $\frac{1}{2}\mathrm{Tr}(J)$ into the constant. Finally, by promoting the spin variables to Pauli spin matrices $s_i \to \hat{\sigma}_i^z$, a cost Hamiltonian in the form of Eq. (8) is obtained.

B. The Quantum Approximate Optimization Algorithm

The QAOA starts from an initial quantum state which is taken as a superposition of all possible computational basis states $|+\rangle^{\otimes n}$. The second step of QAOA is to apply in an alternating sequence two parametrized non-commuting quantum gates, $\hat{U}(\gamma)$ and $\hat{V}(\beta)$, that are defined as:

$$\hat{U}(\gamma) \equiv e^{-i\gamma\hat{H}_C}, \quad \hat{V}(\beta) \equiv e^{-i\beta\hat{H}_M}, \tag{16}$$

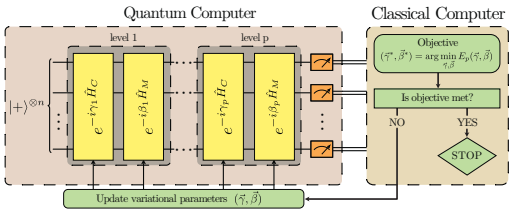


FIG. 1. Schematic representation of the QAOA. The quantum processor prepares the variational state, depending on variational parameters. The variational parameters $(\vec{\gamma}, \vec{\beta})$ are optimized in a closed loop using a classical optimizer.

where $\hat{H}_C$ is the cost Hamiltonian given by Eq. (8), and $\hat{H}_M \equiv \sum_{i=1}^{n} \hat{\sigma}_i^x$ is a so called mixing Hamiltonian. The alternating sequence continues for a total of p times with different variational parameters $\vec{\gamma} = (\gamma_1, \ldots, \gamma_p)$ with $\gamma_i \in [0, 2\pi]$ if $\hat{H}_C$ has integer-valued eigenvalues, and $\vec{\beta} = (\beta_1, \ldots, \beta_p)$ with $\beta_i \in [0, \pi]$, such that the final variational state obtained is:

$$|\psi_p(\vec{\gamma}, \vec{\beta})\rangle \equiv \hat{V}(\beta_p)\hat{U}(\gamma_p) \ldots \hat{V}(\beta_1)\hat{U}(\gamma_1) |+\rangle^{\otimes n} . \quad (17)$$

The parametrized quantum gates are then optimized in a closed loop using a classical optimizer, see Fig. 1. The objective of the classical optimizer is to find the optimal variational parameters that minimize the expectation value of the cost Hamiltonian

$$(\vec{\gamma}^*, \vec{\beta}^*) = \arg\min_{\vec{\gamma}, \vec{\beta}} E_p(\vec{\gamma}, \vec{\beta}), \quad (18)$$

where

$$E_p(\vec{\gamma}, \vec{\beta}) \equiv \langle \psi_p(\vec{\gamma}, \vec{\beta}) | \hat{H}_C | \psi_p(\vec{\gamma}, \vec{\beta}) \rangle . \quad (19)$$

Note that this requires in principle multiple state preparations and measurements. Once the best possible variational parameters are found, they are used to create the state $|\psi_p(\vec{\gamma}^*, \vec{\beta}^*)\rangle$, using the quantum processor for the state preparation. Then, one samples from this state by measuring in the computational basis, and the cost of the configuration obtained in the measurement, given by Eq. (8), is evaluated. The latter step is classically efficient.

The success probability is defined as the probability of finding the qubits in their ground state configuration $|x_{\text{sol}}\rangle$ when performing a single shot measurement of the $|\psi_p(\vec{\gamma}, \vec{\beta})\rangle$ state, i.e.

$$F_p(\vec{\gamma}, \vec{\beta}) \equiv |\langle x_{\text{sol}} | \psi_p(\vec{\gamma}, \vec{\beta}) \rangle|^2, \quad (20)$$

where $x_{\text{sol}} = x_1 x_2 \ldots x_n$ is the bit string corresponding to the solution. Given this success probability we can ask: what is the probability of having observed the solution at least once after m repeated measurements? The answer is given by:

$$1 - (1 - F_p(\vec{\gamma}, \vec{\beta}))^m. \quad (21)$$

Thus to have the probability $(1 - \varepsilon)$ of observing the solution, m has to be

$$m > \frac{\log \varepsilon}{\log (1 - F_p(\vec{\gamma}, \vec{\beta}))}. \quad (22)$$

To fix the ideas, consider a fair coin. In order to have a probability higher than 99.9 % of observing Head at least once, one has to flip and "measure" the coin 10 times.

In what follows, we are going to apply this paradigm to solve the Exact Cover problem, by using the corresponding cost Hamiltonian, expressed by Eq. (8) with J_{ij} and h_i given by Eq. (12) and (13) respectively.

IV. RESULTS

We will examine instances for three different problem sizes of the Tail Assignment problem given in Table I, corresponding to 8, 15 and 25 routes. As clear from Eq. (8), this requires quantum processors with 8, 15 and 25 qubits respectively.

TABLE I. Information about the problem instances.

Routes	Flights	No. of instances	No. of sol. per instance
8	77	10	1
15	77	9	1
25	278	10	1

A. Energy landscape

Firstly, we can reduce the search space by noting that the eigenvalues of both Hamiltonians $\hat{H}_C$ and $\hat{H}_M$ are integer-valued. As a consequence, the expectation value Eq. (19) has even-symmetry, i.e. $E_p(\vec{\gamma}, \vec{\beta}) = E_p(-\vec{\gamma}, -\vec{\beta})$. This symmetry allow us to restrict the domain of each γ_i to $\gamma_i \in [0, \pi]$.

To highlight the difficulty of finding the best variational parameters we can visualize the landscape of the expectation value $E_1(\gamma, \beta)$, as well as the corresponding success probability $F_1(\gamma, \beta)$, as a function of γ and β, for $p = 1$, by evaluating them on a fine grid $[0, \pi] \times [0, \pi]$. Fig. 2 shows the simulation result for one of the 25 route instances. The variational parameters resulting in the lowest expectation value, $(\gamma_{\text{exp}}, \beta_{\text{exp}})$, and those resulting in the highest success probability, $(\gamma_{\text{succ}}, \beta_{\text{succ}})$, are approximately the same. In fact $|\gamma_{\text{exp}} - \gamma_{\text{succ}}| \simeq 0$ and $|\beta_{\text{exp}} - \beta_{\text{succ}}| \simeq 0.047$. Note that this is not obvious, since QAOA only minimizes the expectation value, and does not explicitly maximize the success probability; a low expectation value does not necessarily translates onto a high success probability. For example, consider a variational state that is a linear combination of low energy excited eigenstates of the cost Hamiltonian. This state could potentially have a low expectation value while the

success probability is zero. Similarly, a variational state that is a linear combination of the ground state with high energy eigenstates could have a high success probability, while the cost Hamiltonian expectation value is large. However, in the limit $p \to \infty$, 100 % success probability is always achieved [3]. For our problem, it is clear from Eq. (9) that the minimum energy of the first excited state is at least 1, so if we find an average cost which is lower than 1 for our variational state, we know that the ground state is a part of this state. The corresponding plots for one of the 8 and 15 route instances are shown in Appendix A. We note that all figures have qualitatively similar shape and that the optimal variational parameters for $p = 1$ are located in the same region.

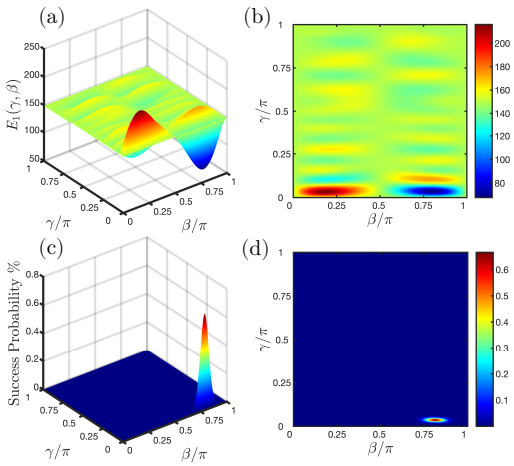


FIG. 2. (color online) Simulation results for one of the 25 route instances as a function of γ and β for $p = 1$. (a) and (b) Expectation value $E_1(\gamma, \beta)$; (c) and (d) Success probability $F_1(\gamma, \beta)$.

B. Low iteration levels: Patterns in optimal variational parameters

Before we look at the performance of QAOA, we will search for patterns in the optimal variational parameters for low iteration levels of the QAOA algorithm, namely up to $p = 5$. Patterns in the optimal variational parameters have been observed before in the context of Max-Cut in Ref. [27], where it was shown that if a pattern exist it is possible to use different heuristics that can drastically speed up the classical optimization part of QAOA. This can potentially help us simulate the solution of our instances for intermediate p-level beyond $p = 5$, namely for $5 < p \leq 20$.

In order to find the optimal variational parameters, one possible approach would consist of a grid search method. However, evaluation of the cost Hamiltonian expectation value on a fine grid for higher dimensions quickly becomes computational expensive due to the large search space $[0, \pi]^p \times [0, \pi]^p$. Therefore, we discard the grid search method and resort to another optimization routine for finding good variational parameters for $1 \leq p \leq 5$. This optimization routine is still exhaustive but more computational efficient. It distributes several random start points in the variational parameter landscape, and runs the gradient based BFGS algorithm [28] for every start point from which it records the global optimum. We provide relevant details in Appendix B. In Fig. 3 we present the optimal variational parameters $(\vec{\gamma}^*, \vec{\beta}^*)$ from $p = 3$ up to $p = 5$ for the 8 route instances. We observe that a persistent pattern shows up, and that both γ_i and β_i tend to increase slowly with $i = 1, 2, \ldots, p$. An analogous analysis for the 15 route instances, shown in the Appendix in Fig. 7, yields a qualitatively similar result. For the 25 route instances, it was not possible to perform this analysis, because for $p > 1$ performing an exhaustive search becomes too computationally expensive.

