CONTENTS

LIST OF TABLES

LIST OF FIGURES

Quantum Annealing for Air Traffic Management

Tobias Stollenwerk
German Aerospace Center, Linder Höhe, 51147 Cologne, Germany

Bryan O'Gorman
University of California, Berkeley
NASA Ames Research Center Quantum Artificial Intelligence Laboratory (QuAIL), Mail Stop 269-1, 94035 Moffett Field CA and Stinger Ghaffarian Technologies Inc., 7701 Greenbelt Rd., Suite 400, Greenbelt, MD 20770

Davide Venturelli
NASA Ames Research Center Quantum Artificial Intelligence Laboratory (QuAIL), Mail Stop 269-1, 94035 Moffett Field CA and USRA

Eleanor G. Rieffel
NASA Ames Research Center Quantum Artificial Intelligence Laboratory (QuAIL), Mail Stop 269-1, 94035 Moffett Field CA

Salvatore Mandrà
NASA Ames Research Center Quantum Artificial Intelligence Laboratory (QuAIL), Mail Stop 269-1, 94035 Moffett Field CA and Stinger Ghaffarian Technologies Inc., 7701 Greenbelt Rd., Suite 400, Greenbelt, MD 20770

Olga Rodionova, Hok K. Ng and Banavar Sridhar
tbd

(Dated: May 19, 2017)

In this paper we present the mapping of air traffic management (ATM) problem on quadratic unconstrained boolean optimization (QUBO) problem. After the representation of the ATM problem in terms of a conflict graph, where nodes of the graph represent flights and edges represent a potential conflict between flights, we proceed by discretize the ATM problem and then mapping it in binary variables. As part of our study, we tested the QUBO formulation of the ATM problem using both classical solvers and the D-Wave 2X quantum chip.

I. INTRODUCTION

There is an overall increase in air traffic over the last decades and this trend is believed to continue. As a result, the air traffic control system is increasingly under pressure. The current system works with pre-planned routes for the flights and more or less manual conflict avoidance with the help of air traffic controllers. With the limited airspace available, novel approaches are necessary to meet the demand of even more air traffic in the coming decades. A promising approach to solve this problem is using wind-optimal trajectories beyond the predefined air traffic network [1]. However, the conflicts in the wind-optimal trajectories need to be avoided, or "deconflicted" [2].

Quantum annealing is a promising computational method which became increasingly important in recent years. This development is driven also by first commercially available quantum annealing device by the company D-Wave Systems. In addition to studying the fundamental properties of quantum annealing, it is imperative to find possible real world application for this technology. Hard operational planning problems are a promising candidate for the latter [3–5].

In this work, we investigate the feasibility of applying quantum annealing to the deconfliction of wind-optimal trajectories. In the course of this, we use wind-optimal trajectory data to extract realistic problem instances. To be amenable to a D-Wave Quantum annealer, the problem has to be formulated as a quadratic unconstrained binary optimization (QUBO) problem. We demonstrate the mapping of deconfliction problem to a QUBO formulation. Since restrictions to the configurations space are necessary for the reformulation of the problem as a QUBO, we investigate the influence of this restriction on the solution quality. Moreover, we study the embeddability and solution quality using both classical solvers and quantum annealing runs.

II. PROBLEM SPECIFICATION

The basic input of the deconflicting problem is a set of ideal flight trajectories (space-time paths). These ideal trajectories are specified by the individual flight operators. Each ideal trajectory represents some independent optimization from the operator's perspective, especially minimizing fuel costs given expected wind conditions between the desired origin and destination at the desired times; for this reason, they are called the "wind-optimal" trajectories. Because of the number of such trajectories and the correlation between them, these trajectories are likely to conflict; that is, two or more aircraft are likely to get dangerously close to each other if their ideal trajectories are followed without modification. The goal thus is to modify the trajectories to avoid such conflicts.

