IEEE Transactions on Quantum Engineering

Received August 31, 2020; accepted September 25, 2020; date of publication October 13, 2020; date of current version November 10, 2020.

Digital Object Identifier 10.1109/TQE.2020.3030609

Toward Quantum Gate-Model Heuristics for Real-World Planning Problems

TOBIAS STOLLENWERK[1], STUART HADFIELD[2,3], AND ZHIHUI WANG[2,3]

[1] German Aerospace Center (DLR), 51147 Cologne, Germany
[2] Quantum Artificial Intelligence Lab, NASA Ames Research Center, Moffett Field, CA 94035 USA
[3] USRA Research Institute for Advanced Computer Science, Mountain View, CA 94043 USA

Corresponding author: Tobias Stollenwerk (Tobias.Stollenwerk@dlr.de)

This work was supported in part by the German Aerospace Center (DLR) and in part by the NASA Ames Research Center. The work of Stuart Hadfield and Zhihui Wang was supported in part by the AFRL Information Directorate under Grant F4HBKC4162G001, in part by the Office of the Director of National Intelligence (ODNI), in part by the Intelligence Advanced Research Projects Activity (IARPA) via IAA 145483, in part by the NASA Advanced Exploration Systems program, in part by the NASA Transformative Aeronautic Concepts Program, and in part by the NASA Academic Mission Services under Contract NNA16BD14C. This article was presented in part at the 2nd International Workshop on Quantum Resource Estimation (QRE2020), Valencia, Spain, May 2020.

ABSTRACT Many challenging scheduling, planning, and resource allocation problems come with real-world input data and hard problem constraints, and reduce to optimizing a cost function over a combinatorially defined feasible set, such as colorings of a graph. Toward tackling such problems with quantum computers using quantum approximate optimization algorithms, we present novel efficient quantum alternating operator ansatz (QAOA) constructions for optimization problems over proper colorings of chordal graphs. As our primary application, we consider the *flight-gate assignment problem*, where flights are assigned to airport gates as to minimize the total transit time of all passengers, and feasible assignments correspond to proper graph colorings of a conflict graph derived instancewise from the input data. We leverage ideas from classical algorithms and graph theory to show our constructions have the desirable properties of restricting quantum state evolution to the feasible subspace, and satisfying a particular reachability condition for most problem parameter regimes. Using classical preprocessing we show that we can always find and construct a suitable initial quantum (superposition) state efficiently. We show our constructions in detail, including explicit decompositions to a universal set of basic quantum gates, which we use to bound the required resource scaling as low-degree polynomials of the input parameters. In particular, we derive novel QAOA mixing operators and show that their implementation cost is commensurate with that of the QAOA phase operator for flight-gate assignment. A number of quantum circuit diagrams are included such that our constructions may be used as a template toward development and implementation of quantum gate-model approaches for a wider variety of potentially impactful real-world applications.

INDEX TERMS Combinatorial optimization, operations research, quantum algorithms, quantum approximate optimization, quantum circuits, quantum computing, quantum resource estimation.

I. INTRODUCTION

Improving our ability to satisfactorily solve challenging real-world combinatorial optimization problems, in particular those related to operational planning and scheduling, is a broad and promising area for potential quantum advantages. Efficient classical approaches to such problems are often heuristic and may lack performance or runtime guarantees; even minor improvements to the solution quality or guarantee can have a significant impact in terms of real-world resource costs, such as the time, money, and energy required to find and implement a satisfactory solution in practice.

An important general class of such problems are scheduling problems with hard constraints related to graph coloring. Often the set of feasible assignments for such problems corresponds to proper colorings of a derived problem graph, such that any scheduling conflicts are avoided, over which we seek to minimize a cost function that incorporates the problem input data. An important practical example is the broad range of problems related to the scheduling of a set temporal events, where the difficulty of finding an optimal schedule depends in a fundamental way on the input data that describes the event durations and other requirements (i.e.,

the derived problem graph may vary dramatically over the same underlying class of problem instances). In this article, we focus on the model problem of optimizing the assignment of airplanes to gates at an airport, the *flight-gate assignment problem* [1]. Applications similar in spirit include, for example, assigning aircraft to flights [2], earth observation satellite mission planning [3], personnel scheduling [4], frequency assignment [5], and content placement in cache networks [6].

Recent work has begun to explore using early quantum devices for tackling optimization problems. While much work has focused on the quantum annealing paradigm [7], [8], reflecting initial hardware availability, recent results have demonstrated that near-term quantum gate-model devices may also be employed for such problems. In particular, the quantum approximate optimization algorithm [9]–[11] as well as its generalization to problems with hard constraints, the quantum alternating operator ansatz (QAOA) [12], utilize cost-based and solution-mixing operators to explore and sample from the space of possible problem solutions, as we elaborate on below. For convenience, we will refer to both variants as QAOA.

Previously, our flight-gate assignment problem was mapped to quantum annealing (adiabatic quantum optimization) and small instances were successfully solved with the D-Wave 2000Q device [13]. In that work, similar to other studies toward real-world problems [7], [8], in order to obtain a problem representation suitable for implementation, the problem hard constraints were reformulated as additional penalty terms in the cost function of a transformed unconstrained problem. Treating the hard constraints in this way, standard in the quantum annealing paradigm, has various drawbacks beyond the increase in physical qubits and couplers typically required. In [13], it was shown for this problem that the penalty terms lead to a significant increase in the precision and coupling requirements for the quantum annealer and therefore to a suppression of the success probability in practice, with similar observations made for different aerospace problems in [14]. Indeed, a motivation for gate-model approaches, such as QAOA, is to enable a wider variety of quantum algorithms to be implemented and explored than those of the more restricted setting of quantum annealing [12]. Generally, the fraction of states, which encode feasible solutions, often becomes exponentially small [15]. Hence, it appears desirable to incorporate hard constraints directly into the driver Hamiltonian for annealing [16], [17], or into the mixing unitary for QAOA [12] such that penalty terms can be avoided and the above issues somewhat mitigated, though coming with different potential tradeoffs involved with implementing the alternative operators; we consider the latter approach in our quantum gate-model constructions to follow.

In this article, we show explicit constructions of QAOA circuits for optimization problems over the proper colorings of an important class of graphs, interval graphs, common in scheduling and assignment problems. As an application, we consider the flight-gate assignment problem [1], [13] in detail. Leveraging ideas from graph theory, we construct novel generalizations of prior QAOA mixers and show both that they preserve state feasibility always, and that any two feasible solutions can be connected by a sequence of such mixers, under mild conditions related to the problem input data. Our constructions allow for incorporation of real-world problem instance data in a straightforward way and with minimal additional overhead. For each subcircuit employed in our constructions, we show a circuit decomposition into basic quantum gates from which we derive estimates of the overall resources required. For the reader's benefit, a number of explicit examples and quantum circuit diagrams are provided as figures throughout.

This article is structured as follows. We next introduce the flight-gate assignment problem and explain its relationship to graph coloring in Section II. We show its QAOA construction in detail in Section III. For our ansätze, quantum circuit decompositions as well as quantum resource estimates are presented in Section IV. We provide discussion of a number of key concepts and open research directions throughout the text, and Section V concludes this article with a summary of our results and discussion of their applicability to a wider variety of problems.

II. FLIGHT-GATE ASSIGNMENT PROBLEM

Airport flight-gate assignment problems have been considered in several different variants [18], [19]. Our formulation as a quadratic binary program with linear constraints follows from [1] and was previously considered for quantum annealing in [13]. Given a temporal schedule of flight arrivals and departures, the passengers on each flight, and an airport-dependent list of passenger transit times between gates, we seek an assignment of flights to gates such that the total transit time of all passengers is minimized (or as small as possible). Assignments must satisfy the hard constraints of allocating a gate for each flight while avoiding assigning any pair of flights with a temporal conflict to the same gate.

The flight-gate assignment optimization problem is a quadratic assignment problem that is NP-hard in general [20]. Hence, we must often settle for approximate solutions in practice, and heuristics are often applied to tackle real-world problem instances within a given computational resource budget. Indeed, a number of classical exact or heuristic approaches have been applied to flight-gate assignment or similar problem variants, including Tabu search [1], [21], integerprogramming [19], and colony optimization [22], among others [23].

A. PROBLEM FORMULATION

A flight-gate assignment problem instance consists of a set of flights V and set of gates K, and a list of flight, airport, and passenger data we describe below and summarize in Table 1. The number of flights is denoted by $n := |V|$ and number of gates by $k := |K|$; we elaborate on the important relationship between these parameters in Section II-B. Gate assignments

IEEE Transactions on Quantum Engineering

TABLE 1. Parameters for a Flight-Gate Assignment Problem Instance

Summary of Problem Data	
Symbol	Meaning
V	Set of n flights ($i \in V = \{1, 2, \ldots, n\}$)
K	Set of k gates ($\alpha \in K = \{1, 2, \ldots, k\}$)
n_i^{dep}	# of passengers departing with flight i
n_i^{arr}	# of passengers arriving with flight i
n_{ij}	# of transfer passengers between flights i and j
t_i^{in}	Arrival time of flight i
t_i^{out}	Departure time of flight i
t_α^{arr}	Transfer time from gate α to baggage claim
t_α^{dep}	Transfer time from check-in to gate α
$t_{\alpha\beta}$	Transfer time from gate α to gate β
t^{buf}	Buffer time between two flights at the same gate
E	Set of forbidden flight pairs
$G = (V, E)$	Conflict graph
P	Set of flight pairs with transfer passengers
$T = (V, P)$	Transfer passenger graph
$c(x)$	Cost function: total transit time of flight assignment x

The top of the table gives the problem input data from which the bottom quantities are derived. Feasible flight-gate assignments correspond to proper vertex colorings of the conflict graph G that use at most k colors. The transfer passenger graph T is used to derive the cost function.

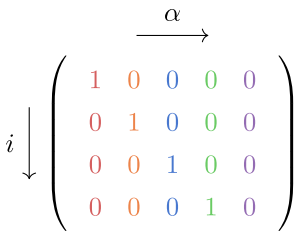


FIGURE 1. Depiction of the binary variables $x_{i\alpha}$ corresponding to the n flights and k gates as an $n \times k$ matrix. The column colors indicate different gates. In general, we may have $k > n$, $k = n$, or $k < n$, which determines the shape of the matrix. Here, for example, we take $n = 4$ and $k = 5$.

are encoded using nk binary decision variables

$$x_{i\alpha} = \begin{cases} 1 & \text{if flight } i \in V \text{ is assigned gate } \alpha \in K \\ 0 & \text{else} \end{cases} \tag{1}$$

for $i \in V$ and $\alpha \in K$. It will be useful for following our constructions to think of the variables $x_{i\alpha}$ as the entries of an $n \times k$ dimensional 0-1 matrix as shown in Fig. 1. An assignment of flights to gates then has a single 1 in each row.

The cost function we seek to minimize is comprised of three parts as depicted in Fig. 2. First, the total transit time of all departing passengers is given as the sum of the individual transit times for departing passengers, as determined by the time t_α^{dep} it takes to get from the check-in to a gate α, as well as the number n_i^{dep} of passengers departing with a flight i

$$c^{\text{dep}}(x) := \sum_{i\alpha} n_i^{\text{dep}} t_\alpha^{\text{dep}} x_{i\alpha}.$$

Next, there is similarly a contribution from the total transit time of all arriving passengers

$$c^{\text{arr}}(x) := \sum_{i\alpha} n_i^{\text{arr}} t_\alpha^{\text{arr}} x_{i\alpha}$$

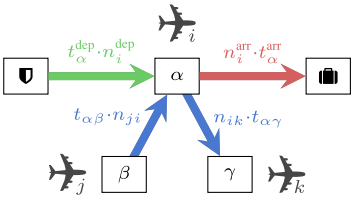


FIGURE 2. Total transit time cost function at a gate α given by three contributions from the departing passengers (green), the arriving passengers (red), and the transfer passengers (blue).

where t_α^{arr} gives the time it takes to get from gate α to the baggage claim, and n_i^{arr} gives the number of passengers arriving with flight i and leaving the airport. The last contribution to the cost function is the total transfer passenger transit time

$$c^{\text{trans}}(x) := \sum_{\substack{i \le j \\ \alpha \le \beta}} n_{ij} t_{\alpha\beta} x_{i\alpha} x_{j\beta}$$

determined by the time $t_{\alpha\beta}$ it takes to travel between gates α and β, and the number of passengers n_{ij} arriving from flight i and departing with flight j or vice versa. Altogether, the total passenger transit time cost function is

$$c(x) := \sum_{i\alpha} \left(n_i^{\text{dep}} t_\alpha^{\text{dep}} + n_i^{\text{arr}} t_\alpha^{\text{arr}} \right) x_{i\alpha} + \sum_{\substack{i \le j \\ \alpha \le \beta}} n_{ij} t_{\alpha\beta} x_{i\alpha} x_{j\beta} \,. \tag{2}$$

The number of nonzero terms in the cost function depends on the *transfer passenger graph* $T := (V, P)$, which has a vertex for each flight and edges $P := \{(i, j) \,|\, n_{ij} > 0\}$ that indicate transfer passengers between two fights. The corresponding QAOA circuit decompositions and resource estimates we show in Section IV-A depend on the quantity $|P| \le \binom{n}{2}$.