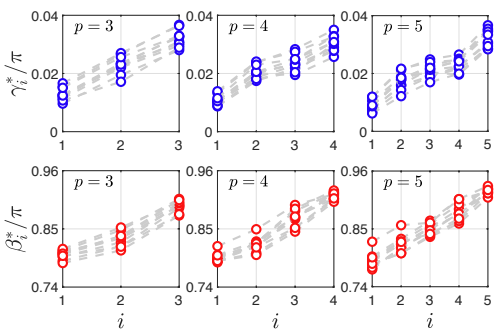


FIG. 3. The optimal QAOA variational parameters $(\vec{\gamma}^*, \vec{\beta}^*)$ for the 8 route instances, for $3 \leq p \leq 5$. The pattern is visualized by plotting the optimal variational parameters where each gray dashed line connects the variational parameters for one 8 route instance.

C. Intermediate iteration levels: Analysis of success probability

Based on the patterns found in the previous section, we now use an interpolation-based strategy, introduced in [27], in order to study the performance of intermediate p-level QAOA. This strategy consists in predicting a good starting point for the variational parameters search at level $p + 1$ for each individual instance based on the best variational parameters found at level p for the same instance. From the produced starting-point we run the gradient-free Nelder-Mead method [29, 30], which is reported in Ref. [27] to work equally well as the BFGS

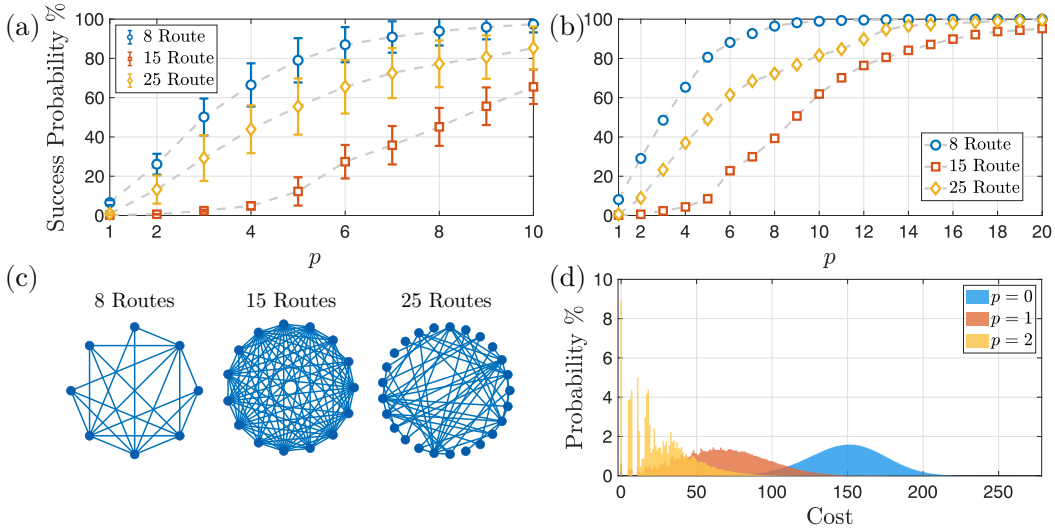


FIG. 4. (color online) (a) Average success probability as a function of the iteration level p using the best found variational parameters for the three different problem sizes. The error-bars in the figure represent the standard deviation of the average success probability. (b) Success probability $F_p(\vec{\gamma}^*, \vec{\beta}^*)$ as a function of p for one selected instance from each problem size. (c) Graph representation of the three instances shown in (b). (d) Probability that a measurement of the state $|\psi_p(\vec{\gamma}^*, \vec{\beta}^*)\rangle$ will yield a certain cost (or equivalently, eigenvalue of the cost Hamiltonian) for the iteration levels $p = 0, 1, 2$, where $p = 0$ is the initial or "random" state $|+\rangle^{\otimes n}$.

method, for this heuristic strategy. The Nelder-Mead algorithm was implemented in MATLAB version R2019b using the `fminsearch` function. Furthermore, in order to force the Nelder-Mead algorithm to terminate after sufficiently many iterations, we set the two stopping criteria - maximum number of function evaluations and iterations - both to $60p$. We furthermore make the assumption that a pattern in the variational parameters also exists in each of the 25 route instances, and we therefore use the interpolation strategy mentioned above for each of these instances as well, as an educated guess. We base this assumption on the qualitatively similar shape of the expectation value landscape that the three different problems sizes investigated had for $p = 1$.

We use the aforementioned interpolation-strategy for finding good local optimal variational parameters up to $p = 10$ for all the instances. The success probability as a function of iteration level p averaged over all the instances for the three different problem sizes is plotted in Fig. 4(a). Moreover, we select one instance from each problem size, for which we perform simulations up to $p = 20$. In Fig. 4(b) we plot the success probability for these three instances. The corresponding variational parameters $\vec{\gamma}^*$ and $\vec{\beta}^*$ are provided in Appendix A, Fig. 8. It is observed that the success probability increases with the parameter p in both the averaged and the single-instance cases, reaching almost 100 % for the instances where we have used high iteration level $p = 20$.

From the results in Figs. 4(a) and 4(b) we also note that the 25 route instances are easier to solve than the 15 route instances, in the sense that the success probability is higher for the former instances at any given iteration level p of the algorithm. This fact can seem counter-intuitive, as one could naively think that larger instances correspond to harder problems. We perform further analysis in order to explain this apparent contradiction.

We start by representing each instance as a graph, by identifying J_{ij} in Eq. (12) with an adjacency matrix. In this way, each vertex in the graph represents a route and two vertices are connected by an edge if they share a flight. The valency of a vertex, i.e. the number of incident edges to the vertex, indicates how many "clauses" the vertex is contained in, or in other words how many other vertices it has to compete with. In Table II we list the average valency of each vertex for the three problem sizes. We note that the 15 instances have more than twice the average valency compared to the 25 route instances. This is also visualized in Fig. 4(c), where the graph connectivity for one instance of each problem size is represented. It is clear that the connectivity for the 15 instance is the most dense. Establishing a general connection between the hardness of the instances and their valency is beyond the scope of our paper. However, such

TABLE II. Valency of the graphs. The first column in the table is the number of routes. The second column is the average valency of a vertex taken as an average over all the instances. The corresponding standard deviation is given in the third column.

Routes	Mean	Standard deviation
8	5.15	0.24
15	12.62	0.42
25	5.54	0.77

a connection is known to exists in some specific contexts, e.g. for Exact Cover by 3-sets [31, 32]. This hints to the fact that such a connection might exist also for our instances, despite they are not in the form of Exact Cover by 3-sets. To elucidate further why denser graphs are more difficult to solve with the QAOA we recall, following Refs. [3, 9], that the expectation value Eq. (19) can be expressed as a sum of expectation values involving all possible subgraphs. Subgraphs are obtained by starting from an edge $\langle ij \rangle$ of a graph, e.g. the type of graph given in Fig. 4(c), and "walking" along the graph at most p steps away from that edge, for a given iteration level p. Indicating with $f_g(\vec{\gamma}, \vec{\beta})$ the contribution to the expectation value from subgraph g, and with w_g the corresponding subgraph occurrence, it is possible to re-write the expectation value as $E_p(\vec{\gamma}, \vec{\beta}) = \sum_g w_g f_g(\vec{\gamma}, \vec{\beta})$. Since the contribution to the expectation value is different for each subgraph, the higher the number of important subgraphs (with a significant w_g) is, the harder it will be to make the cost close to zero for a given iteration level p, since the QAOA need to make each individual term in the sum small. Since the average valency of a graph contributes to the number of subgraphs, this results in a lower success probability for the 15 route instances, as (as we have shown in Fig. 4(c) and Table II) those possess higher average valency.

Finally, in Fig. 4(d) we visualize how the probability of measuring a certain cost, or equivalently an eigenvalue of the cost Hamiltonian, given the state $|\psi_p(\vec{\gamma}^*, \vec{\beta}^*)\rangle$, changes for each iteration $p = 0, 1, 2$ of QAOA using the best found variational parameters for one of the 25 route instances. It is clear that the effect of iterating QAOA is that the probability of configurations with lower cost increases. This validates the effectiveness of QAOA in producing output configurations corresponding to low energy states of the cost Hamiltonian, when the iteration level p is increased. In particular, for $p = 2$ a peak at the zero-cost configuration appears clearly, corresponding to a success probability of 8.97 %. This results in only 74 measurements needed, in order to have a probability greater than 99.9 % of measuring the solution at least once.

In order to benchmark the effectiveness of QAOA in solving this problem against other quantum algorithms, in Appendix C we compare the time to solution of QAOA with that of quantum annealing, and find that QAOA outperforms quantum annealing for all the 8 and 15 route instances.

Finally, noise and imperfection in practical experimental implementations on a quantum computer will induce departures from the obtained success probabilities, and it is an open question whether realistic hardware will still be able to produce the good solution, with satisfactory success probability. Although a complete study of the effect of noise is beyond the scope of the present paper, in Appendix D we characterize the effect of a simple depolarizing noise model, to study how noise affects the performance of QAOA. As expected, we find that with noise an optimal value of p exists. Beyond that value of p, the success probability starts to decrease, due to the larger effect of decoherence when the gate sequence becomes longer. However, for the optimal p, the success probability is only halved, still pointing to relevance of the use of QAOA for solving this problem even in realistic experimental conditions.

V. CONCLUSIONS

In conclusions, we have simulated the solution of instances of the Exact Cover problem that stem as a reduction of the Tail Assignment problem to the case where the goal is to find any solution satisfying all the constraints, using the QAOA.

Our results indicate that these instances can be solved satisfactorily by means of QAOA, yielding relative high success probabilities even for low iteration level of the algorithm. For instance, for the 25 qubits case we obtain a success probability of 8.97 % for $p = 2$ in the single measurement scenario. This corresponds to a success probability of 99.9 % for 74 repeated measurements. This low iteration level translates into a low circuit depth needed for the implementation of this algorithm, corroborating feasibility on a near-term quantum device.

Moreover, we observed patterns for the variational parameters $(\vec{\gamma}, \vec{\beta})$ which allowed for a substantial simplification of the classical optimization problem of finding the best variational parameters, despite the fact that the problem instances have been extracted from a real world problem.