In theory, the configuration space consists of all physically realistic trajectories; in practice, computational bounds constrain us to consider certain perturbations of the ideal trajectories. The simplest such perturbation is a departure delay, which is the main focus of the present work. Previous work [2] additionally considered a global perturbation by which a trajectory is sinusoidally shifted parallel to the Earth's surface. We focus instead on local perturbations to the trajectories, in which a modification to the trajectory is parameterized by some choice of active maneuvers near a potential conflict; such a modification does not affect the preceding part of the trajectory and only affects the subsequent part by the additional delay it introduces.

A full accounting of the cost of such modifications would take into account the cost of departure delays, the change in fuel cost due to perturbing the trajectories, the relative importance of each flight, and many other factors. As in previous work, we consider only the total, unweighted arrival delay, aggregated equally over all of the flights.

Formally, each ideal trajectory $\mathbf{x}_i = (x_{i,t})_{t=t_{i,0}}^{t_{i,1}}$ is specified as a time-discretized path from the departure point $x_{i,t_{i,0}}$ at time $t_{i,0}$ to the arrival point $x_{i,t_{i,1}}$ at time $t_{i,1}$. For each flight i, the geographical coordinates $x_{i,t}$ (as latitude, longitude, and altitude) are specified at every unit of time (i.e. one minute) between $t_{i,0}$ and $t_{i,1}$; we call this interval $T_i = (t_{i,0}, t_{i,0}+1, \dots, t_{i,1})$.

For notational simplicity, suppose momentarily that each trajectory $\mathbf{x}_i$ is modified only by introducing delays between time steps. Let $\delta_{i,t}$ be the accumulated delay of flight i at the time that it reaches the point $x_{i,t}$, and let $\delta^*_{i,t}$ be the maximum such delay.

A pair of flights (i,j) are in conflict with each other if any pair of points from their respective trajectories is in conflict. (The trajectories are reasonably assumed to be sufficiently time-resolved so that if the continuously interpolated trajectories conflict than there is a pair of discrete trajectory points that conflict.) A pair of trajectory points $(x_{i,s}, x_{j,t})$ conflict if their spatial and temporal separations are both within the respective mandatory separation standards Δ_x and Δ_t (i.e. 3 nautical miles and 3 minutes):

$$\|x_{i,s} - x_{j,t}\| < \Delta_x \text{ and } |(s+\delta_{i,s}) - (t+\delta_{j,t})| < \Delta_t \tag{1}$$

. The latter condition can be met for some $(\delta_{i,s}, \delta_{j,t}) \in [0, \delta^*_{i,s}] \times [0, \delta^*_{j,t}]$ if and only if

$$\max\left\{\delta^*_{i,s}, \delta^*_{j,t}\right\} + \Delta_t > |s-t|, \tag{2}$$

in which case we call the pair of trajectory points *potentially* conflicting. The set C of such pairs of potentially conflicting trajectory points contains strongly correlated clusters. To simplify the constraints, we enumerate all such clusters and refer to them simply as *the* conflicts. That is, we partition the potentially conflicting pairs of trajectory points into disjoint sets,

$$C = \bigcup_k C_k, \tag{3}$$

such that if $\{(i,s),(j,t)\}, \{(i',s'),(j',t')\} \in C_k$ for some k then $i = i' < j = j'$ and for all $s'' \in [\min\{s,s'\}, \max\{s,s'\}]$ there exists some $t'' \in [\min\{t,t'\}, \max\{t,t'\}]$ such that $\{(i,s''),(j,t'')\} \in C_k$ and vice versa. Thus every conflict k is associated with a pair of flights $I_k = \{i,j\}$. Let $K_i = \{k | i \in I_k\}$ be the set of conflicts to which flight i is associated, N_{c} the number of conflicts, and N_{f} the number of flights.

