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 Ferrini1

1Wallenberg Centre for Quantum Technology, Department of Microtechnology and Nanoscience, Chalmers University of Technology, 412 96 Gothenburg, Sweden
2Jeppesen Systems AB, 411 03 Gothenburg, Sweden.

(Dated: December 24, 2019)

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 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 combinatorial optimization problems.

* e-mail: vikstal@chalmers.se

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. 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. [13–18], and that 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 [19].

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 analyse 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 resource types such as passengers, crew, aircraft, maintenance and ground staff. The typical airline planning pro-
cess [20] 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 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

$$\text{minimize } \sum_{r \in R} c_r x_r + \sum_{f \in F} C_f u_f,$$  \hspace{1cm} (1)

subject to

$$\sum_{r \in R} a_{fr} x_r + u_f = 1, \quad \forall f \in F,$$  \hspace{1cm} (2)

$$\sum_{r \in R} b_{tr} x_r + v_t = 1, \quad \forall t \in T,$$  \hspace{1cm} (3)

$$x_r, u_f, v_t \in \{0, 1\}$$  \hspace{1cm} (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 [21].

### 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 (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.

### 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 (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, and is know
to be NP-complete [22]:

\[
\begin{align*}
\text{minimize} & \quad 0 \\
\text{subject to} & \quad \sum_{r \in R} a_{fr}x_r = 1, \quad \forall f \in F, \\
& \quad x_r \in \{0, 1\}.
\end{align*}
\]

(5)

(6)

(7)

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 [23]

\[
\hat{H}_C = \sum_{i<j} J_{ij} \hat{\sigma}_i^x \hat{\sigma}_j^x + \sum_{i=1}^n h_i \hat{\sigma}_i^z.
\]

(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 the r.h.s of Eq. (6) and squaring the expression an energy function formulation is obtained:

\[
\mathcal{E}(s_1, \ldots, s_{|R|}) = \sum_{f=1}^{|F|} \left( \sum_{r=1}^{|R|} a_{fr} x_r - 1 \right)^2.
\]

(9)

Here \(|R|\) and \(|F|\) denotes 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

\[
x_r = \frac{s_r + 1}{2},
\]

(10)

and expanding the square of Eq. (9) we obtain the Ising energy function for the Exact Cover problem

\[
\mathcal{E}(s_1, \ldots, s_{|R|}) = \sum_{f=1}^{|F|} \left( \sum_{r=1}^{|R|} a_{fr} s_r + \frac{1}{2} \right)^2 - 1
\]

\[
+ \frac{1}{4} \sum_{f=1}^{|F|} \sum_{r=1}^{|R|} a_{fr} a_{fr'} s_r s_{r'}
\]

\[
+ \frac{1}{2} \sum_{f=1}^{|F|} \sum_{r=1}^{|R|} a_{fr} s_r \left( \sum_{r'=1}^{|R|} a_{fr'} - 2 \right)
\]

\[
+ \frac{1}{4} \sum_{f=1}^{|F|} \left( \sum_{r=1}^{|R|} a_{fr} - 2 \right)^2.
\]

(11)

By defining \(J_{rr'}\) as

\[
J_{rr'} = \frac{1}{2} \sum_{f=1}^{|F|} a_{fr} a_{fr'},
\]

(12)

and \(h_r\) as

\[
h_r = \frac{1}{2} \sum_{f=1}^{|F|} a_{fr} \left( \sum_{r'=1}^{|R|} a_{fr'} - 2 \right),
\]

(13)

the Ising energy function becomes

\[
\frac{1}{2} \sum_{r=1}^{|R|} \sum_{r'=1}^{|R|} J_{rr'} s_r s_{r'} + \sum_{r=1}^{|R|} h_r s_r + \text{const.}
\]

(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 \(\text{Tr}(J)\) since \(s_i^2 = 1\); because \(J_{ij}\) is symmetric i.e. \(J_{ij} = J_{ij}\), we can further simplify the expression and write the Ising energy function as

\[
\mathcal{E}(s_1, \ldots, s_{|R|}) = \sum_{r<r'} J_{rr'} s_r s_{r'} + \sum_{r=1}^{|R|} h_r s_r + \text{const.}
\]

(15)

where we have absorbed \(\frac{1}{2} \text{Tr}(J)\) into the constant. Finally, by promoting the spin variables to Pauli spin matrices \(s_i \rightarrow \hat{\sigma}_i^x\), 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 \(|+\> \otimes^n\). 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:

\[
\hat{U}(\gamma) \equiv e^{-i\gamma \hat{H}_C}, \quad \hat{V}(\beta) \equiv e^{-i\beta \hat{H}_M},
\]

(16)
where \( \hat{H}_C \) is the cost Hamiltonian given by Eq. (8), and \( \hat{H}_M \equiv \sum_{i=1}^m \hat{\sigma}_i^z \) 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:

\[
|\psi_p(\vec{\gamma}, \vec{\beta})\rangle \equiv \hat{V}(\beta_p)\hat{U}(\gamma_p) \cdots \hat{V}(\beta_1)\hat{U}(\gamma_1) |+\rangle^\otimes m. \tag{17}
\]

The parametrized quantum gates are then optimized in a closed loop using a classical optimizer.