Our analysis has revealed non-trivial properties in the connectivity of the instances considered. I.e., the 15 qubit instances were more connected than the 25 qubit ones. A thorough study of the connectivity and graph-type that are relevant for the Tail Assignment problem in the context of complex quantum networks [33, 34] is beyond the scope of the present paper, but stems as an interesting perspective. Another interesting question is whether the implementation of the QAOA algorithm on hardware with restricted connectivity would still yield non-trivial success probabilities, as shown in Ref. [35] for Max-Cut on three-regular graphs.

Our successful solution with QAOA of small-size instances of Exact Cover extracted from Tail Assignment

motivates further studies, such as the use of QAOA for solving instances with multiple feasible solutions, where costs are re-introduced, and where the number of considered routes is larger, towards tackling real-world instances.

It remains an open question how the performance of QAOA compares with existing classical algorithms for solving large instances of the Exact Cover problem extracted from the Tail Assignment problem.However, we expect that current known methods as Branch-and-Bound, Cutting planes or Branch-and-Cut [36] will perform well on these small instances. Further investigations are needed in order to compare the scaling in terms of time complexity of QAOA fixing a target success probability (i.e. the required iteration level p) and standard classical methods, when the size of the problem increases.

While finalizing this work, we became aware of an alternative method for the optimization of the variational parameters, that makes use of the *Gibbs objective function*, defined as $-\log\langle e^{-\eta\hat{H}_C}\rangle$, where $\eta > 0$, instead of the expectation value Eq. (8) [37]. This approach is expected to be superior because the Gibbs objective function rewards lower energy states, which increases the success probability. We leave the use of this approach for optimization of the variational parameters in our problem to further study.

ACKNOWLEDGMENTS

We thank Devdatt Dubhashi and Eleanor Rieffel for fruitful discussions. This work was supported from the Knut and Alice Wallenberg Foundation through the Wallenberg Center for Quantum Technology (WACQT). G. F. acknowledges financial support from the Swedish Research Council through the VR project QUACVA.

[1] M. Grönkvist, *The Tail Assignment Problem*, Ph.D. thesis, Chalmers University of Technology and Göteborg University (2005).

[2] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandr, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, Nature **574**, 505 (2019).

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

[4] G. E. Crooks, "Performance of the quantum approximate optimization algorithm on the maximum cut problem," (2018), arXiv:1811.08419v1.

[5] M. Willsch, D. Willsch, F. Jin, H. D. Raedt, and K. Michielsen, "Benchmarking the quantum approximate optimization algorithm," (2019), arXiv:1907.02359.

[6] R. Shaydulin and Y. Alexeev, "Evaluating quantum approximate optimization algorithm: A case study," (2019), arXiv:1910.04881.

[7] M. Alam, A. Ash-Saki, and S. Ghosh, "Analysis of quantum approximate optimization algorithm under realistic noise in superconducting qubits," (2019), arXiv:1907.09631.

[8] C. Xue, Z.-Y. Chen, Y.-C. Wu, and G.-P. Guo, "Effects of quantum noise on quantum approximate optimization algorithm," (2019), arXiv:1909.02196.

[9] F. G. S. L. Brandao, M. Broughton, E. Farhi, S. Gutmann, and H. Neven, "For fixed control parameters the quantum approximate optimization algorithm's objective function value concentrates for typical instances," (2018), arXiv:1812.04170.

[10] E. Farhi, J. Goldstone, S. Gutmann, and L. Zhou, "The quantum approximate optimization algorithm and the sherrington-kirkpatrick model at infinite size," (2019), arXiv:1910.08187.

[11] M. Y. Niu, S. Lu, and I. L. Chuang, "Optimizing qaoa: Success probability and runtime dependence on circuit depth," (2019), arXiv:1905.12134.

[12] Z. Jiang, E. G. Rieffel, and Z. Wang, Physical Review A **95**, 062317 (2017).

[13] A. Parmentier and F. Meunier, Omega , 102073 (2019).

[14] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, Science **292**, 472 (2001).

[15] A. P. Young, S. Knysh, and V. N. Smelyanskiy, Phys. Rev. Lett. **104**, 020502 (2010).

[16] B. Altshuler, H. Krovi, and J. Roland, Proceedings of the National Academy of Sciences **107**, 12446 (2010).

[17] V. Choi, "Different adiabatic quantum optimization algorithms for the np-complete exact cover and 3sat problems," (2010), arXiv:1010.1221.

[18] H. Wang and L.-A. Wu, Scientific Reports **6**, 22307 (2016).

[19] T. Graß, Phys. Rev. Lett. **123**, 120501 (2019).

[20] A. Bengtsson, P. Vikstål, C. Warren, M. Svensson, X. Gu, A. Frisk Kockum, P. Krantz, C. Križan, D. Shiri, I.-M. Svensson, G. Tancredi, G. Johansson, P. Delsing, G. Ferrini, and J. Bylander, "Quantum approximate optimization of the exact-cover problem on a superconducting quantum processor," (2019), arXiv:1912.10495.

[21] T. L. Jacobs, L. A. Garrow, M. Lohatepanont, Frank, S. Koppelman, G. M. Coldren, and H. W. Purnomo (2017).

[22] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness* (Freeman, San Fransisco, 1979).

[23] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, Operations Research **46**, 316 (1996).

[24] R. M. Karp, "Reducibility among combinatorial problems," (Springer US, Boston, MA, 1972) pp. 85–103.

[25] A. Parmentier and F. Meunier, Omega **93**, 102073 (2020).

[26] A. Lucas, Frontiers in Physics **2**, 5 (2014).

[27] 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," (2018), arXiv:1812.01041.

[28] C. G. Broyden, IMA Journal of Applied Mathematics **6**, 76 (1970); R. Fletcher, The computer journal **13**, 317 (1970); D. Goldfarb, Mathematics of computation **24**, 23 (1970); D. F. Shanno, Mathematics of computation **24**, 647 (1970).

[29] J. A. Nelder and R. Mead, The computer journal **7**, 308 (1965).

[30] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, SIAM Journal of Optimization **9**, 112 (1998).

[31] V. Kalapala and C. Moore, "The phase transition in exact cover," (2005), arXiv:cs/0508037.

[32] J. Raymond, A. Sportiello, and L. Zdeborová, Physical Review E **76**, 011101 (2007).

[33] F. Sansavini and V. Parigi, "Continuous variables graph states shaped as complex networks: optimization and manipulation," (2019), arXiv:1912.03265.

[34] R. Albert and A.-L. Barabási, Rev. Mod. Phys. **74**, 47 (2002).

[35] E. Farhi, J. Goldstone, S. Gutmann, and H. Neven, "Quantum algorithms for fixed qubit architectures," (2017), arXiv:1703.06199.

[36] M. Conforti, G. Cornuejols, and G. Zambelli, *Integer Programming* (Springer International Publishing, 2014) pp. 5–16.

[37] L. Li, M. Fan, M. Coram, P. Riley, and S. Leichenauer, "Quantum optimization with a novel gibbs objective function and ansatz architecture search," (2019), arXiv:1909.07621v1.

[38] T. Albash and D. A. Lidar, Rev. Mod. Phys. **90**, 015002 (2018).

Appendix A: Additional figures

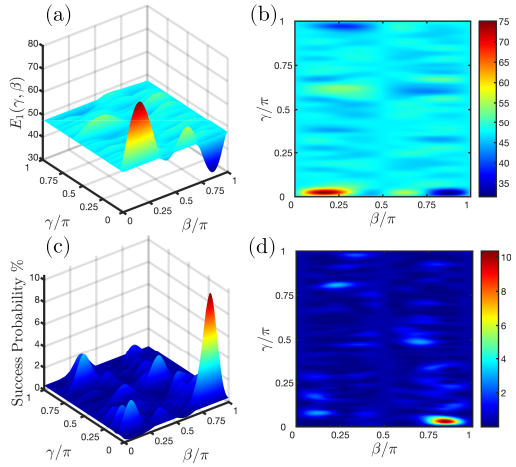


FIG. 5. (color online) Simulation results for one of the 8 route instances as a function of γ and β for $p = 1$. (a) and (b) Expectation value $E_1(\gamma, \beta)$; (c) and (d) Success probability $F_1(\gamma, \beta)$.

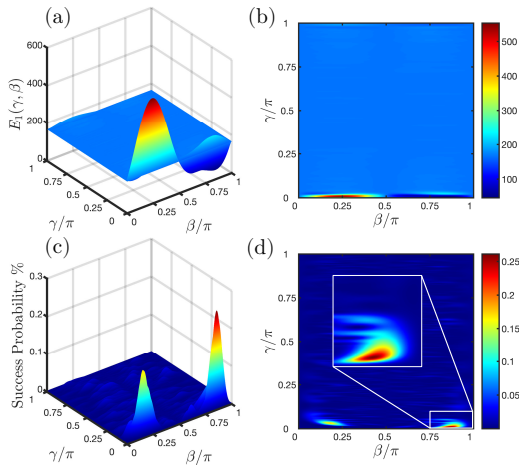


FIG. 6. (color online) Simulation results for one of the 15 route instances as a function of γ and β for $p = 1$. (a) and (b) Expectation value $E_1(\gamma, \beta)$; (c) and (d) Success probability $F_1(\gamma, \beta)$.

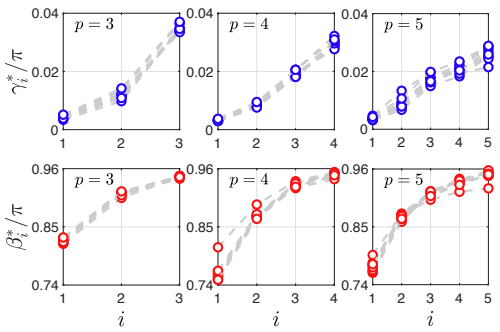


FIG. 7. The optimal QAOA variational parameters $(\vec{\gamma}^*, \vec{\beta}^*)$ for the 15 route instances, for $3 \leq p \leq 5$. The pattern is visualized by plotting the optimal parameters where each grey dashed line connects the optimal variational parameters of one particular instance.