Having identified disjoint sets of conflicts, we relax the supposition that the trajectory modifications only introduce delays between time steps. Instead, we consider modifications to the trajectories that introduce delays local to particular conflicts. Specifically, the configuration space consists of the departure delays $\mathbf{d} = (d_i)_{i=1}^{N_{\mathrm{f}}}$ and the set of local maneuvers $\mathbf{a_k} = (\mathbf{a}_k)_k$, where $\mathbf{a}_k$ represents some parameterization of the local maneuvers used to avoid conflict k. Let $d_{i,k}(\mathbf{d}, \mathbf{a_k})$ be the delay introduced to flight i at conflict k, as a function of the departure delays and local maneuvers. With this notation, we can write the total delay as

$$D = \sum_{i=1}^{N_{\mathrm{f}}} \left(d_i + \sum_{k \in K_i} d_{i,k} \right). \tag{4}$$

This is the quantity we wish to minimize subject to avoiding all potential conflicts.

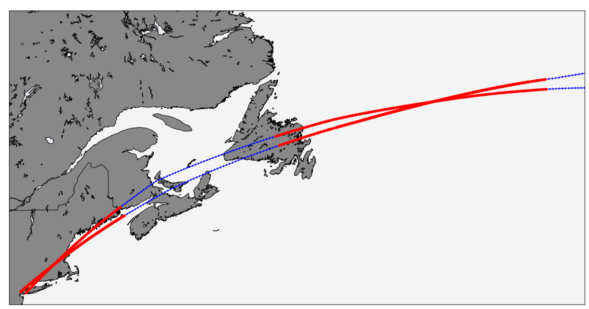

FIG. 1. Example of two parallel potential conflicts between two transatlantic flights starting from the east cost of the USA.

A conflict can be avoided locally by introducing earlier delays differentially, thereby increasing the temporal separation; by some active maneuver of one or both of the flights; or by some combination thereof. We focus on the former case. Let

$$D_{i,k} = d_i + \sum_{k' \in K_{i,k}} d_{i,k'} \tag{5}$$

be the accumulated delay of flight i by the time it reaches conflict k, where $K_{i,k} = \{k' \in K_i | k' < k\}$. In this case,

$\delta_{i,s} = D_{i,k}$ for all s associated with flight i in conflict k. We assume that the set of conflicts K_i associated with flight i is indexed in temporal order, i.e. if $k' < k$ and $k, k' \in K_i$, then flight i reaches conflict k' before conflict k. The pairs of conflicting trajectory points associated with conflict k are given by

$$T_k = \{(s,t) | \{(i,s),(j,t)\} \in C_k, i < j\}\,. \tag{6}$$

Thus the potential conflict is avoided only if

$$D_k = D_{i,k} - D_{j,k} \notin B_k \tag{7}$$

where

$$B_k = \bigcup_{(s,t)\in T_k} (-\Delta_t + t - s, \Delta_t + t - s) = [\Delta_k^{\min}, \Delta_k^{\max}], \tag{8}$$

$$\Delta_k^{\min} = 1 - \Delta_t + \min_{(s,t)\in T_k} \{t - s\}, \tag{9}$$

$$\Delta_k^{\max} = \Delta_t - 1 + \max_{(s,t)\in T_k} \{t - s\}. \tag{10}$$

In the remainder of this paper, we focus on the restricted problem in which only departure delays are allowed. In this simplified case, the configuration space is simply $\mathbf{d} = (d_i)_{i=1}^{N_\mathrm{f}}$, the cost function simply $D = \sum_{i=1}^{N_\mathrm{f}} d_i$, and the constraints simply $d_i - d_j \notin B_k$ for all k.

III. INSTANCES

To assess our methods on realistic instances of the problem, we use the actual wind-optimal trajectories for transatlantic flights on July 29, 2012, as was done in previous work [2]. In these trajectories, each flight has a constant (cruising) altitude and constant speed, to within (classical) machine precision, though our methods generalize to instances without these special properties.