We remark that while we consider here the minimization of total passenger travel time, our methods and results to follow may easily be extended to other cost functions or objectives, e.g., minimizing total aircraft taxi time, or extremizing a linear combination of different objective functions [1].

1) PROBLEM HARD CONSTRAINTS

A *feasible gate assignment* must satisfy several hard constraints. Each flight must be assigned to exactly one gate. This is reflected in the first hard constraint

$$\forall i \in V : \qquad \sum_{\alpha} x_{i\alpha} = 1 \tag{3}$$

which implies a single 1 in each row of the matrix representation of Fig. 1. Following [1], pairs of flights are not allowed to be assigned to the same gate if they will simultaneously require gate access at any point in the schedule, up to a buffer time t^{buf}. The duration at the gate for each flight i is defined as the temporal interval

$$\Delta_i := \left(t_i^{\text{in}} - t^{\text{buf}}, \; t_i^{\text{out}} + t^{\text{buf}} \right)$$

where t_i^{in} and t_i^{out} are the arrival and departure times of flight i. The set of flight pairs with overlapping temporal intervals is called the *set of forbidden flight pairs* and suggestively denoted as

$$E := \{(i, j) \mid \Delta_i \cap \Delta_j \neq \emptyset\} .$$

Clearly, each pair of flights in E cannot be assigned to the same gate, which corresponds to the constraint

$$\forall (i, j) \in E,\ \forall \alpha \in K : \qquad x_{i\alpha} x_{j\alpha} = 0 . \tag{4}$$

In the matrix representation of Fig. 1, this implies that for each pair $(i, j) \in E$, no column can contain a 1 in both rows i and j for the encoded flight assignment to be feasible. Hence, the set of feasible assignments [i.e., assignments (1) satisfying (3) and (4)] correspond bijectively to the proper colorings of the conflict graph $G = (V, E)$ with vertex set V, edge set E, and color set K; we next elaborate on this important connection.

B. RELATIONSHIP TO GRAPH COLORING

Flight-gate assignment is closely related to graph vertex coloring. As explained, by considering the n flights as graph vertices and k gates as colors, feasible gate assignments correspond precisely to proper k-colorings of the *conflict graph* $G := (V, E)$, i.e., the graph of forbidden flight pairs. Hence, the set of feasible gate assignment depends fundamentally on the structure of G, which varies with each problem instance; in particular, on the chromatic number $\chi(G)$ defined as the minimum number of colors necessary to properly color G. We emphasize that a flight-gate assignment (proper coloring) that minimizes the cost function (2) is not necessarily one using the minimal number of gates (colors). Clearly, the difficulty of even finding a feasible assignment to begin with will depend on G and, in particular, the relative magnitudes of n, k, and $\chi(G)$. Trivially, when $k \geq n$, we may arbitrarily assign a different color to each vertex to achieve a proper coloring. For example

$$x_{i\alpha} = \begin{cases} 1 & \text{for } i = \alpha \\ 0 & \text{else.} \end{cases} \tag{5}$$

On the other hand, when $k < n$, we have fewer colors than vertices and it is NP-hard in general to even decide if a proper coloring exists. Fortunately, we take advantage of the problem structure to show that a proper coloring of the conflict graph can always be found efficiently (and hence initial feasible QAOA states can be efficiently constructed), or certify the instance to have no solution (when $k < \chi(G)$), for any possible problem parameters. We remark that the cases $k > \chi(G)$ may be considered to be most practically relevant, as assignments for instances with $k = \chi(G)$ are not robust to small schedule changes (i.e., if a single airport gate becomes unavailable during the course of a schedule, or a flight is delayed as to increase $\chi(G)$, then the schedule necessarily becomes infeasible).

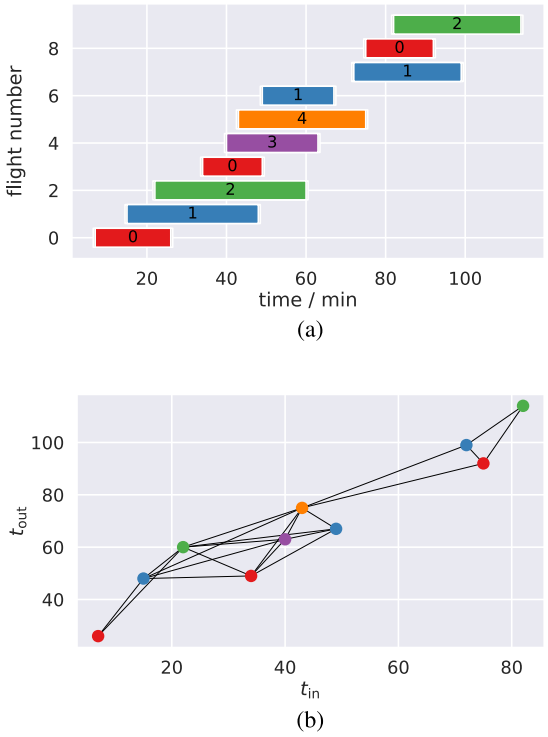


FIGURE 3. Example flight-gate assignment problem instance with $k = 5$ gates and $n = 10$ flights. The five different colors indicate the five gates. (a) Total gate duration intervals for each flight, with interval colors and labels indicating a valid gate assignment. (b) Corresponding conflict graph $G = (V, E)$ plotted in the plane spanned by arrival and departure times. The clique of five vertices shows that the $k = 5$ gates are necessary in this example.

Observe from the derivation of the edge set E it follows that G is an *interval graph* [24], with nodes representing time intervals and edges corresponding to interval overlaps. Fig. 3 shows an example problem and the resulting interval graph. For such graphs,[1] we can efficiently find and apply a *perfect elimination ordering* to the vertices, combined with a simple classical greedy algorithm, to efficiently find a proper coloring (using the minimal number of colors). If this coloring uses at most k colors, then we have found a feasible starting solution, else, no proper colorings exist and so the given problem instance is certified to have no solution. We elaborate on the details of this classical algorithm and summarize our procedure for finding an initial feasible flight assignment in Algorithm 1. We utilize this procedure as a classical preprocessing step in our QAOA constructions.

[1] Interval graphs are chordal (and hence perfect). A chordal graph is one in which every cycle of four of more vertices contains a chord (an additional edge connecting two vertices in the cycle), or equivalently, if it has a perfect elimination ordering [25]. For chordal graphs, we can efficiently compute and evaluate the chromatic polynomial [26], which gives the number of proper k-colorings for each k, which is #P-hard for general graphs. A graph is perfect when the chromatic number is equal to the size of the largest clique for itself and every induced subgraph. We utilize these properties in our QAOA constructions to follow.

IEEE Transactions on Quantum Engineering

Algorithm 1: Initial Assignment (Proper Coloring).

Input: Problem instance data (summarized top of Table 1)

Output: A feasible flight-gate assignment $\Gamma : V \mapsto K$

1: Construct conflict graph $G = (V, E)$ from input data
2: Arbitrarily order the colors (gates) $\alpha_1 < \ldots < \alpha_k l$
3: Reindex the vertices (flights) V by computing a perfect elimination ordering $v_1 \leq v_2 \leq \cdots \leq v_n$
4: **for** $i = 1$ to n **do**
5: Set $\Gamma(v_i)$ to be the smallest color not used by any of the neighbors of v_i which precede it in the ordering
6: **if** no unused colors remain **then**
7: **return** "Infeasible problem instance"
8: **return** Proper coloring Γ

1) FINDING INITIAL FEASIBLE ASSIGNMENTS

A perfect elimination ordering is a total ordering of the vertices of a graph such that, for each vertex v, v and its neighbors that appear later in the order form a clique. For chordal graphs, a perfect elimination ordering can be found in time $O(n + |E|)$ using lexicographical breadth-first search [27]. Given such an ordering, an optimal coloring can then be found in linear time with the simple greedy choice indicated in Step 5 of Algorithm 1. It is straightforward to show that if a proper k-coloring of G exists, then Algorithm 1 returns a minimal $\chi(G)$-coloring. Clearly, given such a coloring, additional feasible assignments may be easily found, for example, by permuting the order of the colors.

Thus, we have a classical procedure to find a proper coloring of an interval graph, or certify no such assignment exists, in time $O(n^2)$, for all possible values of n, k, and possible conflict graphs G, i.e., for all possible flight-gate assignment instances. Hence, for the remainder of this article, we assume $k \geq \chi(G)$. Importantly, Algorithm 1 implies we can always efficiently construct a suitable initial quantum state for QAOA as we describe below.

2) RELATIONSHIP TO OTHER OPTIMIZATION PROBLEMS

In certain parameter regimes, our problem reduces to other well-known optimization problems; one may take advantage of such special cases to substitute simpler quantum circuits yielding reduced resource requirements. Consider the case when $k = n = \chi(G)$, which implies each vertex must have a different color (each flight is assigned to a different gate) and n gates are necessary. This case occurs when the conflict graph is the complete graph $G = (V, E) = K_n$. The feasible assignments in this case are easily seen to be equivalent to the set of total orderings on n objects, i.e., the hard constraints reduce to the simpler requirement that each flight be assigned a unique gate. Hence, this case is effectively equivalent to the traveling salesperson problem (TSP), where each color indicates a different position in the ordering of the cities (vertices), such that a proper coloring gives a tour, but with tour cost assigned according to (2). Another interesting special case is when $E = \emptyset$, which implies $\chi(G) = 1$ and so the hard constraints become trivial. Here, each vertex can be colored independent of all other vertices, and the problem reduces to unconstrained optimization over k-valued dits (as opposed to bits) [12]. In Section III-C4, we show how both these cases result in QAOA constructions with simpler quantum operators and reduced resource requirements.

III. QAOA FOR FLIGHT-GATE ASSIGNMENT

We now show constructions of QAOA for the flight-gate assignment problem. We apply the techniques of [12] to design suitable problem encodings and mixing operators such that the probability amplitude is restricted to quantum states encoding feasible solutions. Our circuits yield parameterized quantum states for generating candidate flight-gate assignments (via repeated state preparations and measurements). Measuring such a state in the computational basis returns a feasible gate assignment.

A *QAOA* [12] consists of an initial state $|s\rangle$ and two unitary operators, a *phase separation operator* $U_{\mathrm{P}}(\gamma)$ and a *mixing operator* $U_{\mathrm{M}}(\beta)$, each parameterized by real parameters γ and β, applied in alternation p times each to create the quantum state

$$|\boldsymbol{\gamma\beta}\rangle_p = U_{\mathrm{M}}(\beta_p)U_{\mathrm{P}}(\gamma_p)U_{\mathrm{M}}(\beta_{p-1})\cdots U_{\mathrm{M}}(\beta_1)U_{\mathrm{P}}(\gamma_1)|s\rangle\,.$$

We use QAOA_p to denote such a level-p QAOA circuit. Thus, for a given problem, it suffices to specify the operators $U_{\mathrm{P}}(\gamma)$ and $U_{\mathrm{M}}(\beta)$ and state $|s\rangle$, such that $|s\rangle$ belongs to the feasible subspace and $U_{\mathrm{M}}(\beta)$ preserves the feasibility property while it enables the exploration of all possible feasible states; see [12, Sec. 3.1] for a detailed discussion of *QAOA design criteria*.