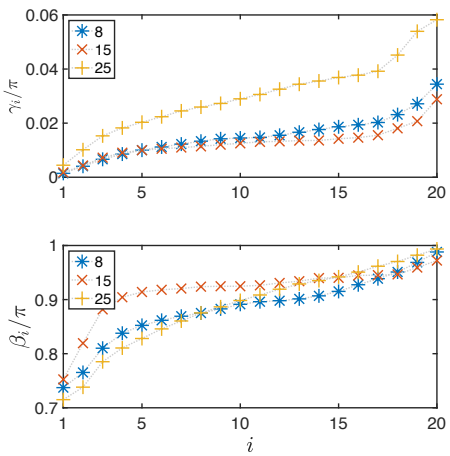


FIG. 8. The best found $\vec{\gamma}^*$ and $\vec{\beta}^*$ for the three instances shown in Fig. 4 (b).

Appendix B: Numerical simulations

The numerical simulations for the exhaustive search method was done in MATLAB version R2019b where the `MultiStart` function was used to search thoroughly for the optimal variational parameters. `MultiStart` attempts to find multiple local minimums to the objective function by starting from various points in the variational parameter landscape. When run, it distributes start points to multiple processors (cpus) that run in parallel. From a start point it runs a *local solver* and when the solver reaches a stopping criterion it terminates and the obtained minima from the solver is stored in an array. When `MultiStart` runs out of start points it stops, and the array with minimums from the solver is sorted by the objective function value in ascending order. The parameters where the objective function is the lowest is then returned as output. As *local solver* we used the BFGS algorithm [28] which is implemented as `fmincon` in MATLAB. The number of random start points was chosen to be 4×10^3. This number was empirically determined by running the simulations a few times for this value and observing that the minimum of the objective function always converged to the same value and gave the same parameters. As mentioned the solver stops when the solver's stopping criteria is met. Two examples of such criterion's are the function tolerance and the step tolerance. The first one, the function tolerance, is a lower bound on the change in the value of the objective function during a step, that is if $|F_p(\vec{\gamma}, \vec{\beta}) - F_p(\vec{\gamma}', \vec{\beta}')| <$ `FunctionTolerance`, the iteration ends. The second one, the step tolerance, is such that if the solver attempts to take a step that is smaller than $|\vec{\gamma} - \vec{\gamma}'|^2 + |\vec{\beta} - \vec{\beta}'|^2 <$ `StepTolerance`, the iteration ends. Both `StepTolerance` and `FunctionTolerance` were set to their default values which was 10^{-6}.

Appendix C: Comparison: Time to solution of Quantum Annealing versus QAOA

In this section we compare the time to solution of the quantum annealing (QA) algorithm with that of the QAOA. In quantum annealing we start from the same initial state as the QAOA, which is in fact the ground state of the mixing Hamiltonian that we use in QAOA, but with a minus sign in front, $\hat{H}_M^{\rm QA} \equiv -\hat{H}_M = -\sum \hat{\sigma}_i^x$. By adiabatically changing from the mixing Hamiltonian to the cost Hamiltonian the system will remain in its instantaneous ground state throughout the evolution, and end up in the ground state of the cost Hamiltonian. For a linear time dependence, the quantum annealing Hamiltonian is given by

$$\hat{H}(t) = \frac{t}{T}\hat{H}_C + \left(1 - \frac{t}{T}\right)\hat{H}_M^{\rm QA}, \quad 0 \leq t \leq T, \qquad \text{(C1)}$$

where $\hat{H}_C$ is the cost Hamiltonian, $\hat{H}_M^{\rm QA}$ is the quantum annealing starting Hamiltonian, and T is the total annealing time. It is known that rather than running the algorithm adiabatically, it can be advantageous to run the algorithm for a shorter time (non fully adiabatically). On the one hand, this yields to a finite probability to excite higher energy states and decreases the success probability on a single run; on the other hand, since the annealing time T is shorter, one can then increase the number of repetitions, yielding an increase of the total success probability, on several runs. Therefore, one can define the *time to solution*, which is a measure of how

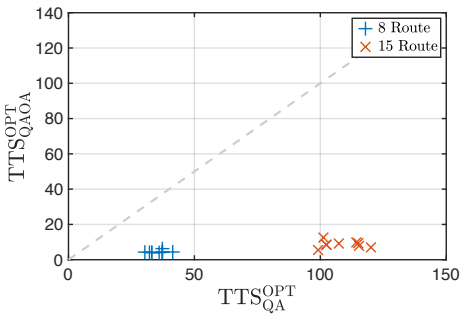


FIG. 9. The optimal time to solution for QAOA and QA. The fact that the markers are below the dotted line means that QAOA outperforms QA in the time required to achieve a 99% success probability.

quickly the algorithm can find the optimal solution. The time to solution for QA is defined by [38]

$$\mathrm{TTS}_{\mathrm{QA}}(T) = T\frac{\log(1-p_{\mathrm{d}})}{\log(1-F_{\mathrm{gs}}(T))},$$

where p_{d} is the target success probability that we fix to 99 %, and $F_{\mathrm{gs}}(T)$ is the single shot success probability after running the algorithm for a time T. The optimal $\mathrm{TTS}_{\mathrm{QA}}(T)$ is thus given by the time T that minimize

$$\mathrm{TTS}^{\mathrm{OPT}}_{\mathrm{QA}} = \min_{T>0} \mathrm{TTS}_{\mathrm{QA}}(T).$$

Following the spirit of Ref. [27], it is possible interpret the sum of the optimal variational parameters of the QAOA as the total "annealing" time that is used, in order to sequentially evolve the system under the action of each of the two Hamiltonians, $T_p = \sum_{i=1}^{p}(|\gamma_i^*| + |\beta_i^*|)$. Thus, the time to solution for QAOA is

$$\mathrm{TTS}_{\mathrm{QAOA}}(p) = T_p\frac{\log(1-p_{\mathrm{d}})}{\log\left(1-F_p(\vec{\gamma}^*,\vec{\beta}^*)\right)},$$

where $F_p(\vec{\gamma}^*,\vec{\beta}^*)$ is given by Eq. (20). Analogously as for QA, the optimal $\mathrm{TTS}_{\mathrm{QAOA}}(p)$ is given by

$$\mathrm{TTS}^{\mathrm{OPT}}_{\mathrm{QAOA}} = \min_{p>0} \mathrm{TTS}_{\mathrm{QAOA}}(p).$$

We would of course like T_p to be as small as possible, therefore we subtract all the optimal β^* values by π. We can do this since $\psi_p(\vec{\gamma},\vec{\beta})$ is π-periodic in β up to a global phase. This π-shifted value of β is the value that one would obtain, if one would choose to use the quantum annealing mixer Hamiltonian (i.e. the one with a minus in front of the summation), instead of the mixer commonly used for the QAOA.

We run the QA algorithm for all the 8 and 15 route instances for different total annealing times T and record the optimal TTS that we find. In Fig. 9 we plot the $\mathrm{TTS}^{\mathrm{OPT}}$ for both algorithms, and find that the $\mathrm{TTS}^{\mathrm{OPT}}_{\mathrm{QAOA}}$ is smaller than $\mathrm{TTS}^{\mathrm{OPT}}_{\mathrm{QA}}$ for all the instances. For the 15 route instances, QAOA is one order of magnitude faster in achieving 99 % success probability.

Appendix D: Depolarizing noise

In this Appendix we perform a simple study of how depolarizing noise affects the performance of QAOA. We model the depolarizing noise as random uncorrelated Pauli-X, Y or Z operations using the error gate

$$\mathcal{E} = (1-\eta)I + \frac{\eta}{3}(X+Y+Z), \tag{D1}$$

where η is the probability that an error occurs, that we fix to 1 %. This error gate acts on each individual qubit between the applications of the cost and mixing Hamiltonian, see Fig. 10(a). We then repeat the circuit sufficiently many times to get a statistical average over the noise. In Fig. 10(b) we plot the success probability with noise for the same 8 and 15 route instances as shown in Fig. 4(b). A trade-off appears between the level of iteration of the algorithm p, and the success probability. In particular, we observe that for $p > 6$ the success probability starts to decrease for the 8 route instance, while for the 15 route instance it levels off, indicating that the gain of increasing one level p equals the decrease due to the noise. This is expected, as faulty gates decrease the fidelity of the prepared state with the best theoretically found variational state. However, the resulting success probabilities at $p = 6$ are roughly halved with respect to the noiseless case.

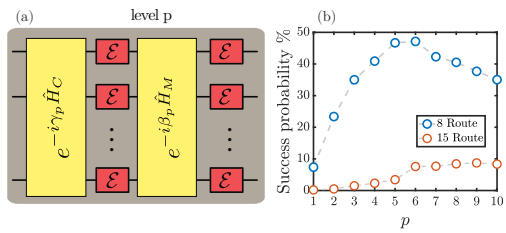


FIG. 10. (a) After each application of the cost and mixing Hamiltonian of the QAOA an error gate $\mathcal{E}$ given by Eq. (D1) is independently applied to every qubit. (b) Success probability with noise for one of the 8 and 15 route instances.

PAPER D

Improved success probability with greater circuit depth for the quantum approximate optimization algorithm

Andreas Bengtsson, Pontus Vikstål, Christopher Warren, **Marika Svensson**, Xiu Gu, Anton Frisk Kockum, Philip Krantz, Christian Križan, Daryoush Shiri, Ida-Maria Svensson, Giovanna Tancredi, Göran Johansson, Per Delsing, Giulia Ferrini, and Jonas Bylander

Phys. Rev. Applied 14, 034010 (2020)

Improved success probability with greater circuit depth for the quantum approximate optimization algorithm

Andreas Bengtsson,[1] Pontus Vikstål,[1] Christopher Warren,[1] Marika Svensson,[2,3] Xiu Gu,[1] Anton Frisk Kockum,[1] Philip Krantz,[1] Christian Križan,[1] Daryoush Shiri,[1] Ida-Maria Svensson,[1] Giovanna Tancredi,[1] Göran Johansson,[1] Per Delsing,[1] Giulia Ferrini,[1] and Jonas Bylander[1,*]

[1] *Microtechnology and Nanoscience, Chalmers University of Technology, SE-412 96, Göteborg, Sweden*
[2] *Computer Science and Engineering, Chalmers University of Technology, SE-412 96, Göteborg, Sweden*
[3] *Jeppesen, SE-411 03, Göteborg, Sweden*

(Dated: August 18, 2020)

Present-day, noisy, small or intermediate-scale quantum processors—although far from fault-tolerant—support the execution of heuristic quantum algorithms, which might enable a quantum advantage, for example, when applied to combinatorial optimization problems. On small-scale quantum processors, validations of such algorithms serve as important technology demonstrators. We implement the quantum approximate optimization algorithm (QAOA) on our hardware platform, consisting of two superconducting transmon qubits and one parametrically modulated coupler. We solve small instances of the NP-complete exact-cover problem, with 96.6% success probability, by iterating the algorithm up to level two.