One perspective into the nature of an instance of the deconflicting problem is the *conflict graph*, whose vertices correspond to flights and which has an edge between a pair of vertices if there is at least one potential conflict between the corresponding flights. Note that the conflict graph for a given set of trajectories depends on the parameters of the problem. In the case of only departure delays, whether or not a potential conflict, and thus an edge in the conflict graph, exists between two flights is a function of the maximum allowable departure delay $d_{\max}$. For a certain value of $d_{\max}$, the conflict graph may contain several connected components, which can be considered as smaller, independent instances. Figure 2 shows this dependence of the number of connected components (both including and excluding trivial connected components, i.e. those containing a single vertex) on the maximum delay $d_{\max}$, and Figure 3 shows the distribution of the sizes of the connected components for various values of $d_{\max}$. Interestingly, most of the connected components are very small; for example, with $d_{\max} = 60$ minutes, approximately 75% of the connected components contain no more than 10 flights.

In the remainder of this paper, we consider sets of smaller instances corresponding to the connected components of the conflict graph from the larger single instance for various values of $d_{\max}$. Let $\mathcal{I}_{d_{\max}}$ be the set of such instances for a particular value of $d_{\max}$, excluding trivial instances. We say that an instance is trivial if there are no conflicts when all flights therein depart without delay; in particular, this includes instances containing only a single flight.

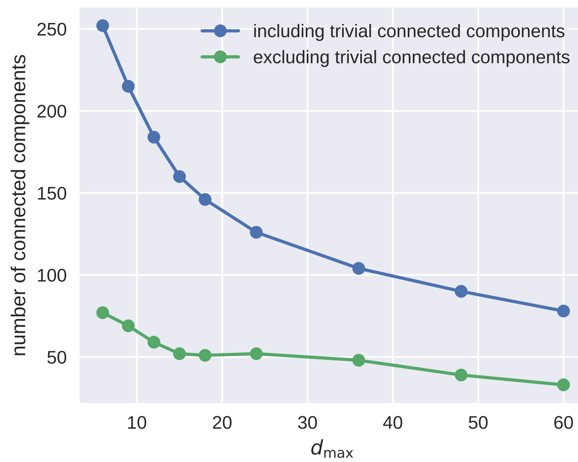

FIG. 2. Number of connected components versus $d_{\max}$.

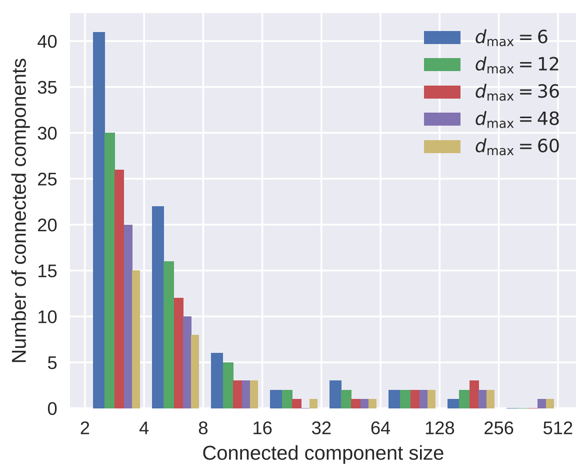

FIG. 3. Histogram of the connected component size for various values of the maximum delay time $d_{\max}$.

As part of our analysis, we also studied the probability distribution of the degree, namely the number of flights for which a given flight share a potential conflict with. Figure (4) shows the distribution of degrees in the conflict graph for $d_{\max} = 60$, which seem to be approximately distributed according to a power law, i.e. the number of vertices with degree d is proportional to d^{α}. This is consistent with a so-called "small-world" model believed to be typical of many real-world graphs [6], which are generated by preferential attachment and resultingly contain a few number of highly-connected hubs, as is the case with air traffic. Figure (5) shows the dependence of this empirical power-law exponent α as a function of $d_{\max}$. As $d_{\max}$ increases, the exponent decreases. The larger the delay, the less the structure of the trajectories matters and the flatter the distribution of degrees in the conflict graph.