Given a problem and a suitable mapping taking each instance to an efficiently implementable QAOA, we may apply the *quantum approximate optimization algorithm* as follows:

1) create the ansatz state $|\boldsymbol{\gamma\beta}\rangle_p$ for some algorithm parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$;
2) measure in the computational basis (which returns a feasible flight-gate assignment x);
3) repeat the first two steps a number of times, keeping the best solution found.

Commonly, samples drawn are used to estimate a quantity such as the expectation value of the cost function for the quantum ansatz state with a fixed set of parameters, from which the parameters may be iteratively updated, variationally or otherwise. Clearly, the cost of such a procedure proportionately increases the overall number of quantum state preparations and measurements required. In some cases, we can determine satisfactory parameters *a priori* [28], [29], but finding generally effective parameter setting strategies remains a challenging open problem. Similarly, most questions remain unresolved as to the fundamental performance

achievable for such algorithms. We do not attempt to resolve either issue here and focus instead on showing efficiently constructable ansatz states and demonstrating their potential to be implemented on quantum hardware of the not-too-distant future. Indeed, such near-term experiment may be vital for informing the development of future quantum algorithms and heuristics for optimization; see, e.g., [12] for further discussion.

A. INITIAL STATE

When there are at least as many gates as flights $k \geq n$, a computational basis state $|x\rangle$ encoding an initial feasible gate assignment $x \in \{0, 1\}^{nk}$ can be trivially constructed as in (5), where x may be selected arbitrarily, for example, as the best assignment found by a classical algorithm, or randomly, as desired. Indeed, for any k, an initial state can be found efficiently with classical preprocessing as described in Section II-B1, or the problem certified infeasible if no feasible assignments exist. The corresponding quantum state $|x\rangle$ can be created with n Pauli X-gates (bit flips) applied in parallel to the nk-qubit state $|00\ldots0\rangle$. We denote the corresponding unitary by U_x with $U_x|00\ldots0\rangle = |x\rangle$.

On the other hand, when a single classical feasible state $|x\rangle$ is used as initial state, the first phase operator $U_P(\gamma_1)$ acts as a global phase and can be ignored. Indeed, it may be desirable to use quantum superpositions of classical feasible states as initial states [10], [12]. In particular, such a state $|s\rangle := U_{\text{init}}|0\rangle$ can be constructed by applying a single layer of the mixing operator $U_{\text{M}}(\beta)$ to the state $|x\rangle$, leading to the total initial state preparation unitary

$$U_{\text{init}}(\beta_0) = (U_{\text{M}}(\beta_0))^{r_0} U_x \tag{6}$$

applied to the $|00\ldots0\rangle$ state, where $r_0 \in \mathbb{N}$ is a repetition parameter, and U_{M} is a suitable mixing operator which ensures that $U_{\text{init}}(\beta_0)|0\rangle$ is restricted to the feasible subspace, as we elaborate on below. The mixing angle β_0 determines the distribution of amplitudes over a set of classical feasible states with support in $|s\rangle$ (i.e., $|x\rangle$ such that $|\langle x|s\rangle| > 0$), and may be fixed (e.g., setting $\beta_0 = \pi/6$) or treated as an additional ansatz free parameter. For such mixers, we often have $U_{\text{M}}(\beta)U_{\text{M}}(\beta) \neq U_{\text{M}}(2\beta)$, which motivates taking $r_0 > 1$ to produce superpositions $|s\rangle$ supported on increasing numbers of feasible basis states. For example, if we take $r_0 = O(1)$, then this additional mixing stage will not affect the overall resource estimates significantly. In general, the quantum circuit U_{init}, where U_x may be derived as in Algorithm 1, gives a way of generating nontrivial quantum superpositions of states encoding feasible flight-gate assignments. We show resource estimates for the initial state preparation in Section IV-D.

B. PHASE SEPARATION OPERATOR

For the phase separation operator, the cost function $c(x)$ in (2) is encoded as the diagonal cost Hamiltonian C that acts on nk-qubit computational basis states as $C|x\rangle = c(x)|x\rangle$. Applying standard techniques for mapping such functions to Pauli Z operators [30] yields the quadratic Hamiltonian

$$C = c_0 I + \sum_{i\alpha} c_{i\alpha} Z_{i\alpha} + \sum_{\substack{i\leq j \\ \alpha\leq\lambda}} c_{ij\alpha\lambda} Z_{i\alpha} Z_{j\lambda} \tag{7}$$

where $c_0 = \frac{1}{2}\sum_{i\alpha}(n_i^d t_\alpha^d + n_i^a t_\alpha^a) + \frac{1}{4}\sum_{ij\alpha\lambda} n_{ij} t_{\alpha\lambda}$, $c_{i\alpha} = -\frac{1}{2}(n_i^d t_\alpha^d + n_i^a t_\alpha^a) + \frac{1}{2}\sum_{j\lambda} n_{ij} t_{\alpha\lambda}$, and $c_{ij\alpha\lambda} = \frac{1}{4}\sum_{i\leq j}\sum_{\alpha\leq\lambda} n_{ij} t_{\alpha\lambda}$. (Explicitly, the symbol $Z_{i\alpha}$ indicates the Pauli Z operator acting on the qubit with index $i\alpha$ that encodes the variable $x_{i\alpha}$, with implicit identity factors acting on the remaining qubits.) We employ the standard QAOA phase separation operator construction given by time evolution under the cost Hamiltonian

$$U_{\text{P}}(\gamma) := \exp(-\text{i}\gamma C)\,.$$

As the terms in (7) mutually commute, $U_{\text{P}}(\gamma)$ may be efficiently and exactly implemented as a product of 1- and 2-local qubit rotations (see Section IV-A).

C. MIXING OPERATORS

In this section, we derive novel mixing operators for problems with feasible solutions equivalent to proper graph colorings of chordal graphs, including, but not limited to, the flight-gate assignment problem. Following the approach of [12], we decompose each mixer U_{M} as an ordered product of unitaries $U_{\text{M}} = U_{\text{M},\ell}\ldots U_{\text{M},2}U_{\text{M},1}$, where the simpler *partial mixers* $U_{\text{M},j}$ do not mutually commute in general, such that different sequences of partial mixers may yield distinct mixing operators. As only the number of partial mixers, not their order, affects our resource estimates to follow, we do not deal in detail with term ordering selection, which may be hardware or instance dependent; see [12] for discussion.

We propose controlled-unitary partial mixers of the form

$$U_{\text{M},j} = \Lambda_{f_j}(U)$$

where this notation indicates a target "action unitary" U controlled by the value of a Boolean function f_j, with control and target acting on distinct sets of qubits. For example, the controlled-not (CNOT) gate may be written as $\Lambda_x(X) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X$, where x indicates the value of the first (qu)bit and the Pauli X operator flips the state of the second. We employ the strategy of [12] of designing such U to encode (or generalize) classical moves for exploring the space of feasible solutions. The primary move we consider is that of changing the color of a given vertex, i.e., changing the gate assigned to a particular flight, such that feasibility is preserved. This results in control functions f_j that depend on the colors of the vertices adjacent to the those involved in U. We define these operations precisely below. Moreover, we leverage results from graph theory to show that a sequence of such partial mixers suffices to connect any two feasible assignments for almost all problem cases.

According to the design criteria of [12], a full mixer $U_{\text{M}}(\beta)$ should i) preserve the feasible subspace (i.e., map feasible states to feasible states), and ii) enable reaching the whole feasible subspace in the following sense. For all pairs of

IEEE Transactions on Quantum Engineering

TABLE 2. Mixing Operators and Problem Domains Where They Suffice in the Sense of the Reachability Criterion of Proposition 1, in Terms of the Number of Flights *n*, Number of Gates *k*, and the Conflict Graph Chromatic Number $\chi(G)$

Domain	Mixer	Definition
$k > \chi(G)$	Controlled gate-change mixer $U_{\mathrm{MC}}(\beta)$ Alternative: Controlled gate-change-and-swap mixer $U_{\mathrm{MCS}}(\beta_1, \beta_2)$	Eq. (12) Eq. (16)
$k = \chi(G) = n$ $\chi(G) = 1$	TSP-mixer $U_{\mathrm{TSP}}(\beta)$ XY-mixer $U_{XY}(\beta)$	Eq. (19) Eq. (20)

The bottom of the table gives two special cases corresponding to complete and isolated vertices conflict graphs.

feasible configurations x, y, there exists β and integer r such that $\langle x|(U_{\mathrm{M}}(\beta))^r|y\rangle > 0$; we refer to the latter condition as the *reachability criterion* for the remainder of this article. We remark that for our purposes, reachability is a measure of soundness in mixer design; smaller values of r may suffice in practice. For example, for level-p QAOA, we may select r for each mixing stage such that the reachability criterion is instead satisfied for the product rp. Alternatively, we may take r to be a small constant in order to reduce resource requirements. Indeed, the precise relationship of such reachability criterion to algorithm performance remains unclear; a detailed investigation of mixer parameter selection and performance is beyond the scope of this article.[2]

Nevertheless, our constructions below satisfy i) by design, and we show sufficient problem parameter regimes such that ii) is guaranteed. In particular, we show that the controlled color-change mixer U_{MC}^r [as defined below in (12)] satisfies the reachability criteria for most cases of the flight-gate assignment problem, with the exception of the corner cases where the number of gates k is equal to the conflict graph's chromatic number $\chi(G)$, i.e., when the problem instance is such that removing a single gate makes the instance infeasible.

Table 2 summarizes the mixing operators we detail in the remainder of this section. We denote as QAOA$_p$–MC a level-p QAOA circuit using the mixer U_{MC} (or U_{MC}^r, as discussed). We also show an alternative mixer U_{MCS} [as defined below in (16)] for the same set of instances as U_{MC}, as well as further mixers for special cases as indicated in the table. The more general mixer U_{MCS} takes an extra mixing parameter that facilitates additional mixing between assignments using the same numbers of colors (gates), based on a different classical search move (see Section III-C3). The remaining corner cases $k = \chi(G) < n$ not covered by the entries of the table are discussed in Section III-C4. Quantum circuit decompositions and resource estimates for these mixing operators are presented in Section IV.

[2]For example, when $0 < \Delta\beta \ll 1$ and r is proportional to $\beta/\Delta\beta$, the operator $(U_{\mathrm{M}}(\Delta\beta))^r$ may be viewed as a Trotterized approximation of the alternative *simultaneous Hamiltonian mixer* approach for QAOA of [10], [12].

1) PARTIAL CONTROLLED COLOR-CHANGE MIXERS

We propose partial mixers $U_{i\lambda\alpha}^{\mathrm{MC}}$ related to swapping the color of vertex i between the colors λ and α, controlled by the coloring of vertices that are adjacent to i in the conflict graph G. We describe the underlying classical mixing rule and show the resulting action of the mixers on quantum states.

For each flight i, let $N(i) = \{j \mid (i, j) \in E\}$ denote the set of flights that are forbidden to be assigned to the same gate, i.e., the neighbors of vertex i in G. We denote the number of neighbors, i.e., the degree of vertex i, as $d_i = |N(i)|$.

Consider a feasible initial assignment x, and the pair of variables $x_{i\lambda}, x_{i\alpha}$ that indicate if vertex i is colored λ or α. From (3), at most one of these variables is set to 1. When flight i is colored λ or α, if none of the vertices in $N(i)$ are colored λ or α, then we may safely swap the values of $x_{i\lambda}$ and $x_{i\alpha}$ as to change the color of vertex i between λ and α, else we do nothing. Moreover, such a color change acts trivially when $x_{i\lambda} = 0 = x_{i\alpha}$. We refer to this overall transformation as a *classical controlled color-change* move.