I. INTRODUCTION

Quantum computing promises exponential computational speedup in a number of fields, such as cryptography, quantum simulation, and linear algebra [1]. Even though a large, fault-tolerant quantum computer is still many years away, impressive progress has been made over the last decade using superconducting circuits [2–4], leading to the noisy intermediate-scale quantum (NISQ) era [5]. It was predicted that NISQ devices should allow for "quantum supremacy" [6], that is, solving a problem that is intractable on a classical computer in a reasonable time. This was recently demonstrated on a 53-qubit processor by sampling the output distributions of random circuits [7].

Two of the most prominent NISQ algorithms are the quantum approximate optimization algorithm (QAOA), for combinatorial optimization problems [8–10], and the variational quantum eigensolver (VQE), for the calculation of molecular energies [11–13]. QAOA is a heuristic algorithm that could bring a polynomial speedup to the solution of specific problems encoded in a quantum Hamiltonian [14, 15]. Moreover, QAOA should produce output distributions that cannot be efficiently calculated on a classical computer [16].

QAOA is a hybrid algorithm, as it is executed on both a classical and a quantum computer. The quantum part consists of a circuit with p levels, where better approximations to the solution of the encoded problem are generally achieved with higher p. In this work, we report on using our superconducting quantum processor to demonstrate QAOA with up to $p = 2$, enabled by adequately high gate fidelities. We solve small toy instances of the NP-complete exact-cover problem with 96.6% success probability. For $p > 1$, the QAOA solution cannot be efficiently calculated on a classical computer, as the computational complexity scales doubly exponentially in p [8].

Our interest in solving the exact-cover problem originates from its use in many real-world applications, for instance, the exact-cover problem can provide feasible solutions to airline planning problems such as tail assignment [17]. Currently, this is solved by well-developed optimization techniques in combination with heuristics. By leveraging heuristic quantum algorithms such as QAOA, the current approach can be augmented and might provide high-quality solutions while reducing the running time. Applying QAOA to instances of the exact-cover problem extracted from real-world data in the context of the tail assignment has been numerically studied with 25 qubits, corresponding to 25 routes and 278 flights [18]. Other quantum algorithms for solving the exact-cover problem, specifically quantum annealing, have been considered in Refs. [19–21].

II. QAOA

All NP-complete problems can be formulated in terms of finding the ground state of an Ising Hamiltonian [22]. QAOA aims at finding this state by applying two non-commuting Hamiltonians, $\hat{B}$ and $\hat{C}$, in an alternating sequence (with length p) to an equal superposition state of n qubits [visualized in Fig. 1(a)],

$$|\vec{\gamma},\vec{\beta}\rangle = \prod_{i=1}^{p}\left[e^{-i\beta_i\hat{B}}e^{-i\gamma_i\hat{C}}\right]\left(\frac{|0\rangle+|1\rangle}{2}\right)^{\otimes n}, \quad (1)$$

where γ_i and β_i are (real) variational angles. The first Hamiltonian in the sequence is the Ising (cost) Hamiltonian specifying the problem,

$$\hat{C} = \sum_{i=1}^{n} h_i\hat{\sigma}_i^z + \sum_{i<j} J_{ij}\hat{\sigma}_i^z\hat{\sigma}_j^z, \quad (2)$$

* bylander@chalmers.se

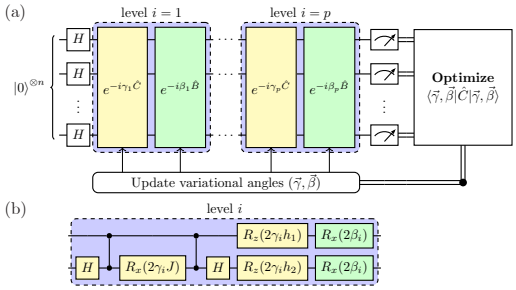


FIG. 1. (a) The quantum approximate optimization algorithm (QAOA) for a problem specified by the Ising Hamiltonian $\hat{C}$. An alternating sequence of two Hamiltonians ($\hat{C}$ and $\hat{B}$) is applied to an equal superposition of n qubits. After measurement of the qubit states, a cost is calculated, which a classical optimization algorithm minimizes by varying the angles $\vec{\gamma},\vec{\beta}$. (b) Our implementation of one QAOA level with $n = 2$ using controlled-phase and single-qubit gates. The background color of each gate identifies which part in (a) it implements.

and the second is a transverse field (mixing) Hamiltonian defined by

$$\hat{B} = \sum_{i=1}^{n} \hat{\sigma}_i^x, \tag{3}$$

where h_i and J_{ij} are real coefficients, and $\hat{\sigma}_i^{x(z)}$ are the Pauli X (Z) operators applied to the i^{th} qubit.

The ground state of Eq. (2) corresponds to the lowest-energy state. We therefore define the energy expectation value of Eq. (1) as a cost function

$$F(\vec{\gamma},\vec{\beta}) = \langle\vec{\gamma},\vec{\beta}|\hat{C}|\vec{\gamma},\vec{\beta}\rangle = \sum_{i=1}^{n} h_i\langle\hat{\sigma}_i^z\rangle + \sum_{i<j} J_{ij}\langle\hat{\sigma}_i^z\hat{\sigma}_j^z\rangle. \tag{4}$$

This cost function is evaluated by repeatedly preparing and measuring $|\vec{\gamma},\vec{\beta}\rangle$ on a quantum processor. To find the state that minimizes Eq. (4), a classical optimizer is used to find the optimal variational angles $\vec{\gamma}^*,\vec{\beta}^*$. For a high enough p, $|\vec{\gamma}^*,\vec{\beta}^*\rangle$ is equal to the ground state of $\hat{C}$ and hence yields the answer to the optimization problem [8]. However, for algorithms executed on real hardware without error correction, noise will inevitably limit the circuit depth, implying that there is a trade-off between algorithmic errors (too low p) and gate errors (too high p). Note that, in order to find the solution to the optimization problem, it is not necessary for $|\vec{\gamma}^*,\vec{\beta}^*\rangle$ to be equal to the ground state: as long as the ground-state probability is high enough, the quantum processor can be used to generate a shortlist of potential solutions that can be checked efficiently (in polynomial time) on a classical computer. For instance, even if the success probability of measuring the ground state is only 5%, we could measure 100 instances and still attain a probability greater than 99% of finding the correct state. Moreover, the angles $\vec{\gamma}^*,\vec{\beta}^*$ themselves are not interesting, as long as they yield the lowest-energy state. This gives some robustness against coherent gate errors, since any over- or under-rotations can be compensated for by a change in the variational angles [12].

We apply QAOA to the exact-cover problem, which reads: given a set X and several subsets S_i containing parts of X, which combination of subsets include all elements of X just once? Mathematically speaking, this combination of subsets should be disjoint, and their union should be X. This problem can be mapped onto an Ising Hamiltonian, where the number of spins equals the number of subsets, while the size of X can be arbitrary.

Let us consider $n = 2$, for which the two-spin Ising Hamiltonian is

$$\hat{C} = h_1\hat{\sigma}_1^z + h_2\hat{\sigma}_2^z + J\hat{\sigma}_1^z\hat{\sigma}_2^z. \tag{5}$$

The exact-cover problem is mapped onto this Hamiltonian by choosing h_i and J as follows [23]:

$$\begin{aligned} J &> \min(c_1, c_2), \\ h_1 &= J - 2c_1, \\ h_2 &= J - 2c_2, \end{aligned} \tag{6}$$

where c_i is the number of elements in subset i, and $J > 0$ if the two subsets share at least one element. We are free to choose J, as long as it fulfills the criterion in Eq. (6). For example, consider $X = \{x_1, x_2\}$ and two subsets $S_1 = \{x_1, x_2\}$ and $S_2 = \{x_1\}$. This gives $c_1 = 2$ and $c_2 = 1$, and we could choose $J = 2$, yielding $h_1 = -2$ and $h_2 = 0$. It is easy to check that the corresponding ground state is $|10\rangle$ (i.e., S_1 is the solution). Finally, we normalize J and h_i such that the Ising Hamiltonian has integer eigenvalues, allowing us to restrict γ_i and β_i to the interval $[0, \pi[$.

For two subsets, four different problems exist, which all yield different sets of h_i and J. These are summarized in Table I. Problem A is the example given above; it is the most interesting, as the other problems are trivial. Problems B and C are trivial since they do not contain any qubit-qubit interaction ($J = 0$). Problem D is also trivial since both subsets are equal. Additionally, the ground states are degenerate for problems B and D.

III. REALIZATION ON QUANTUM HARDWARE

We implement Eq. (1) on our quantum processor using the circuit in Fig. 1(b). The circuit can be somewhat compiled by simple identities (e.g., two Hadamard gates equal identity). We stress that our implementation of QAOA is scalable in that we do not use any exponentially costly pre-compilation (e.g., calculating the final circuit unitary and using Cartan decomposition to minimize the number of two-qubit gates).

TABLE I. The four different exact-cover problems available with two subsets, and their solutions and respective sets of coefficients in the Ising Hamiltonian $\hat{C} = h_1\hat{\sigma}_1^z + h_2\hat{\sigma}_1^z + J\hat{\sigma}_1^z\hat{\sigma}_2^z$.