Thus to have the probability \((1 - \varepsilon)\) of observing the solution, \(m\) has to be

\[
m > \frac{\log \varepsilon}{\log (1 - F_p(\vec{\gamma}, \vec{\beta})).} \tag{22}
\]

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.

| 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 allows us to restrict the domain of each \( \gamma_i \) 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 \( E_1(\vec{\gamma}, \vec{\beta}) \), as well as the corresponding success probability \( F_1(\vec{\gamma}, \vec{\beta}) \), as a function of \( \gamma \) and \( \beta \), 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 \( |\gamma_{\text{exp}} - \gamma_{\text{succ}}| \approx 0 \) and \(|\beta_{\text{exp}} - \beta_{\text{succ}}| \approx 0.047 \). 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 translate 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]. The corresponding plots for one of the 8 and 15 route instances are shown in Appendix B. We note that all figures have qualitatively similar shape and that the optimal variational parameters for $p = 1$ are located in the same region.

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. [24], 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 computationally 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 [25] for every start point from which it records the global optimum. We provide relevant details in Appendix A. In Fig. 3 we present the optimal variational parameters $(\vec{\gamma}^*, \vec{\beta}^*)$ from $p = 3$ to $p = 5$ for the 8 route instances. We observe that a persistent pattern shows up, and that both $\gamma_i$ and $\beta_i$ 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.

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 [24], 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 $p + 1$ based on the best variational parameters found at level $p$. From the produced starting-point we run the gradient-free Nelder-Mead method [26, 27], which is reported in Ref. [24] to work equally well as the BFGS method, for this heuristic strategy. The Nelder-Mead algorithm was implemented in MATLAB version R2019b using the \texttt{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 60$p$. We furthermore make the assumption that the same pattern in the variational parameters exists for the 25 route instances, and we therefore use the interpolation strategy mentioned above for 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 $p = 20$, see Fig. 4(b). 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 visualised in Fig. 4(d), 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. However, this does not yet answer the question as to why denser graphs are more difficult to solve with the QAOA. To elucidate further this matter we recall, following Ref. [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(d), and “walking” along the graph at most $p$ steps away from that edge, for a given iteration level $p$. Indicating with $F_p(\vec{\gamma}, \vec{\beta})$ 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 
\[
E_p(\vec{\gamma}, \vec{\beta}) = \sum_g w_g f_p(\vec{\gamma}, \vec{\beta}).
\]
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(d) and Table II) those possess higher average valency.

Finally, in Fig. 4(c) 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.

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               |

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 qubit 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 [28, 29] is beyond the scope of the present paper, but stems as an interesting perspective.

Finally, it is 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. To our knowledge, the currently best known classical algorithm for solving the Exact Cover problem is Algorithm X, developed by Donald Knuth, which, as expected for NP-complete problems, runs in exponential time for worst-case-type instances [30]. 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 Algorithm X, 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 
\[
- \log \langle e^{-\eta H_C} \rangle,
\]
where \( \eta > 0 \), instead of the expectation value Eq. (8) [31]. 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 through Wallenberg Centre for Quantum Technology. 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).
Appendix A: 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 minima 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 numerical solver which 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 [25] which is implemented as fmincon in MATLAB. The number of random start points was chosen to be $4 \times 10^3$. This number was em-
typically 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 $|F_p(\vec{\gamma}, \vec{\beta}) - F_p(\vec{\gamma}', \vec{\beta}')| < \text{FunctionTolerance}$, the iteration ends. The second one, the step tolerance, is such that if the solver attempts to take a step that is smaller than $|\gamma - \gamma'|^2 + |\beta - \beta'|^2 < \text{StepTolerance}$, the iteration ends. Both \text{StepTolerance} and \text{FunctionTolerance} were set to their default values which was $10^{-6}$.

**Appendix B: Additional figures**

**FIG. 5.** (color online) Simulation results for one of the 8 route instances as a function of $\gamma$ and $\beta$ 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 $\gamma$ and $\beta$ 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.