Hence, we define *partial controlled color-change mixers*

$$U_{i\lambda\alpha}^{\mathrm{MC}}(\beta) := \Lambda_{f_{i\lambda\alpha}(x)}\left(U_{i\lambda\alpha}^{XY}(\beta)\right) \tag{8}$$

where we employ *partial XY-mixers* [12], [15], [31] defined as

$$U_{i\lambda\alpha}^{XY}(\beta) := \exp\left(-\mathrm{i}\beta\left(X_{i\lambda}X_{i\alpha} + Y_{i\lambda}Y_{i\alpha}\right)/2\right) \tag{9}$$

where the factor $\frac{1}{2}$ is included for convenience. This XY-rotation is controlled by the condition that all of the neighbors of vertex i are colored neither λ nor α, i.e., the Boolean function

$$f_{i\lambda\alpha}(x) := \wedge_{j\in N(i)}\bar{x}_{j\lambda}\bar{x}_{j\alpha}\,. \tag{10}$$

The Hamiltonian $(X_1X_2 + Y_1Y_2)/2$ is closely related to the two-qubit SWAP operation (see [12, App. C] and [15]).

The action of the color-change partial mixer is depicted in Fig. 4. Acting on quantum superpositions of feasible states, this partial mixer either rotates between assignments where vertex i is colored λ or α, on the subspace of states where this operation preserves feasibility, or otherwise acts as the identity. Consider a feasible assignment $|x\rangle$. As explained, when vertex i is colored, one of λ or α, swapping between the two may or may not preserve feasibility. The control function $f_{i\lambda\alpha}$ ensures that such a move is only allowed when feasibility is guaranteed to be preserved. Hence, for states with $f_{i\lambda\alpha}$=1, such a color swap is guaranteed to preserve feasibility, and $U_{i\lambda\alpha}^{\mathrm{MC}}(\beta)$ acts as a complex rotation in the space spanned by $|x\rangle$ and $|x'\rangle$, where x' is the string derived from swapping the $x_{i\lambda}$ and $x_{i\alpha}$ bits of x. When instead $f_{i\lambda\alpha} = 0$, then $U_{i\lambda\alpha}^{\mathrm{MC}}$ acts trivially. Altogether, the action on classical feasible states is

$$U_{i\lambda\alpha}^{\mathrm{MC}}|x\rangle = \begin{cases} |x\rangle & \text{if } f_{i\lambda\alpha}(x) = 0 \\ & \text{or } x_{i\lambda} = 0 = x_{i\alpha} \\ \cos(\beta)|x\rangle - i\sin(\beta)|x'\rangle & \text{else.} \end{cases} \tag{11}$$

The last case follows from the fact that $(X_{i\lambda}X_{i\alpha} + Y_{i\lambda}Y_{i\alpha})^2$

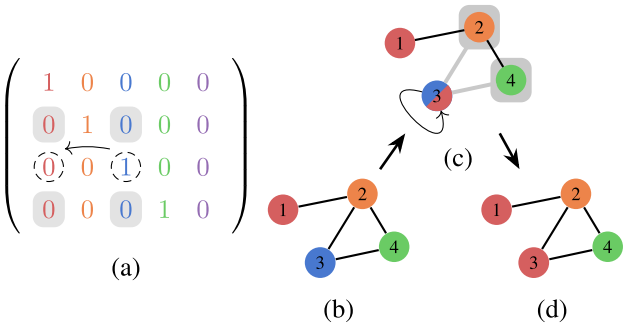


FIGURE 4. Classical action corresponding to a partial controlled color-change mixer from the variable and graph perspectives. An example feasible assignment for a problem with $n = 4$ flights and $k = 5$ gates is depicted as a decision variable matrix in (a) corresponding to the proper coloring of the conflict graph shown in (b). Here, we have $k > n > \chi(G) = 3$. The classical action of the partial mixer $U^{\mathrm{MC}}_{i\lambda\alpha}$ that changes the color of vertex $i = 3$ from blue to red (i.e., $\alpha = 3$ to $\lambda = 1$), and vice versa, is indicated in (a) and (c). The resulting graph is shown in (d). As the neighbors of vertex 3 [indicated by the shaded entries in (a) and (c)] are neither red nor blue, the control condition of the partial mixer is satisfied.

acts proportional to the identity operator on the subspace of states where one of $x_{i\lambda} = 1$ or $x_{i\alpha} = 1$, and as 0 otherwise, which leads to a closed form for (9).

2) COLOR-CHANGE MIXER

We define the *fully connected controlled color-change mixer* as an ordered product of the partial controlled color-change mixers

$$U_{\mathrm{MC}}(\beta) := \prod_{i\in[n]} \prod_{\lambda<\alpha} U^{\mathrm{MC}}_{i\lambda\alpha}(\beta) \tag{12}$$

where a partial mixer corresponding to each possible color change for each possible vertex applied in turn. Here, the order of the partial mixers is selected arbitrarily; different orderings will have only a minor effect on our resource estimates to follow. Since each of the partial controlled color-change mixers preserve feasibility, so does $U_{\mathrm{MC}}(\beta)$ for all values of β. We now show that U_{MC} satisfies the reachability criterion discussed above. Recall we may always replace the mixer U_{MC} with $U^r_{\mathrm{MC}} := (U_{\mathrm{MC}}(\beta))^r$ with cost at most r times that of a single application.

Proposition 1: Let G be a chordal graph on n vertices and x, y be any two proper colorings of G using at most k colors for a fixed $k \geq \chi(G) + 1$, and let $r := 2n^2$. Then, there exists a constant $\beta \in [0, 2\pi]$ such that $|\langle y|U^r_{\mathrm{MC}}|x\rangle| > 0$.

We prove the proposition using results from graph theory which imply that any two proper colorings can be connected by a sequence of $O(n^2)$ classical controlled color-change moves. Clearly, much shorter sequences may suffice to connect specific pairs of feasible assignments.

Proof: Consider the *reconfiguration graph* of G with respect to the (controlled) color change move, constructed by creating a vertex for each proper coloring of G using at most k colors, and creating an edge between two vertices if the corresponding colorings differ only by changing the color of a single vertex. Necessarily, such color changes must satisfy the control condition (10). For chordal graphs, Bonamy *et al.* [32] prove that the reconfiguration graph is necessarily connected if $k \geq \chi(G) + 1$, with graph diameter at most $2n^2$. This implies that any two feasible flight-gate assignments can be connected by a sequence of at most $O(n^2)$ classical controlled color changes.

Next, let $r = 2n^2$ and select β such that $|\sin(\beta)|$ is not 0 or 1 (for example, $\beta = 1/8$). Fix two feasible bitstrings x, y and consider the overall mixer $U^r_{\mathrm{MC}} = (U_{\mathrm{MC}}(\beta))^{2n^2}$. Using (12) to expand the parentheses gives a complete ordered sequence of all the $n\binom{k}{2}$ partial mixers $U^{\mathrm{MC}}_{i\lambda\alpha}(\beta)$, repeated $2n^2$ times, to which repeatedly applying (11) shows that U^r_{MC} acts on $|x\rangle$ to produce a superposition of feasible states connected to x by classical controlled color-change moves. Clearly, this sequence contains all possible length $2n^2$ subsequences of partial mixers $U^{\mathrm{MC}}_{i\lambda\alpha}$. Hence, from the above at least, one such sequence must connect x to y. Moreover, from (11), we see that if another such sequence of equal length exists, then it will add constructively (i.e., carry the same amplitude); furthermore, sequences of different length connecting x and y will carry different polynomials of $\cos(\beta)$ and $\sin(\beta)$, which can only cancel for specific values of β (i.e., on a set of measure 0). Hence, we conclude that there exists β such that $\langle y|U^r_{\mathrm{MC}}|x\rangle \neq 0$ as desired, where the interval $\beta \in [0, 2\pi]$ suffices from (11). ■

The condition $k \geq \chi(G) + 1$ of the proposition applies to all problem instances except the boundary cases with $k = \chi(G)$, where we recall that when $k < \chi(G)$, the instance is easily certified to have no feasible assignments. In particular, observe that $k \geq \chi(G) + 1$ necessarily when $k > n$. Hence, we see that the color-change mixer U_{MC} satisfies the design criteria of [12] for most problem instances (i.e., where $k > \chi(G)$).

3) COLOR-CHANGE-AND-SWAP MIXERS

Mixing operators may be derived from different classical moves for exploring the feasible solutions. Here, we derive an alternative family of mixers for our problem, extending constructions of [12] for the TSP, which may be used to augment the partial controlled color-change mixers in constructing total mixing operators. Whereas we previously considered changing the color of a given vertex, here the partial mixer is derived from the classical move of swapping the assigned colors of two vertices. We then explain how these color-swap partial mixers may be used to augment the color-change mixers of the previous section.

The *partial controlled color-swap mixer* is defined as

$$U^{\mathrm{MS}}_{ij\lambda\alpha}(\beta) := \Lambda_{\bigwedge_{k\in N(i,j)} \bar{x}_{k\lambda}\bar{x}_{k\alpha}}\left(\exp\left(-\mathrm{i}\beta H_{\mathrm{M},ij\lambda\alpha}\right)\right) \tag{13}$$

where $N(i, j) := N(i) \cup N(j) \setminus \{i, j\}$ is the joint neighborhood of the vertices i, j of size $d_{ij} := |N(i, j)|$, and the 4-local Hamiltonian

IEEE Transactions on Quantum Engineering

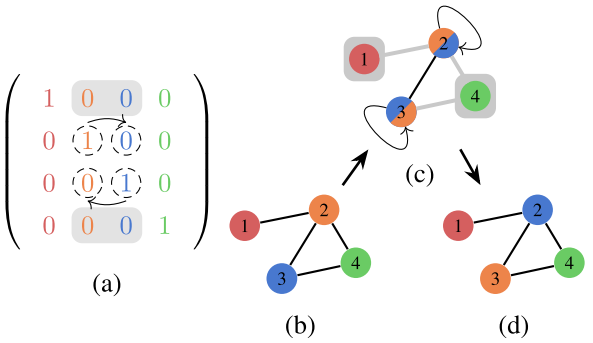


FIGURE 5. Classical action corresponding to a partial controlled color-swap mixer from the variable and graph perspectives. An example feasible assignment for a problem with $n = 4$ flights and $k = 4$ gates is depicted as a decision variable matrix in (a) corresponding to the proper coloring of the conflict graph shown in (b). Here, we have $k = n > \chi(G) = 3$. The classical action of the partial mixer $U^{\mathrm{MS}}_{ij\lambda\alpha}$ that swaps the colors of vertices $i = 2$ and $i = 3$ from blue to orange (i.e., $\alpha = 2$ $\lambda = 3$) is indicated in (a) and (c). The resulting graph is shown in (d). As the neighbors of vertices 2 and 3 [indicated by the shaded entries in in (a) and (c)] are neither blue nor orange, the control condition of the partial mixer is satisfied.

$$H_{\mathrm{M},ij\lambda\alpha} := |0_{i\alpha}1_{i\lambda}1_{j\alpha}0_{j\lambda}\rangle\langle 1_{i\alpha}0_{i\lambda}0_{j\alpha}1_{j\lambda}| + |1_{i\alpha}0_{i\lambda}0_{j\alpha}1_{j\lambda}\rangle\langle 0_{i\alpha}1_{i\lambda}1_{j\alpha}0_{j\lambda}| \tag{14}$$

acts to simultaneously swap the colors of the vertices i and j from λ to α and vice versa. The exponential is controlled by the condition that none of the vertices adjacent to i or j are colored λ nor α (and hence the color swap is guaranteed to preserve feasibility). Note that in the special case $k = \chi(G) = n$, the controls are no longer necessary and the partial controlled color-swap mixer reduces to the TSP mixer as defined in [12] (see Sections II-B2 and III-C4).

For implementation, the Hamiltonian $H_{\mathrm{M},ij\lambda\alpha}$ can be written as a sum of eight mutually commuting 4-local strings of X and Y Pauli operators [12], [31], suppressing the indices

$$\frac{1}{8}(XXXX - YYXX + XYXY + YXYX + XYYX + YXXY - XXYY + YYYY). \tag{15}$$

Fig. 5 illustrates the action of the partial controlled color-swap mixer. The underlying classical color-swap operation described above closely relates to that of Kempe changes [33]–[36], from classical graph algorithms. Given a proper k-coloring and two colors α and β, a Kempe chain is defined as a connected component of the subgraph induced by the vertices colored by α or β. A Kempe change is a total color swap within a Kempe chain. If two colorings can be transformed into each other by a series of Kempe changes, they belong to the same equivalence class, called Kempe class. Moreover, if $k \geq n$, all k-colorings belong to a single Kempe class [33]. In [36], it is shown that at most n Kempe changes are necessary to reach a k-coloring from any other k-coloring, which can be used to explore reachability results similarly to and potentially improving on those of Proposition 1. Indeed, when $\chi(G) = n$, each Kempe chain consist of a single edge, and each possible Kempe change corresponds to a partial-controlled color-swap mixer.