#	Subsets	h_1	h_2	J	Solution
A	$S_1 = \{x_1, x_2\}$, $S_2 = \{x_1\}$	-1/2	0	1/2	$\|10\rangle$
B	$S_1 = \{x_1, x_2\}$, $S_2 = \{\}$	-1	0	0	$\|10\rangle$ or $\|11\rangle$
C	$S_1 = \{x_1\}$, $S_2 = \{x_2\}$	-1/2	-1/2	0	$\|11\rangle$
D	$S_1 = \{x_1, x_2\}$, $S_2 = \{x_1, x_2\}$	0	0	1	$\|10\rangle$ or $\|01\rangle$

Our quantum processor is fabricated using the same processes as in Ref. [24] and consists of two fixed-frequency transmon qubits with individual control and readout. Both qubits are coupled to a common frequency-tunable coupler used to mediate a controlled-phase gate (CZ) between the qubits. The CZ gate is realized by a full coherent oscillation between the $|11\rangle$ and $|02\rangle$ states. The interaction is achieved by parametrically modulating the resonant frequency of the coupler at a frequency close to the difference frequency between the $|0\rangle - |1\rangle$ and $|1\rangle - |2\rangle$ transitions of qubit 1 and 2, respectively [25, 26]. We have benchmarked such a gate on the same device during the same cooldown to above 99%; however, the benchmark performed closest in time to the experiments presented here showed a fidelity of 98.6%. These kinds of fidelity fluctuations might be related to fluctuations in the qubits' coherence times [24]. Single-qubit X rotations are driven by microwave pulses at the qubit transition frequencies with fidelities of 99.86% and 99.93% for the respective qubits. Z-rotations are implemented in software as a shift in drive phase and thus have unity fidelity [27]. All the reported gate fidelities were measured by (interleaved) randomized benchmarking [28]. More experimental details, a measurement setup along with a device schematic, and benchmarking results are found in Appendix A and Appendix B.

IV. APPLYING QAOA TO FOUR PROBLEMS

For $p = 1$, we apply a simple grid (61×61) search of $\beta_1, \gamma_1 \in [0, \pi[$ while recording 5000 measurements of each qubit. From these, we calculate $\langle\sigma_i^z\rangle$, $\langle\sigma_1^z\sigma_2^z\rangle$, the cost function F, and the occupation probability for each of the four possible states, while accounting for the limited, but calibrated, readout fidelity (86 % and 95 % for the two qubits). By collecting sufficiently many samples, the statistical error on the estimated quantities can be made small.

The grid search allows us to explore the shape of the optimization landscape, which may bring important understanding in the difficulty of finding global minima for black-box optimizers. In Fig. 2, we show measured

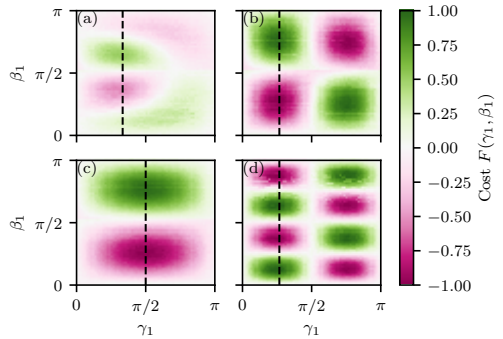


FIG. 2. Cost functions $F(\vec{\gamma}, \vec{\beta})$ for QAOA applied to four instances of the exact-cover problem with $p = 1$ and $n = 2$. (a-d) correspond to problems A-D in Table I. Each experimental data point is evaluated from the average of 5000 measurements on our quantum processor. The dashed lines indicate the positions of the linecuts in Fig. 3.

cost functions for the four problems in Table I. Due to the normalization of h_i and J, the ground state for each problem corresponds to $F = -1$. In Fig. 2(a), the cost function for problem A never reaches below -0.5. To achieve costs approaching -1, additional levels ($p > 1$) are needed. Moreover, the existence of a local minimum around $\gamma_1 \approx \beta_1 \approx 3\pi/4$ could cause difficulties for optimizers trying to find the global minimum. For problems B-D [Fig. 2(b-d)], we see clear minima where $F \approx -1$, indicating that we have found the optimal variational angles $|\vec{\gamma}^*, \vec{\beta}^*\rangle$ corresponding to the ground state.

In Fig. 3, we take linecuts along the dashed lines in Fig. 2 and benchmark our measured cost functions and state probabilities against those of an ideal quantum computer without any noise. We see excellent agreement between measurement and theory: the measured positions of each minimum and maximum are aligned with those of the theory, consistent with low coherent-error rates. In addition, we observe excellent agreement between the absolute values at the minima and maxima, indicating low incoherent-error rates as well. Even with high gate fidelities, a high algorithmic fidelity is not guaranteed. Randomized benchmarking gives the average fidelity over a large number of random gates, which transforms any coherent errors into incoherent ones. For real quantum algorithm circuits, the gates are generally not random. Therefore, any coherent errors can quickly add up and yield algorithmic performance far lower than expected from randomized benchmarking fidelities alone [29, 30].

To quantify the performance of QAOA with $p = 1$, we compare the highest-probability state at the minima of F with the solutions in Table I. Problem A [Fig. 3(a)] does not reach its ground state ($F \approx -0.5$); however, the probability of measuring the correct state ($|10\rangle$) is

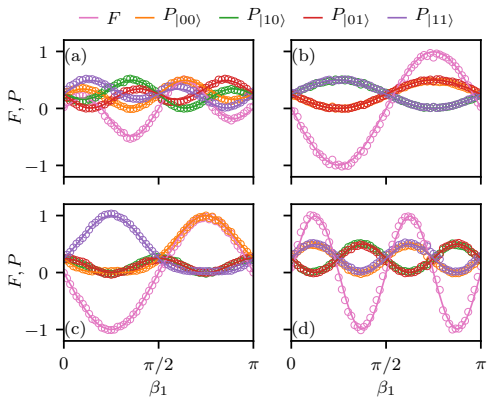


FIG. 3. A comparison between experiment (open circles) and theory (solid lines) for four exact-cover problems using QAOA with $p = 1$. Each color (given at the top) corresponds to either a state probability or the value of the cost function F. The four panels (a-d) correspond to the four problems (A-D) in Table I. The linecuts are taken at the vertical dashed lines in Fig. 2. The theory curves are calculated assuming an ideal quantum processor, whereas each experimental data point is derived from the average of 5000 measurements on our quantum processor.

approximately 50%, which is still better than random guessing. For problem C [Fig. 3(c)], we see that $F \approx -1$ does indeed correspond to a probability close to unity of measuring the ground state ($|11\rangle$). Problems B and D [Fig. 3(b) and (d)] have degenerate ground states, indicated by two state probabilities close to 50% each at $F \approx -1$.

V. INCREASING THE SUCCESS PROBABILITY

To increase the success probability for problem A, we add an additional level ($p = 2$). For $p > 1$, a grid search to map out the full landscape becomes unfeasible due to the many parameters (equal to $2p$). Therefore, we instead use black-box optimizers to find the optimal variational angles. We try three different gradient-free optimizers: Bayesian optimization with Gaussian processes (BGP), Nelder-Mead, and covariance matrix adaptation evolution strategy (CMA-ES). We choose BGP due to its ability to find global minima, Nelder-Mead due to it being common and simple, and CMA-ES due to its favorable scaling with the number of optimization parameters.

We evaluate the optimizer performances by running 200 independent optimizations with random starting values $(\vec{\gamma}, \vec{\beta} \in [0, \pi[)$ for each optimizer. For each set of variational angles, we repeat the circuit and measure 5000 samples to accurately estimate the expectation values. We set a threshold for convergence at $F < -0.95$ and count the number of converged optimization runs as well as the number of calls to the quantum processor (function calls) required to converge. We also record the success probability of measuring the problem solution ($P_{|10\rangle}$). The results are summarized in Table II.

TABLE II. Comparison between different optimizers. We run QAOA for problem A over 200 iterations with random starting parameters. We extract the convergence probability for reaching a cost below -0.95, the average number of function calls required to reach that level, and the highest achieved probability of measuring the problem solution ($P_{|10\rangle}$).

Optimizer	Convergence	Function calls	$P_{\|10\rangle}$
BGP	61.5%	44 ± 16	96.5%
Nelder-Mead	20.0%	38 ± 13	96.3%
CMA-ES	49.5%	121 ± 46	96.6%

We observe that the success probabilities after convergence are similar for all three optimizers. However, there is a difference in convergence probability, of which BGP has the highest and Nelder-Mead has the lowest. The lower performance of Nelder-Mead is most likely due to its sensitivity to local minima, a well-known problem for most optimizers. In contrast, one of the strengths of Bayesian optimization is its ability to find global minima, which could explain why it performs better than Nelder-Mead and CMA-ES. Additionally, Bayesian optimization is designed to handle optimization where the time of each function call is high (costly), such that the number of call is kept low. However, for more optimization parameters (higher p), the performance of BGP is generally decreased due to an increasing need for classical computation. CMA-ES, on the other hand, excels when the number of parameters are high, and thus might be a good optimizer for QAOA with tens or hundreds of parameters. Here, with just four parameters, CMA-ES has a convergence-probability similar to that of BGP, although with a greater number of function calls on average.