Observe that in the example of Fig. 5, a single color swap move suffices to swap the colors of the indicated pair of vertices, whereas at least three color-change moves are required to obtain the same result. This suggests that the partial mixers $U^{\mathrm{MS}}_{ij\lambda\alpha}$ may lead to improved mixing. However, sequences of $U^{\mathrm{MS}}_{ij\lambda\alpha}$ clearly act on classical basis states as to preserve the total number of vertices with each given color. In particular, this means that they cannot change the number of unused colors (number of unassigned gates). Hence, mixers (and initial states) derived from the partial mixers $U^{\mathrm{MS}}_{ij\lambda\alpha}$ alone will not be able to reach all feasible solutions in general. Nevertheless, for instances with $k > \chi(G)$ generally, Proposition 1 suggests that arbitrary Kempe changes may be implemented with a combination of color-swap and color-change moves.

Therefore, we define the *fully connected controlled color-change-and-swap mixer* as

$$U_{\mathrm{MCS}}(\beta_1, \beta_2) := U_{\mathrm{MS}}(\beta_2)U_{\mathrm{MC}}(\beta_1) \tag{16}$$

where we now allow two mixing angles per stage (though one could always set, e.g., $\beta = \beta_1 = \beta_2$), and where

$$U_{\mathrm{MS}}(\beta) := \prod_{(j,k)\in E}\prod_{\lambda<\alpha} U^{\mathrm{MS}}_{jk\lambda\alpha}(\beta) \tag{17}$$

is the *fully connected controlled color-swap mixer*, defined analogously to U_{MC}. The first product is taken over the edges of the conflict graph $(i, j) \in E$, as motivated by the discussion above (cf. Fig. 5); taking instead all pairs $i < j$ does not affect the resulting worst case resource estimates. As before, the order of the partial mixers may be set arbitrarily. It is obvious that (16) and (17) each preserves feasibility since each of the partial mixers involved preserves feasibility. Moreover, by setting $\beta_2 = 0$, we see that $U_{\mathrm{MCS}}(\beta_1, \beta_2)$ satisfies the conditions of Proposition 1 and so U_{MCS} also satisfies the same reachability criteria, at least for the same set of problem instance where $k > \chi(G)$.

4) SPECIAL CASES

Here, we discuss simplifications of our mixing operator constructions in the special cases $k = \chi(G) = n$ and $\chi(G) = 1$ of Section II-B2, and the limitations of our constructions in the remaining unaddressed cases with $k = \chi(G)$.

First consider the case $k = \chi(G) = n$ which corresponds to a complete conflict graph $G = (V, E) = K_n$. We emphasize that different flight-gate assignment problem instances can result in the same conflict graph. In this case, the set of feasible solutions is the same as that of the TSP, i.e., orderings of n elements. From this perspective, tour positions are given by colors, and the Hamiltonian $H_{\mathrm{M},ij\lambda\alpha}$ of (14) acts either to swap the colors of i and j for basis states $|x\rangle$ where i and j are assigned λ and α, or trivially otherwise. Hence, for this case, we define the *partial TSP mixer* [12], [31] as

$$U^{\mathrm{TSP}}_{ij\lambda\alpha}(\beta) := \exp\left(-\mathrm{i}\beta H_{\mathrm{M},ij\lambda\alpha}\right) \tag{18}$$

which is equal to $U^{\mathrm{MS}}_{ij\lambda\alpha}(\beta)$ but without the control and hence requires reduced resources for implementation. (As the control function of $U^{\mathrm{MS}}_{ij\lambda\alpha}$ always evaluates to true in this case, the two operators act equivalently on the feasible subspace.) Sequences of this partial mixer suffice to connect all feasible states (for $k = \chi(G) = n$) [12]. Hence, for this case, we define the *fully connected TSP mixer* by replacing each $U^{\mathrm{MS}}_{jk\lambda\alpha}$ by $U^{\mathrm{TSP}}_{jk\lambda\alpha}$ in (17) to give

$$U_{\mathrm{TSP}}(\beta) := \prod_{j<k} \prod_{\lambda<\alpha} U^{\mathrm{TSP}}_{jk\lambda\alpha}(\beta). \tag{19}$$

We discuss implementation of U_{TSP} in the next section.

Clearly, $U_{\mathrm{TSP}}(\beta)$ corresponds to $U_{\mathrm{MCS}}(0,\beta) = U_{\mathrm{MS}}(\beta)$ when $k = \chi(G) = n$. Observe that the condition $\chi(G) > n$ of Proposition 1 concerning reachability of the color-change mixer U_{MC} is not satisfied here. Indeed, it is easy to see that U_{MC} acts trivially in this case, and is hence unnecessary, as the control function (10) of each partial color-change mixer always evaluates to false (because there can be no colors unused by a neighbor). Therefore, we see that this special case motivates the introduction of more sophisticated partial mixers such as the color-swap mixers of the previous section.

Next, consider the case when $\chi(G) = 1$, which implies G consists of n isolated vertices, and so flight-gate assignment reduces to unconstrained optimization over n-many k-valued variables. As in this case, the control functions (10) always evaluate to true, the control part of each $U^{\mathrm{MC}}_{i\lambda\alpha}$ can be similarly discarded. Hence, the partial-controlled color-change mixers reduce to the partial XY-mixers (9), which act to independently change the values of each k-dit, and also entail reduced resource requirements (see Section IV-E). For this case, we likewise define the *fully connected XY mixer* as

$$U_{XY}(\beta) := \prod_{i\in[n]} \prod_{\lambda<\alpha} U^{XY}_{i\lambda\alpha}(\beta)\,. \tag{20}$$

Finally, we remark on the remaining set of problem instances where $1 < k = \chi(G) < n$, which are the only cases not covered by the condition of Proposition 1 or the special cases above. (Recall we expect problem instances with $k = \chi(G)$ to be less practically relevant.) We explain how even combined sequences of partial controlled color-change and color-swap mixers may not be guaranteed to connect any two feasible solutions when $k = \chi(G) < n$. Indeed, it is straightforward to construct toy examples for this case that necessarily require mixers involving swaps of progressively higher order (i.e., involving $2, 4, 6, \ldots,$ qubits). We give such an example in Fig. 6; there we have a red-green Kempe chain of size 3 comprised of the induced subgraph involving vertices 1, 2, and 4. In this example, to swap the red vertices to green, and vice versa, it is easily seen that a controlled 6-qubit local (3-vertex-local color swap) is necessary to connect between such feasible states, and hence no combination of the partial color-change and color-swap mixers can suffice

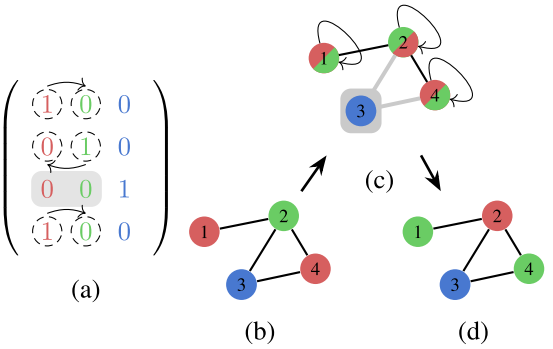


FIGURE 6. Corner case $\chi(G) = k < n$: Example flight-gate assignment problem instance with 4 flights and 3 gates depicted as (a) a matrix and (b) a colored conflict graph with $\chi(G) = 3$. The colors of the red and green vertex may be swapped with a partial mixer derived from extending the color-swap mixer to an operator acting nontrivially on 6 qubits controlled by the neighbors of the Kempe chain as indicated in (a) and (c). The resulting graph is shown in (d).

alone. It is straightforward to extend this example to ones with larger Kempe chains and hence requiring arbitrary order color swaps. We leave as a future research direction to determine whether different problem encodings or different types of classical moves can be used to construct effective mixers for such cases.

IV. CIRCUIT DECOMPOSITION AND RESOURCE ESTIMATES

In this section, we derive bounds to the scaling of the quantum resources required for the QAOA presented in Section III. We show explicit quantum circuit decompositions for our proposed QAOA operators and states. Rather than seeking out optimal circuit compilations, which may be hardware dependent, our focus will be to demonstrate intuitive constructions that yield efficient resource scaling. For the sake of simplicity, and comparability, we select CNOTs and arbitrary single-qubit gates as the universal set of basic quantum gates [37]. This choice aims to be reasonably hardware agnostic; a variety of computationally equivalent universal gate sets exist [38]. Hence, our quantum circuit resource estimates are shown in terms of the number of single-qubit gates $\mathcal{N}_S$ and the number of CNOT (entangling) gates $\mathcal{N}_C$, as well as the overall number of qubits needed $\mathcal{N}_Q$. We derive worst case upper bounds for each quantity. We emphasize that circuit optimizations for specific problem instances may yield significantly reduced resources in practice.

Recall that our problem encoding requires nk qubits. Our constructions take advantage of a small number of scratchpad ancilla qubits as to facilitate simple quantum circuit constructions. By considering ancilla qubit reuse, we show that the overall number of qubits required is $\mathcal{N}_Q = nk + \mathcal{O}(n)$. In practice, a variety of different compilation options with different tradeoffs may be possible [38].

IEEE Transactions on Quantum Engineering

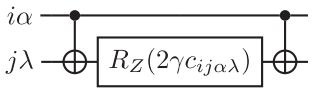
A. PHASE SEPARATION OPERATOR

Consider the phase separation operator $U_{\mathrm{P}}(\gamma)$ of Section III-B, which we write suggestively as

$$e^{-\mathrm{i}\gamma c_0}\prod_{i\alpha}\exp(-\mathrm{i}\gamma c_{i\alpha}Z_{i\alpha})\prod_{ij\alpha\lambda}\exp(-\mathrm{i}\gamma c_{ij\alpha\lambda}Z_{i\alpha}Z_{j\lambda})$$

using the mutual commutativity of the terms in the cost Hamiltonian (7); in particular, the terms of the products can be implemented in any order. Note the global phase term $e^{-i\gamma c_0}$ acts inconsequentially and can be ignored. Since we assume that each flight i has passengers (i.e., $n_i^d + n_i^a > 0$), the coefficients $c_{i\alpha}$ are nonzero for each $i \in V$, $\alpha \in K$. Hence, the first product above can be implemented using at most nk single-qubit rotations. Similarly, each factor in the second product can be decomposed into two CNOTs and one single-qubit rotation using

$$\exp(-\mathrm{i}\gamma c_{ij\alpha\lambda}Z_{i\alpha}Z_{j\lambda}) =$$



To bound the number of nonzero $c_{ij\alpha\lambda}$, which from (2) occurs only between flight pairs with transfer passengers, recall the transfer passenger graph $T = (V, P)$, $P = \{(i, j) \mid n_{ij} > 0\}$ defined in Section II-A. For each edge in P, we have $\sum_{\alpha\leq\lambda} 1 = k(k+1)/2$ corresponding nonzero $c_{ij\alpha\lambda}$, and hence $\frac{1}{2}|P|k(k+1)$ such terms in the second product above.[3] Therefore, the total number of CNOT gates to implement $U_{\mathrm{P}}(\gamma)$ in this way is

$$\mathcal{N}_C = |P|k(k+1)$$

using a number of single-qubit gates

$$\mathcal{N}_S = \frac{1}{2}|P|k(k+1) + nk$$

and not requiring any ancilla qubits. In the worst case of a complete transfer passenger graph, we have $\mathcal{N}_S, \mathcal{N}_C = \mathcal{O}(n^2k^2)$. We emphasize that the input data of Table 1 only affects the resource counts in terms of the number of edges in T; the particular values of the time parameters translate to single-qubit rotation gate angles and do not affect their number.

B. COLOR-CHANGE MIXERS

Here, we consider implementations of the partial and total color-change mixers of Sections III-C1 and III-C2. We derive explicit quantum circuit decompositions into CNOT and single-qubit gates to bound the required resources.

Our decompositions make use of multiqubit Toffoli gates acting on $\ell \geq 3$ qubits. The $\ell = 3$ case gives the standard Toffoli gate that performs a controlled-controlled-NOT operation. An ℓ-qubit Toffoli gate is a bit-flip controlled by $\ell - 1$ qubits. As applying an X gate before and after each control qubit flips the parity of the control line

$$\text{---}\bullet\text{---} \;=\; \text{---}\boxed{X}\text{---}\bullet\text{---}\boxed{X}\text{---}$$

we can always convert between 1 and 0 control lines with a single-qubit gate cost linear in the number of controls. (Moreover, such gates can often be made unnecessary through standard compilation techniques.) Hence, here, we use *multiqubit Toffoli gates* to refer to any such gate independent of the control parities, and we account for any required single-qubit gates as needed.

[3]Note that due to nonzero gate re-entry times $t_{\alpha\alpha} > 0$, we have $c_{ij\alpha\alpha} \neq 0$ in general.

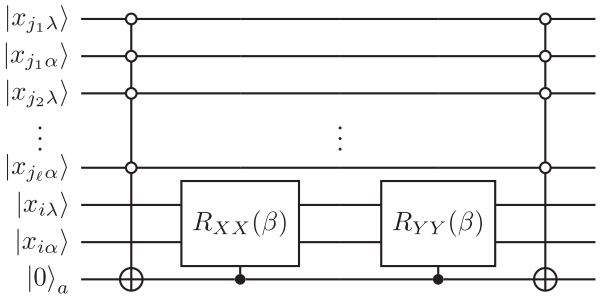


FIGURE 7. Quantum circuit implementing the partial controlled color-change mixer $U^{\mathrm{MC}}_{i\lambda\alpha}(\beta)$ for flight i with conflict graph neighbors $j \in N(i)$. An ancilla qubit is initialized and returned to $|0\rangle$ for reuse. The open circles indicate control by the $|0\rangle$ qubit state.

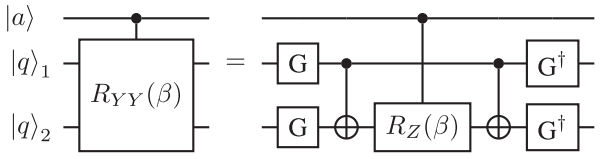


FIGURE 8. Quantum circuit implementing the controlled YY-rotation $\Lambda_a(R_{YY}(\beta)) = \Lambda_a(e^{-\mathrm{i}\beta Y_1 Y_2/2})$. Here, the gate $\mathrm{G} := R_X(\pi/2)$, and $\mathrm{G}^\dagger = R_X(-\pi/2)$. Replacing each G gate with a Hadamard gate $\mathrm{H} = \mathrm{H}^\dagger$ gives instead the controlled XX-rotation $\Lambda_a(R_{XX}(\beta)) = \Lambda_a(e^{-\mathrm{i}\beta X_1 X_2/2})$.

1) PARTIAL-CONTROLLED COLOR-CHANGE MIXER

Consider a single partial-controlled color-change mixer $U^{\mathrm{MC}}_{i\lambda\alpha}(\beta)$ of (8). First, observe that the target unitary (9) (the XY partial mixer) can be decomposed as

$$\begin{aligned} e^{-\mathrm{i}\beta(X_{i\lambda}X_{i\alpha}+Y_{i\lambda}Y_{i\alpha})/2} &= e^{-\mathrm{i}\beta X_{i\lambda}X_{i\alpha}/2}\cdot e^{-\mathrm{i}\beta Y_{i\lambda}Y_{i\alpha}/2} \\ &= R^{i\alpha i\lambda}_{XX}(\beta)\cdot R^{i\alpha i\lambda}_{YY}(\beta) \end{aligned}$$

since $[X_\ell X_{\ell'}, Y_\ell Y_{\ell'}] = 0$. Here, we denote the two-qubit rotation operators $R^{ij}_{AB}(\beta) := \exp(-\mathrm{i}\beta A_i B_j/2)$. For the control part of (8), multiqubit Toffoli gates can be used to compute the control value in an ancilla qubit register [37, Lemma 7.11], as shown in Fig. 7. With this step, we introduce one ancilla qubit and two $(2d_i + 1)$-qubit Toffoli gates with negated controls. The controlled two-qubit rotations in Fig. 7 can each be further decomposed using the circuits shown in Fig. 8 which requires four single-qubit gates, two CNOTs, and one controlled single-qubit rotation.

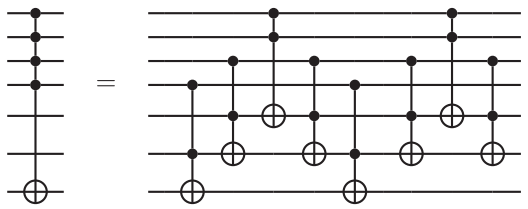


FIGURE 9. Decomposition of a 5-qubit Toffoli gate into eight 3-qubit Toffoli gates while using two ancilla qubits according to [37]. Other decompositions are possible. Each left-hand side control line may be changed to an open circle (negated) by applying an initial and final X gate to the corresponding right-hand side qubit.

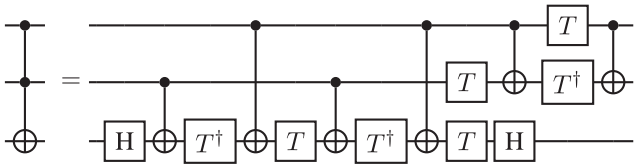
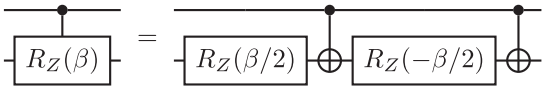


FIGURE 10. Example decomposition of a (3-qubit) Toffoli gate into CNOT and single-qubit gates. Here, $T = \exp(-\mathrm{i}\frac{\pi}{8}Z)$. The 6 CNOT gates are necessary [39].

Next, the controlled Z rotation can be decomposed into two Z rotations and two CNOT gates [37, Lemma 5.4]



(21)

We turn to the decomposition of the two multiqubit Toffoli gates of Fig. 7. As discussed, the two $2d_i$-fold control lines can be converted to positive controls with $4 \cdot 2d_i$ single-qubit gates, which can be reduced to $4d_i$ in number using the fact that the inner pairs of such gates can be canceled for each control line for the particular circuit of Fig. 7. (We emphasize that similar circuit optimization techniques may be applied instancewise to further reduce the overall number of gates required in practice.) According to [37, Lemma 7.2], an $(\ell+1)$-qubit Toffoli gate may be decomposed into $4(\ell-2)$ 3-qubit Toffoli gates using $\ell-2$ ancilla qubits, when $\ell \geq 3$; an example is shown in Fig. 9.[4] Finally, the 3-qubit Toffoli gates can be decomposed into 6 CNOT and 9 single-qubit gates [39] as shown in Fig. 10. We remark that a variety of alternative implementations of standard and multiqubit Toffoli gates are known, with different tradeoffs (see, e.g., [38], [40], [41]).

Combining the above results gives a decomposition of the partial controlled color-change mixer $U^{\mathrm{MC}}_{i\lambda\alpha}(\beta)$ into basic quantum gates to which we now bound the required resources. To this end, we introduce the resource tuple

$$\mathcal{R} := (\mathcal{N}_S, \mathcal{N}_C)$$

of the number of single-qubit gates $\mathcal{N}_S$ and the number of CNOT gates $\mathcal{N}_C$. Assume each vertex i is not isolated, i.e., $d_i > 0$; as isolated vertices require fewer resources this is without loss of generality. Following Fig. 7 gives

$$\mathcal{R}\left(U^{\mathrm{MC}}_{i\lambda\alpha}\right) = 2\mathcal{R}\left((2d_i+1)\text{-qubit negated Toffoli gates}\right) + 2\mathcal{R}\left(\boxed{R_{XX}(\beta)}\right) R_{XX}(\beta) \tag{22}$$

Decomposing the $(2d_i+1)$-qubit Toffoli gates with negated controls (cf. Fig. 9) then yields

$$\mathcal{R}\left((2d_i+1)\text{-qubit negated Toffoli gates}\right) = 4(2d_i-2)\mathcal{R}\left(\text{Toffoli}\right) + 2d_i(1,0), \tag{23}$$

where the first contribution stems from the decomposition of the $(2d_i+1)$-qubit Toffoli, and the second from the $2d_i$ X gates used to flip its control lines. Using Fig. 10, the resources of a Toffoli gate are

$$\mathcal{R}\left(\text{Toffoli}\right) = (9,6). \tag{24}$$

For the second term in (22), from Fig. 8 and (21), we have

$$\mathcal{R}\left(\boxed{R_{XX}(\beta)}\right) = (4,2) + \underbrace{\mathcal{R}\left(\boxed{R_Z(\beta)}\right)}_{(2,2)} = (6,4)$$

Hence, the total quantum gate resources for the partial controlled color-change mixer $U^{\mathrm{MC}}_{i\lambda\alpha}(\beta)$ are at most

$$\begin{aligned}\mathcal{R}\left(U^{\mathrm{MC}}_{i\lambda\alpha}\right) &= 2\{4(2d_i-2)(9,6) + 2d_i(1,0) + (6,4)\} \\ &= (148d_i - 132,\ 96d_i - 88)\,. \end{aligned} \tag{25}$$

This implies the number of single-qubit gates as well as the number of CNOT gates scales linearly with the degree of vertex i in the conflict graph, i.e., $\mathcal{N}_S, \mathcal{N}_C = \mathcal{O}(d_i)$. From Figs. 7 and 9, the total number of ancilla qubits needed is $1 + (2d_i - 2) = 2d_i - 1$.

Note that in the case $d_i = 0$, as controls are unnecessary (cf. Fig. 7), the circuit for $U^{\mathrm{MC}}_{i\lambda\alpha}(\beta)$ instead consists of a single pair of R_{XX} and R_{YY} rotations as in (9). Each rotation can be implemented with 5 single-qubit and 2 CNOT gates (cf. Fig. 8). Hence, the resources for such partial mixers are $\mathcal{N}_S = 10$ and $\mathcal{N}_C = 4$, with no ancilla qubits needed.

[4] When the number of controls is small such that $\ell < 3$, e.g., corresponding to a vertex of a degree 1, then slightly fewer resources are needed; nevertheless, the stated bounds remain true up to constants, which suffices for our purposes in regards to resource scaling.

IEEE Transactions on Quantum Engineering

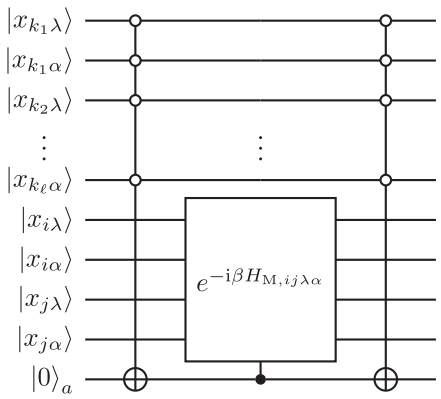


FIGURE 11. Quantum circuit implementing the partial controlled color-swap mixer $U^{\mathrm{MS}}_{ij\lambda\alpha}(\beta)$.

2) FULLY CONNECTED CONTROLLED COLOR-CHANGE MIXER

For the fully connected controlled color-change mixer (12), we use (25) to obtain

$$\mathcal{R}\left(U_{\mathrm{MC}}\right) = \sum_{i\in[n]}\sum_{\lambda<\alpha}\mathcal{R}\left(U^{\mathrm{MC}}_{i\lambda\alpha}\right) = \frac{k(k-1)}{2}(296|E|-132,\ 192|E|-88) \quad (26)$$

where we used that the sum of degrees in the conflict graph $\sum_{i\in[n]} d_i = 2|E|$ is twice the number of edges. Hence, the number of single-qubit gates as well as the number of CNOT gates scale as $\mathcal{N}_S, \mathcal{N}_C = \mathcal{O}(k^2|E|)$. In the worst case, the number of edges scales quadratically with the number of vertices (flights), resulting in a scaling $\mathcal{N}_S, \mathcal{N}_C = \mathcal{O}(k^2n^2)$.

Since we assume that ancilla registers are reused by subsequently applied partial mixers, $2\max_i d_i - 1$ ancilla qubits suffice. This number is bounded from above by the largest vertex degree in the conflict graph. In the worst case, we have $\max_i d_i = n-1$, which gives $nk + \mathcal{O}(n)$ overall qubits.