To quantify the optimization further, we study the trajectories of each optimization run (Fig. 4). For each run, we plot the costs F. The trajectories for BGP and Nelder-Mead [Fig. 4(a-b)] corroborate the indications about local minima. We see groups of horizontal lines corresponding to different local minima, especially clear at $F \approx -0.55$ for both BGP and Nelder-Mead. We also see that BGP tries, and sometimes succeeds, to escape these local minima, which is one of the advantages of Bayesian optimization. In comparison, Nelder-Mead rarely gets out of a local minimum once it has found it. For the third optimizer, CMA-ES [Fig. 4(c)], it is hard to draw any conclusions from the trajectories other than that the convergence is slower than for the other optimizers. However, we include the CMA-ES trajectories for completeness. For each optimizer, we also plot the averaged (over all the converged) trajectories for F and the probability of finding

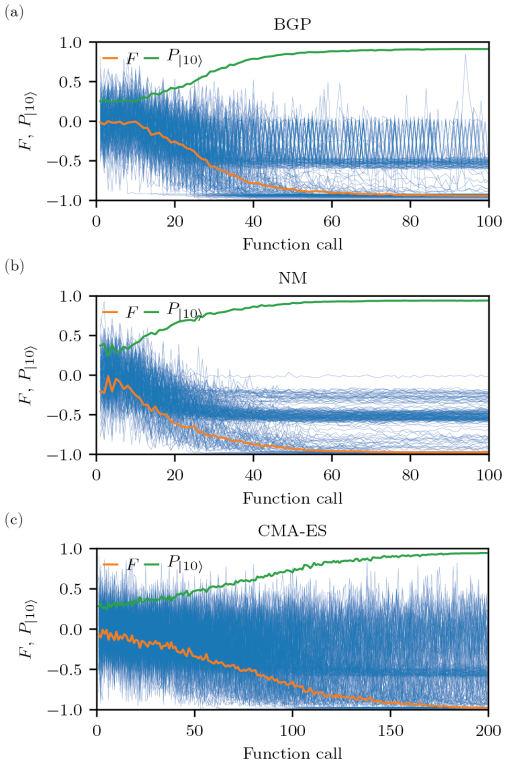


FIG. 4. Optimization of variational angles for problem A using $p = 2$ iterations of the algorithm and three different black-box and gradient-free optimizers: (a) Bayesian optimization with Gaussian processes, (b) Nelder-Mead, and (c) covariance matrix adaptation evolution strategy. We run the optimization 200 times with random starting parameters. Plotted as blue lines are the individual optimization trajectories for F, where each data-point is the average of 5000 measurements. In orange and green are the cost (F) and success probability ($P_{|10\rangle}$) averaged over the converged runs.

the solution state $P_{|10\rangle}$.

At the end of the optimization, the highest recorded probability of generating the correct state is 96.6%. The success probability is limited by imperfect gates (we have verified that an ideal quantum computer and $p = 2$ can achieve $P_{|10\rangle} = 1$). We compare our measured success probability to what we would expect from the randomized-benchmarking fidelities. The quantum circuit for $p = 2$ consists of 6 X, 4 Hadamard, 4 Z, and 3 CZ gates, which, when multiplied together with the fidelities for each gate, predicts a total fidelity of 96.3%, in good agreement with the measured fidelity considering experimental uncertainties (e.g., fluctuations in qubit coherence and gate fidelities). Note that $p = 3$ would not yield a higher success

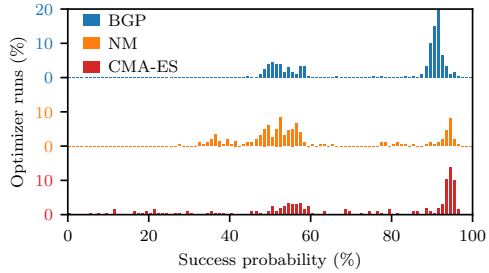


FIG. 5. Histogram of the final success probabilities ($P_{|10\rangle}$) for three different optimizers on problem A using $p = 2$ iterations of the algorithm. Each optimizer was run 200 times. The bars are vertically offset for clarity.

probability, since adding more gates would lower the total fidelity further (predicted to be 94.2%).

Finally, we examine histograms over the success probabilities at the end of each optimization run for the three different optimizers, see Fig. 5. Again, we observe that BGP has the most converged runs out of the three. We see clusters around 55 and 95 % success probability for all three optimizers, possibly corresponding to one local and the global minima. For CMA-ES the success probabilities are more scattered, where some runs even have below 40 % success. All in all, Bayesian optimization performs the best; however, further studies will be needed on which classical optimizer is the most suitable for variational quantum algorithms, such as QAOA and VQE.

VI. CONCLUSION

In conclusion, we have implemented the quantum approximate optimization algorithm with up to $p = 2$ levels. Using a superconducting quantum processor with state-of-the-art performance, we successfully optimized four instances of the exact-cover problem. For the non-trivial instance (problem A), we used $p = 2$ and black-box optimization to reach a success probability to 96.6% (up from 50% with $p = 1$), in good agreement with a prediction from our gate fidelities. Even if many more qubits are needed to solve problems that are intractable for classical computers, algorithmic performance serves as a critical quantum-processor benchmark since performance can be much lower than what individual gate fidelities predicts. Although further experiments with larger devices are needed to explore if QAOA can have an advantage over classical algorithms, our results show that QAOA can be used to solve the exact-cover problem.

ACKNOWLEDGMENTS

We are grateful to the Quantum Device Lab at ETH Zürich for sharing their designs of sample holder and printed circuit board, and Simon Gustavsson and Bruno Küng for valuable support with the measurement infrastructure. We also thank Morten Kjaergaard, Devdatt Dubhashi, and Kevin Pack for insightful discussions. This work was performed in part at Myfab Chalmers. We acknowledge financial support from the Knut and Alice Wallenberg Foundation, the Swedish Research Council, and the EU Flagship on Quantum Technology H2020-FETFLAG-2018-03 project 820363 OpenSuperQ.

Appendix A: Measurement setup

The experimental measurement setup used here is a standard circuit-QED setup, see schematic in Fig. 6. The quantum processor consists of two xmon-style transmon qubits coupled via a frequency-tunable anharmonic oscillator. The tunability is provided by two Josephson junctions in a SQUID configuration. The two qubits are capacitively coupled to individual control lines and quarter-wavelength resonators for readout. There is also a readout resonator for the coupler, which is only used as a debugging tool (i.e., it is not involved during any algorithm execution). The SQUID for the tunable coupler is inductively coupled to a waveguide to allow for both static and fast modulation of the resonant frequency.

The processor is fabricated on a high-resistivity intrinsic silicon substrate. After initial chemical cleaning, an aluminium film is evaporated. All features except the Josephson junctions are patterned by direct-write laser lithography and etched with a warm mixture of acids. The Josephson junctions are patterned by electron-beam lithography, and evaporated from the same target as previously. A third lithography and evaporation step (with in-situ ion milling) is performed to connect the Josephson junctions to the rest of the circuit. Finally, the wafer is diced into individual dies and subsequently cleaned by a combination of wet and dry chemistry.

Then, a die is selected and packaged in a copper box and wire bonded to a palladium- and gold-plated printed circuit board with 16 non-magnetic coaxial connectors. For the present device, we use 5 of these connectors, 2 for readout, 2 for single-qubit control, and 1 for control of the magnetic flux through the SQUID loop of the coupler. These are connected to filtered and attenuated coaxial lines leading up to room temperature. We point out that the DC current for the static flux bias is also provided through the coaxial line. Finally, the processor is attached to the mixing chamber of a Bluefors LD250 cryo-free dilution refrigerator. There, it is shielded from stray magnetic fields by two shields of cryoperm/mu-metal and two shields of superconductors.

We perform multiplexed readout by using a Zurich Instruments UHFQA for generating and detecting the read-

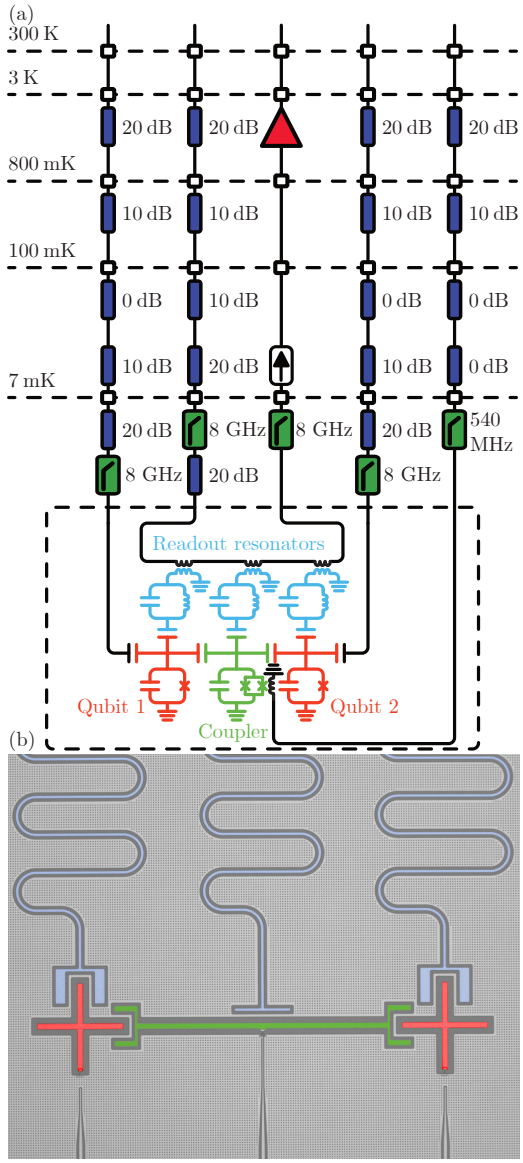


FIG. 6. (a) Cryogenic setup and electrical circuit of the quantum processor. All lines are attenuated and filtered to minimize the amount of noise reaching the qubits. The readout output contains cryogenic isolators and a high electron mobility transistor amplifier. (b) False-colored micro-graph of the processor. The colors match the circuit elements in (a). The three waveguides at the bottom are for control over the qubits and the coupler.

TABLE III. Device parameters. Readout-resonator frequency f_R and qubit transition frequencies f_{ij}. g is the coupling between qubit and resonator, and j is the coupling between qubit and coupler. T_1 and T_2^* are the relaxation and free induction decay times measured over 14 hours. F_{1q}, F_m, and F_{CZ} are the single-qubit, measurement, and CZ fidelities, respectively.

Parameter	Qubit 1	Qubit 2
f_R	6.17 GHz	6.04 GHz
f_{01}	3.82 GHz	4.30 GHz
$f_{12} - f_{01}$	-229 MHz	-225 MHz
j	29.1 MHz	33.0 MHz
g	53.2 MHz	56.9 MHz
T_1	77 µs	55 µs
T_2^*	49 µs	82 µs
F_{1q}	0.9986	0.9993
F_m	0.86	0.95
F_{CZ}	0.986	

out signals, together with a Rohde & Schwarz SGS100A continuous-wave signal generator and two Marki IQ mixers for up- and down-conversion. The single-qubit pulses are synthesized by a Zurich Instruments HDAWG and upconverted using Rohde & Schwarz SGS100A vector signal generators. The flux drive is generated directly by the HDAWG since the modulation frequency is within the bandwidth of the instrument. Finally, all instruments are controlled and orchestrated by the measurement and automation software Labber. Labber also does cost-function evaluations and calls external Python packages for the three different optimizers. All three optimizers were run using publicly available packages: Scikit-Optimize for BGP, scipy for Nelder-Mead, and pycma for CMA-ES.