C. COLOR-CHANGE-AND-SWAP MIXERS

We now present the decomposition and resource estimates of the color-swap mixers covered in Section III-C3.

1) PARTIAL CONTROLLED COLOR-SWAP MIXER

The decomposition of the partial controlled color-swap mixer $U^{\mathrm{MS}}_{ij\lambda\alpha}(\beta)$ from (13) into single-qubit and CNOT gates can be done similarly to that of the partial-controlled color-change mixer (see Fig. 11). Using (15), the controlled $\exp(-\mathrm{i}\beta H_{\mathrm{M},ij\lambda\alpha})$ operation in Fig. 11 can be decomposed into eight controlled 4-local XY-rotations. Each of these controlled rotations can be decomposed into eight single-qubit gates, six CNOT gates, and a controlled R_Z gate. This can be seen by replacing the R_Z gate in the circuit of Fig. 12 by its controlled version. The latter can be decomposed into two single-qubit gates and two CNOTs [cf. (21)]. We have $2d_{ij} = 2|N(i,j)| = 2|N(i)\cup N(j)\setminus\{i,j\}|$ controls in Fig. 11. Similar to Section IV-B1, these controls can

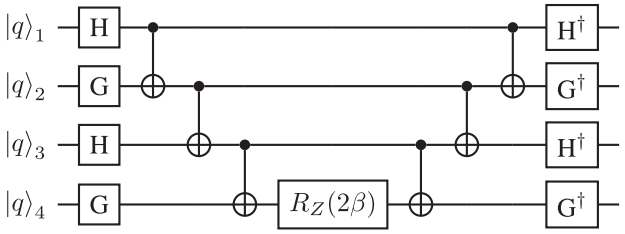


FIGURE 12. Quantum circuit performing the $XYXY$-rotation $R_{X_1Y_2X_3Y_4}(2\beta) = \exp(-i\beta X_1Y_2X_3Y_4)$ on four qubits labeled 1 to 4. Generally, the exponential of any tensor product of four X and Y operators can be implemented by a similar circuit where Hadamard $\mathrm{H} = \mathrm{H}^\dagger$ and $\mathrm{G} = R_X(\pi/2)$ gates have been substituted appropriately, corresponding to which of X or Y acts on each qubit. A controlled version is obtained substituting the R_Z gate for its controlled version as in Fig. 8.

be decomposed into single-qubit and CNOT gates. The resources for a partial controlled color-swap mixer are then

$$\mathcal{R}\left(U^{\mathrm{MS}}_{ij\lambda\alpha}\right) = 2\mathcal{R}\left((2d_{ij}+1)\text{-qubit negated Toffoli gates}\right) + \mathcal{R}\left(\boxed{e^{(-\mathrm{i}\beta H_{\mathrm{M},ij\lambda\alpha})}}\right).$$

From (23) and (24) we have

$$\mathcal{R}\left((2d_{ij}+1)\text{-qubit negated Toffoli gates}\right) = 4(2d_{ij}-2)(9,6) + 2d_{ij}(1,0)$$

and as discussed above and in Section III-C3, the controlled rotation $\exp(-\mathrm{i}\beta H_{\mathrm{M},ij\lambda\alpha})$ can be decomposed into eight controlled 4-local rotations of the same cost. Hence

$$\mathcal{R}\left(\boxed{e^{(-\mathrm{i}\beta H_{\mathrm{M},ij\lambda\alpha})}}\right) = 8\mathcal{R}\left(\boxed{e^{(-\mathrm{i}\beta XXYY)}}\right) = 8((8,6)+(2,2)) = (80,64).$$

Putting this together gives the partial controlled color-swap mixer resource estimate

$$\mathcal{R}(U^{\mathrm{MS}}_{ij\lambda\alpha}) = (\mathcal{N}_S, \mathcal{N}_C) = (\,74d_{ij}+8,\ 48d_{ij}+16\,). \quad (27)$$

From Figs. 9 and 11, the total number of ancilla qubits needed is $1 + (2d_{ij}-2) = 2d_{ij}-1$.

2) TSP MIXER

As we discussed in Section III-C4, no controls at all are necessary in the case $k = n = \chi(G)$ and the partial-controlled color-swap mixer reduces to the partial TSP mixer (18). The resources for such a mixer are given by considering the eight circuits shown in Fig. 12 that correspond to the terms of (15). This yields $\mathcal{R}(U^{\mathrm{TSP}}_{ij\lambda\alpha}) = (72,48)$ and no ancilla qubits. For

the full TSP mixer (19), we have $\mathcal{R}(U_{\mathrm{TSP}}) = n(n-1)k(k-1)\cdot(18, 12)$.

3) FULLY CONNECTED CONTROLLED COLOR-SWAP MIXER

According to (27), the resources for the fully connected controlled color-swap mixer (17) read

$$\begin{aligned}\mathcal{R}(U_{\mathrm{MS}}) &= \sum_{(i,j)\in E}\sum_{\lambda<\alpha}\mathcal{R}\left(U^{\mathrm{MS}}_{ij\lambda\alpha}\right)\\ &= \sum_{(i,j)\in E}\frac{k(k-1)}{2}\cdot(76d_{ij}+8, 48d_{ij}+16)\\ &= \frac{k(k-1)}{2}(76D+8, 48D+16)\end{aligned}$$

where we introduced $D := \sum_{(i,j)\in E} d_{ij} = \sum_{(i,j)\in E} |N(i)\cup N(j)\setminus\{i,j\}|$. For upper bounds, we consider

$$\begin{aligned}D &= \sum_{(i,j)\in E}|N(i)\cup N(j)\setminus\{i,j\}|\\ &\le \sum_{(i,j)\in E}(d_i+d_j-2) = -2|E| + \sum_{i=1}^{n} d_i^2\\ &\le -2|E| + 2|E|(n-1) = 2|E|(n-2)\end{aligned}$$

where d_i is the degree of node i, and we have used the identities $\sum_{(i,j)\in E}(d_i+d_j) = \sum_i d_i^2$, $d_i \le n-1$, and $\sum_i d_i = 2|E|$. Note that tighter but more complicated bounds for D may be derived as in [42]. Hence

$$\mathcal{N}_S, \mathcal{N}_C = \mathcal{O}(nk^2|E|) \tag{28}$$

which in the worst case scales as $\mathcal{O}(n^3k^2)$. The number of ancilla qubits used is $2\max_{(i,j)} d_{ij} - 1$, which is $nk + \mathcal{O}(n)$ in the worst case.

4) FULLY CONNECTED CONTROLLED COLOR-CHANGE-AND-SWAP MIXER

The resources for the fully connected controlled color-change-and-swap mixer (16) are given by

$$\mathcal{R}(U_{\mathrm{MCS}}) = \mathcal{R}(U_{\mathrm{MC}}) + \mathcal{R}(U_{\mathrm{MS}}). \tag{29}$$

Since the scaling behavior is dominated by the second part, the resource estimates (28) also hold for this mixer.

D. INITIAL STATE

The unitary U_x for creating a classical feasible state $|x\rangle$ encoded in nk qubits requires n Pauli-X gates. Therefore, $\mathcal{R}(U_x) = (n, 0)$, This state suffices as a QAOA initial state.

However, we assume that an initial feasible superposition state is used, which may be desirable as discussed in Section III-A. From (6), the resources for preparation of the state $U_{\mathrm{init}}(\beta_0)|00\ldots0\rangle$ are

$$\mathcal{R}(U_{\mathrm{init}}) = r_0\mathcal{R}(U_{\mathrm{M}}) + (n, 0)$$

TABLE 3. Resource Scaling (Order of) Estimates for the QAOA Operators, and Overall Ansatze for Which $\mathcal{N}_Q = nk + \mathcal{O}(n)$ Qubits Suffice

operator	$\mathcal{N}_C$	$\mathcal{N}_S$
$U_{\mathrm{P}}(\gamma)$	$k^2\|P\|$	$k^2\|P\| + nk$
$U_{\mathrm{MC}}(\beta)$	$k^2\|E\|$	$k^2\|E\|$
$U_{\mathrm{MCS}}(\beta_1,\beta_2)$	$nk^2\|E\|$	$nk^2\|E\|$
QAOA$_p$–MC	$pk^2\|P\| + (p+1)k^2\|E\|$	$pk^2\|P\| + (p+1)k^2\|E\|$
(worst-case)	pk^2n^2	pk^2n^2
QAOA$_p$–MCS	$pk^2\|P\|+(p+1)k^2n\|E\|$	$pk^2\|P\|+(p+1)k^2n\|E\|$
(worst-case)	pk^2n^3	pk^2n^3

Initial state preparation utilizes an additional mixing operator application that results in the $(p+1)$ factors. Here, we assume that each mixing stage uses $r = \mathcal{O}(1)$ repetitions, and for simplicity, the overall estimates assume $|P| >> n/k$.

where U_{M} is the fully connected controlled color-change mixer (26) or the fully connected controlled color-change-and-swap mixer (29), or the special case mixers U_{TSP} or U_{XY} when applicable, depending on which one is chosen for the ansatz. We assume that the same mixer is used for the initial state, applied in iteration $r_0 = O(1)$ times; alternative selection of an r_0 that scales with n is possible and affects the resource estimates proportionately. Hence, the resource estimates for initial state preparation corresponds to the resources for one additional mixing round with cost as derived above.

E. OVERALL QUANTUM RESOURCES

Combining the resource estimates for the full mixers, phase separation operator, and initial state preparation yields the resource scaling summarized in Table 3.The resource scaling estimates are determined by the connectivity of the conflict graph G and passenger transfer graph T, and scale linearly with the QAOA level p and polynomially with the number of flights n and number of gates k, even in the worst case, for both the QAOA$_p$–MC and QAOA$_p$–MCS variants.

Our resource estimates exhibit different cost scaling in different problem parameter regimes. For families of problem instances where $\chi(G) < k = \mathcal{O}(1)$ (i.e., when the number of airport gates is fixed), the required quantum gates for QAOA$_p$–MC scale as $\mathcal{O}(pn^2)$. On the other hand, if k scales with n, this becomes $\mathcal{O}(pn^4)$. For the special cases $k = n = \chi(G)$ and $\chi(G) = 1$, we showed that the gates costs $\mathcal{N}_S$, $\mathcal{N}_C$ are constant for the corresponding partial mixers. Constructing full mixers according to (12) and (17) yields $\mathcal{N}_S$, $\mathcal{N}_C = \mathcal{O}(n^4)$ for the TSP mixer in the $k = n = \chi(G)$ case, and $\mathcal{N}_S$, $\mathcal{N}_C = \mathcal{O}(k^2n)$ for the XY mixer in the $\chi(G) = 1$ case. Note, however, in general, that the phase separation operator may require a number of gates that scales as $\mathcal{O}(k^2n^2)$, which follows directly from the number of terms in the cost Hamiltonian (i.e., from the structure of the transfer passenger graph). Hence, we see that the mixer and overall QAOA resource estimates are commensurate with those of the phase operator, which yields a relatively balanced allocation of quantum resources between the phase and mixing stages of the ansatz.

V. DISCUSSION

In this work, we presented QAOA, including the design of novel mixing operators, for the flight-gate assignment problem. Our results extend the constructions of [12] to a new class of problems. By leveraging classical results from graph theory, we demonstrated that efficient initial state preparation and exploration of the entire feasible subspace are possible with our constructions. For each component of our ansatz, we showed explicit decompositions into basic quantum gates, with resulting resource estimates that scale as low-degree polynomials in the problem size. As explained, our constructions directly extend to more general constrained combinatorial optimization problems where feasible solutions correspond to the proper vertex colorings of a chordal or otherwise suitable graph (e.g., graphs of bounded treewidth [43]).

Indeed, the methodology and constructions of this article seek to give a prototype that may be generalized to other scheduling and optimization problems in transportation, aerospace, and beyond; for example, train routing problems with feasible states corresponding to independent sets of a conflict graph [44]. Note that such constructions rely on the ability to efficiently find and prepare a feasible initial state, which depends on the problem constraints and underlying conflict graph [12]. Nevertheless, the temporal or spatial character of many real-world problems suggests that similar constructions may be possible for a variety of problems with important practical applications.

Future work, indubitably, should include the analysis and empirical study of algorithm performance, and comparison to classical algorithms, with the eventual goal of experiments on real quantum gate-model hardware. In particular, such experiments would facilitate direct performance comparison with results obtained using a quantum annealer [13]. Such investigations are beyond the scope of this work. As mentioned, several particular aspects of our constructions can be investigated and potentially optimized in practice, such as the selection of the initial state, the term ordering of partial mixers within each total mixer, and the number of total mixer repetitions within each mixing stage. Generally, for paradigms such as QAOA, a variety of important aspects remain to be more thoroughly investigated [12], including effective parameter setting strategies, overall algorithm performance, and the tradeoffs with increasing ansatz circuit depth. We are optimistic that the availability of increasingly powerful quantum gate-model devices will provide valuable insights in these directions.

ACKNOWLEDGMENT

This work was initiated at the DLR Workshop on Quantum Computing for Applications in Science and Industry in Cologne, Germany, and continued at subsequent visits of Tobias Stollenwerk at the NASA Ames Research Center. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Office of the Director of National Intelligence, Intelligence Advanced Research Projects Activity, AFRL, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purpose notwithstanding any copyright annotation thereon.

REFERENCES

[1] S. H. Kim, E. Feron, J.-P. Clarke, A. Marzuoli, and D. Delahaye, "Airport gate scheduling for passengers, aircraft, and operations," *J. Air Transp.*, vol. 25, no. 4, pp. 109–114, Oct. 2017, doi: 10.2514/1.D0079.

[2] D. Marx, "Graph colouring problems and their applications in scheduling," *Periodica Polytechnica Elect. Eng.*, vol. 48, no. 1/2, pp. 11–16, 2004. [Online]. Available: https://pp.bme.hu/ee/article/view/926

[3] "Imaging acquisition planning for earth observation satellites with a quantum annealer," 2020, *arXiv:2006.09724*. [Online]. Available: https://arxiv.org/abs/2006.09724

[4] M. Gamache, A. Hertz, and J. O. Ouellet, "A graph coloring model for a feasibility problem in monthly crew scheduling with preferential bidding," *Comput. Oper. Res.*, vol. 34, no. 8, pp. 2384–2395, 2007, doi: 10.1016/j.cor.2005.09.010.

[5] A. Eisenblätter, M. Grötschel, and A. M. C. A. Koster, "Frequency planning and ramifications of coloring," *Discussiones Mathematicae, Graph Theory*, no. 22, pp. 51–88, 2002, doi: 10.7151/dmgt.1158.

[6] M. Javedankherad, Z. Zeinalpour-Yazdi, and F. Ashtiani, "Content placement in cache networks using graph coloring," *IEEE Syst. J.*, vol. 14, no. 3, pp. 3129–3138, Sep. 2020, doi: 10.1109/JSYST.2020.2978105.

[7] D. Venturelli, D. J. J. Marchand, and G. Rojo, "Quantum annealing implementation of job-shop scheduling," Jun. 2015, *arXiv:1506.08479*. [Online]. Available: http://arxiv.org/abs/1506.08479

[8] E. G. Rieffel, D. Venturelli, B. O'Gorman, M. B. Do, E. M. Prystay, and V. N. Smelyanskiy, "A case study in programming a quantum annealer for hard operational planning problems," *Quantum Inf. Process.*, vol. 14, no. 1, pp. 1–36, 2015, doi: 10.1007/s11128-014-0892-x.

[9] A. Fabrikant and T. Hogg, "Graph coloring with quantum heuristics," in *Proc. 18th Nat. Conf. Aritif. Intell.*, 2002, pp. 22–27. [Online]. Available: https://www.aaai.org/Library/AAAI/2002/aaai02-004.php

[10] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," 2014, *arXiv:1411.4028*. [Online]. Available: https://arxiv.org/abs/1411.4028

[11] F. Arute *et al.*, "Quantum approximate optimization of non-planar graph problems on a planar superconducting processor," 2020, *arXiv:2004.04197*. [Online]. Available: https://arxiv.org/abs/2004.04197

[12] 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, Feb. 2019, Art. no. 34, doi: 10.3390/a12020034.

[13] T. Stollenwerk, E. Lobe, and M. Jung, "Flight gate assignment with a quantum annealer," in *Proc. 1st Int. Workshop Quantum Technol. Optim. Problems*, no. 9, Mar. 2019, doi: 10.1007/978-3-030-14082-3_9.

[14] T. Stollenwerk *et al.*, "Quantum annealing applied to de-conflicting optimal trajectories for air traffic management," *IEEE Trans. Intell. Trans. Syst.*, vol. 21, no. 1, pp. 285–297, Jan. 2020, doi: 10.1109/TITS.2019.2891235.

[15] Z. Wang, N. C. Rubin, J. M. Dominy, and E. G. Rieffel, "*XY*-mixers: Analytical and numerical results for the quantum alternating operator ansatz," *Phys. Rev. A*, vol. 101, no. 1, 2020, Art. no. 012320, doi: 10.1103/PhysRevA.101.012320.

[16] I. Hen and F. M. Spedalieri, "Quantum annealing for constrained optimization," *Phys. Rev. Appl.*, vol. 5, Mar. 2016, Art. no. 034007, doi: 10.1103/PhysRevApplied.5.034007.

[17] I. Hen and M. S. Sarandy, "Driver Hamiltonians for constrained optimization in quantum annealing," *Phys. Rev. A*, vol. 93, no. 6, 2016, Art. no. 062312, doi: 10.1103/PhysRevA.93.062312.

[18] R. Mangoubi and D. F. Mathaisel, "Optimizing gate assignments at airport terminals," *Transp. Sci.*, vol. 19, no. 2, pp. 173–188, 1985, doi: 10.1287/trsc.19.2.173.

[19] A. Haghani and M.-C. Chen, "Optimizing gate assignments at airport terminals," *Transp. Res. A, Policy Pract.*, vol. 32, no. 6, pp. 437–454, 1998, doi: 10.1016/S0965-8564(98)00005-6.

[20] M. R. Garey and D. S. Johnson, *Computers and Intractability*, vol. 29. New York, NY, USA: W. H. Freeman, 2002. [Online]. Available: https://dl.acm.org/doi/book/10.5555/578533

[21] H. Ding, A. Lim, B. Rodrigues, and Y. Zhu, "New heuristics for over-constrained flight to gate assignments," *J. Oper. Res. Soc.*, vol. 55, no. 7, pp. 760–768, Jul. 2004, doi: 10.1057/palgrave.jors.2601736.

[22] M. Marinelli, M. Dell'Orco, and D. Sassanelli, "A metaheuristic approach to solve the flight gate assignment problem," *Transp. Res. Procedia*, vol. 5, pp. 211–220, 2015, doi: 10.1016/j.trpro.2015.01.013.

[23] C.-H. Cheng, S. C. Ho, and C.-L. Kwan, "The use of meta-heuristics for airport gate assignment," *Expert Syst. Appl.*, vol. 39, no. 16, pp. 12430–12437, 2012, doi: 10.1016/j.eswa.2012.04.071.

[24] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*. Amsterdam, The Netherlands: Elsevier, 2004. [Online]. Available: https://www.elsevier.com/books/algorithmic-graph-theory-and-perfect-graphs/golumbic/978-0-12-289260-8

[25] D. Fulkerson and O. Gross, "Incidence matrices and interval graphs," *Pac. J. Math.*, vol. 15, no. 3, pp. 835–855, 1965, doi: 10.2140/PJM.1965.15.835.

[26] N.-W. Lin, "Approximating the chromatic polynomial of a graph," in *Proc. Int. Workshop Graph-Theoretic Concepts Comput. Sci.* 1993, pp. 200–210, doi: 10.1007/3-540-57899-4_53.

[27] D. J. Rose, R. E. Tarjan, and G. S. Lueker, "Algorithmic aspects of vertex elimination on graphs," *SIAM J. Comput.*, vol. 5, no. 2, pp. 266–283, 1976, doi: 10.1137/0205021.

[28] Z. Wang, S. Hadfield, Z. Jiang, and E. G. Rieffel, "Quantum approximate optimization algorithm for maxcut: A fermionic view," *Phys. Rev. A*, vol. 97, no. 2, 2018, Art. no. 022304, doi: 10.1103/PhysRevA.97.022304.

[29] Z. Jiang, E. G. Rieffel, and Z. Wang, "Near-optimal quantum circuit for Grover's unstructured search using a transverse field," *Phys. Rev. A*, vol. 95, no. 6, 2017, Art. no. 062317, doi: 10.1103/PhysRevA.95.062317.

[30] S. Hadfield, "On the representation of Boolean and real functions as Hamiltonians for quantum computing," 2018, *arXiv:1804.09130*. [Online]. Available: https://arxiv.org/abs/1804.09130

[31] S. Hadfield, "Quantum algorithms for scientific computing and approximate optimization," Ph.D. dissertation, Dept. Comput. Sci., Columbia Univ., New York, NY, USA, 2018, *arXiv:1805.03265*. [Online]. Available: https://arxiv.org/abs/1805.03265

[32] M. Bonamy, M. Johnson, I. Lignos, V. Patel, and D. Paulusma, "Reconfiguration graphs for vertex colourings of chordal and chordal bipartite graphs," *J. Combinatorial Optim.*, vol. 27, no. 1, pp. 132–143, Jan. 2014, doi: 10.1007/s10878-012-9490-y.

[33] M. L. Vergnas and H. Meyniel, "Kempe classes and the Hadwiger conjecture," *J. Combinatorial Theory, B*, vol. 31, no. 1, pp. 95–104, 1981, doi: 10.1016/S0095-8956(81)80014-7.

[34] R. Lewis, "Graph coloring and recombination," in *Springer Handbook of Computational Intelligence*. New York, NY, USA: Springer, 2015, pp. 1239–1254, doi: 10.1007/978-3-662-43505-2_63.

[35] N. Nishimura, "Introduction to reconfiguration," *Algorithms*, vol. 11, no. 4, 2018, Art. no. 52, doi: 10.3390/a11040052.

[36] M. Bonamy *et al.*, "Diameter of colorings under Kempe changes," in *Proc. Int. Comput. Combinatorics Conf.*, 2019, pp. 52–64, doi: 10.1007/978-3-030-26176-4_5.

[37] A. Barenco *et al.*, "Elementary gates for quantum computation," *Phys. Rev. A*, vol. 52, pp. 3457–3467, Nov. 1995, doi: 10.1103/PhysRevA.52.3457.

[38] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*. Cambridge U.K.: Cambridge Univ. Press, 2000, doi: 10.1017/CBO9780511976667.

[39] V. V. Shende and I. L. Markov, "On the CNOT-cost of TOFFOLI gates," *Quantum Inf. Comput.*, vol. 9, no. 5, pp. 461–486, May 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=2011791.2011799

[40] M. Saeedi and M. Pedram, "Linear-depth quantum circuits for n-qubit Toffoli gates with no ancilla," *Phys. Rev. A*, vol. 87, no. 6, 2013, Art. no. 062318, doi: 10.1103/PhysRevA.87.062318.

[41] Y. He, M.-X. Luo, E. Zhang, H.-K. Wang, and X.-F. Wang, "Decompositions of n-qubit Toffoli gates with linear circuit complexity," *Int. J. Theor. Phys.*, vol. 56, no. 7, pp. 2350–2361, 2017, doi: 10.1007/s10773-017-3389-4.

[42] K. C. Das, "Maximizing the sum of the squares of the degrees of a graph," *Discr. Math.*, vol. 285, no. 1, pp. 57–66, 2004, doi: 10.1016/j.disc.2004.04.007.

[43] M. Bonamy and N. Bousquet, "Recoloring bounded treewidth graphs," *Electron. Notes Discrete Math.*, vol. 44, pp. 257–262, 2013, doi: 10.1016/j.endm.2013.10.040.

[44] G. Caimi, F. Chudak, M. Fuchsberger, M. Laumanns, and R. Zenklusen, "A new resource-constrained multicommodity flow model for conflict-free train routing and scheduling," *Transp. Sci.*, vol. 45, no. 2, pp. 212–227, 2011, doi: 10.1287/trsc.1100.0349.