Appendix B: Characterization and tune-up

Initially, we perform basic spectroscopy and decoherence benchmarking of each qubit individually. This allows us to extract readout frequencies, qubit frequencies and anharmonicities, relaxation and dephasing times, and static couplings between qubit and resonator, as well as between qubit and coupler. The extracted parameters are found in Table III.

After the initial characterization, we tune up high-fidelity single-qubit gates. The drive pulses have cosine envelopes together with first-order DRAG components to compensate for the qubit frequency shift due to the driving. Our rather long (50 ns) pulses makes leakage from $|1\rangle$ to $|2\rangle$ minimal even without DRAG. To find optimal pulse amplitudes and DRAG coefficients, we use error amplification by applying varying lengths of trains of π pulses. Qubit drive frequencies are measured accurately by detuned Ramsey fringes.

Next, we calibrate our readout fidelities. By collecting raw voltages of the readout signals (as measured by the digitizer in the UHFQA), with and without a calibrated pi-pulse applied to the qubit ($|0\rangle$ and $|1\rangle$ states, respectively), and as a function of readout frequency and amplitude, we can find the optimal readout parameters. Due to our rather low coupling strengths, we cannot achieve short readout times in this device. However, QAOA does not require any measurement feedback, so a long readout time is not an issue as long as the time is shorter than the relaxation times of the qubits. Also, longer readout times give greater signal-to-noise ratios, which allows us to achieve high readout fidelities even in the absence of a quantum-limited amplifier. Here, the readout is 2.3 µs long, well below our relaxation times (several tens of microseconds). After finding the optimal readout parameters, a voltage threshold is used to differentiate between $|0\rangle$ or $|1\rangle$ of the measured qubit.

To accurately extract state probabilities in the presence of limited readout fidelity, we collect statistics of the measured qubit population as a function of qubit drive amplitude (Rabi oscillations). Since the measured population increases monotonically with the expected population, we can renormalize the populations, similarly to Ref. [31]. This calibration allows us to accurately measure the average quantities $\langle\sigma_i^z\rangle$, $\langle\sigma_i^1\sigma_i^2\rangle$ and state probabilities even in the presence of limited readout fidelities.

Our two-qubit gate of choice is the controlled phase (CZ). This interaction is induced by parametrically modulating the resonant frequency of the coupler at a frequency close to the difference of $|11\rangle$ and $|02\rangle$. For our device, this frequency is 255 MHz. However, due to the frequency modulation and the non-linear relationship between flux and frequency, the transition frequencies are slightly lowered. This frequency shift will also induce deterministic phase shifts on the individual qubits, which we compensate for by applying Z gates on both qubits after each CZ gate. We choose a static bias point and a modulation amplitude that yield a moderate effective coupling strength of 5 MHz between the two states. From here, we find the modulation frequency and time that yield a full oscillation between the $|11\rangle$ and $|02\rangle$ states. We then fine-tune the frequency and time such that the controlled phase is π and the leakage to $|02\rangle$ is minimal. Here, the final gate frequency and duration were 253 MHz and 271 ns.

We benchmark our single and two-qubit fidelities using randomized benchmarking. A sequence of random gates drawn from the Clifford group is applied together with a final recovery gate which should take the system back to the ground state. The number of random gates is varied and the probability of measuring the ground state is recorded. In Fig. 7, we plot these probabilities for each qubit individually and for the two-qubit case. In the single-qubit case, it is important to note that it was done simultaneously for both qubits. Generally, the gate fidelities are higher if they are done in isolation. However, to reduce the total run time of algorithms, we usually run single-qubit gates in parallel. Therefore, simultaneous randomized benchmarking fidelities are more relevant metrics than isolated ones.

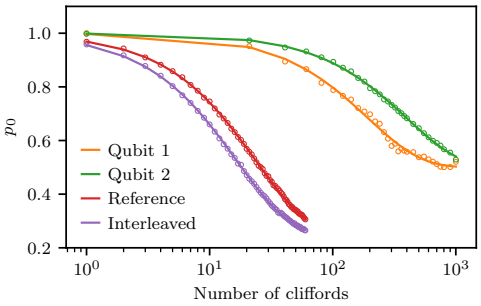


FIG. 7. Randomized benchmarking of single- and two-qubit gates. Plotted are the probability of measuring the ground state as a function of number of Clifford gates applied. Circles are data, and lines are fits to extract the gate fidelity. For qubit 1 and 2, the extracted single-qubit fidelities (averaged over all possible single-qubit Cliffords) are 0.9986 and 0.9993. For benchmarking of the two-qubit gate, we take a reference (random Clifford gates) and an interleaved (a CZ gate between each Clifford) trace to extract the CZ fidelity (0.986).

[1] A. Montanaro, Quantum algorithms: an overview, npj Quantum Information **2**, 15023 (2016).

[2] G. Wendin, Quantum information processing with superconducting circuits: a review, Reports on Progress in Physics **80**, 106001 (2017).

[3] X. Gu, A. F. Kockum, A. Miranowicz, Y. X. Liu, and F. Nori, Microwave photonics with superconducting quantum circuits, Physics Reports **718-719**, 1 (2017).

[4] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, Superconducting qubits: Current state of play, Annual Review of Condensed Matter Physics **11**, 369 (2020).

[5] J. Preskill, Quantum computing in the nisq era and beyond, Quantum **2**, 79 (2018).

[6] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Bremner, J. M. Martinis, and H. Neven, Characterizing quantum supremacy in near-term devices, Nature Physics **14**, 595 (2018).

[7] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, *et al.*, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[8] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv preprint arXiv:1411.4028 (2014).

[9] J. Otterbach, R. Manenti, N. Alidoust, A. Bestwick, M. Block, B. Bloom, S. Caldwell, N. Didier, E. S. Fried, S. Hong, *et al.*, Unsupervised machine learning on a hybrid quantum computer, arXiv preprint arXiv:1712.05771 (2017).

[10] G. Pagano, A. Bapat, P. Becker, K. Collins, A. De, P. Hess, H. Kaplan, A. Kyprianidis, W. Tan, C. Baldwin, *et al.*, Quantum approximate optimization with a trapped-ion quantum simulator, arXiv preprint arXiv:1906.02700 (2019).

[11] 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 Communications **5**, 4213 (2014).

[12] P. J. J. O'Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding, *et al.*, Scalable quantum simulation of molecular energies, Physical Review X **6**, 031007 (2016).

[13] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature **549**, 242 (2017).

[14] Z. Jiang, E. G. Rieffel, and Z. Wang, Near-optimal quantum circuit for grover's unstructured search using a transverse field, Physical Review A **95**, 062317 (2017).

[15] M. Y. Niu, S. Lu, and I. L. Chuang, Optimizing qaoa: Success probability and runtime dependence on circuit depth, arXiv preprint arXiv:1905.12134 (2019).

[16] E. Farhi and A. W. Harrow, Quantum supremacy through the quantum approximate optimization algorithm, arXiv preprint arXiv:1602.07674 (2016).

[17] M. Grönkvist, *The Tail Assignment Problem*, Ph.D. thesis, Chalmers University of Technology and Göteborg University (2005).

[18] P. Vikstål, M. Grönkvist, M. Svensson, M. Andersson, G. Johansson, and G. Ferrini, Applying the quantum approximate optimization algorithm to the tail assignment problem, arXiv preprint arXiv:1912.10499 (2019).

[19] 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 **292**, 472 (2001).
[20] H. Wang and L.-A. Wu, Ultrafast adiabatic quantum algorithm for the np-complete exact cover problem, Scientific reports **6**, 22307 (2016).
[21] T. Graß, Quantum annealing with longitudinal bias fields, Physical Review Letters **123**, 120501 (2019).
[22] A. Lucas, Ising formulations of many np problems, Frontiers in Physics **2**, 5 (2014).
[23] V. Choi, Adiabatic quantum algorithms for the np-complete maximum-weight independent set, exact cover and 3sat problems, arXiv preprint arXiv:1004.2226 (2010).
[24] J. J. Burnett, A. Bengtsson, M. Scigliuzzo, D. Niepce, M. Kudra, P. Delsing, and J. Bylander, Decoherence benchmarking of superconducting qubits, npj Quantum Information **5**, 9 (2019).
[25] D. C. McKay, S. Filipp, A. Mezzacapo, E. Magesan, J. M. Chow, and J. M. Gambetta, Universal gate for fixed-frequency qubits via a tunable bus, Physical Review Applied **6**, 064007 (2016).
[26] S. Caldwell, N. Didier, C. Ryan, E. Sete, A. Hudson, P. Karalekas, R. Manenti, M. da Silva, R. Sinclair, E. Acala, *et al.*, Parametrically activated entangling gates using transmon qubits, Physical Review Applied **10**, 034050 (2018).
[27] D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, and J. M. Gambetta, Efficient z gates for quantum computing, Physical Review A **96**, 022330 (2017).
[28] A. D. Córcoles, J. M. Gambetta, J. M. Chow, J. A. Smolin, M. Ware, J. Strand, B. L. T. Plourde, and M. Steffen, Process verification of two-qubit quantum gates by randomized benchmarking, Physical Review A **87**, 030301(R) (2013).
[29] K. Michielsen, M. Nocon, D. Willsch, F. Jin, T. Lippert, and H. De Raedt, Benchmarking gate-based quantum computers, Computer Physics Communications **220**, 44 (2017).
[30] M. Kjaergaard, M. Schwartz, A. Greene, G. Samach, A. Bengtsson, M. O'Keeffe, C. McNally, J. Braumüller, D. Kim, P. Krantz, *et al.*, A quantum instruction set implemented on a superconducting quantum processor, arXiv preprint arXiv:2001.08838 (2020).
[31] J. M. Chow, L. DiCarlo, J. M. Gambetta, A. Nunnenkamp, L. S. Bishop, L. Frunzio, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf, Detecting highly entangled states with a joint qubit readout, Physical Review A **81**, 062325 (2010).