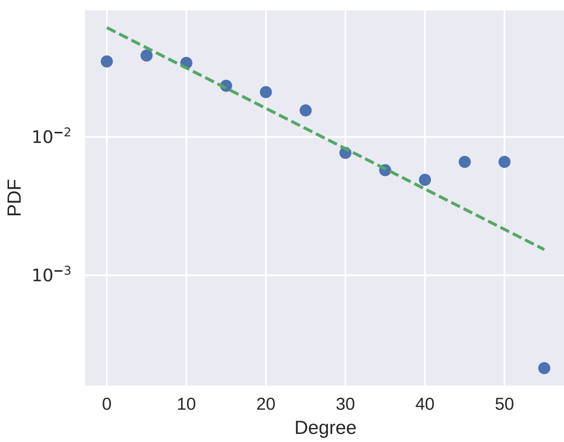

FIG. 4. Histogram of the degrees of vertices in the conflict graph for $d_{\max} = 60$. The distribution of the degrees approximately follows a power law, with the exponent depending on $d_{\max}$.

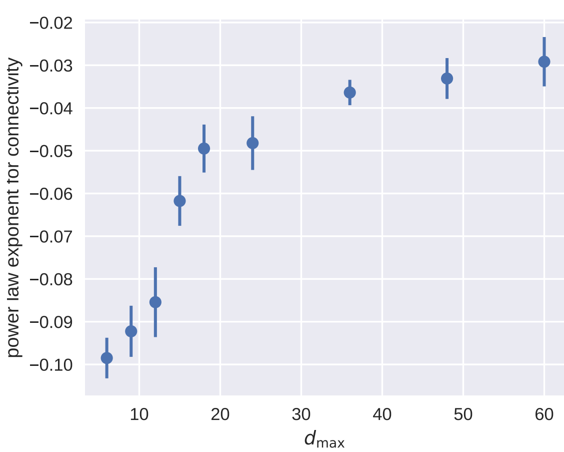

FIG. 5. Empirical power-law exponent versus $d_{\max}$.

In many cases, generally hard problems are easy when restricted to tree-like instances [7, 8]. For example, if the conflict graph here is a tree, then the optimum could be easily found by propagating the delays along the tree; on the other hand, if the conflict graph is a complete graph, finding the optimum is much harder. The tree-width of a graph formalizes this notion of tree-likeness, ranging from 1 for a tree to $n-1$ for fully connected graph. We examine the treewidth of the connected components as a proxy for the hardness of the instances they represent.

Figure 6 shows that the treewidth of a connected component scales approximately linearly with its size. This suggests that realistic instances of the deconflicting are indeed hard, and not restricted to easier (bounded tree-width) instances of the generally hard problem. Moreover, the correlation γ between the tree-width of a connected component and its size increases with $d_{\max}$, as shown in Figure 7. The larger $d_{\max}$, the more potential conflicts there are; restricting $d_{\max}$ also restricts the number of conflicts.

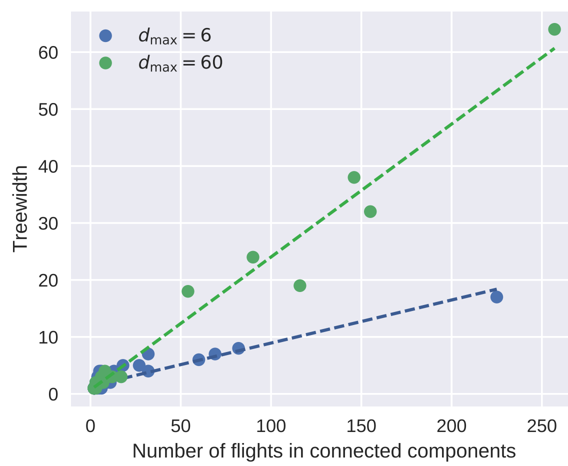

FIG. 6. The treewidths of connected components versus their sizes for various values of $d_{\max}$. The correlation is approximately linear, with a slope γ that depends on $d_{\max}$.

IV. DISCRETIZING THE CONFIGURATION SPACE

To apply quantum annealing to the deconflicting problem, we must encode the configuration space $\mathbf{d}$ in binary-valued variables. To do so, we must first discretize and bound the allowed values. Let Δ_d be the resolution of the allowed delays and $d_{\max} = N_d \Delta_d$ the maximum allowed delay, so that $d_i \in \{\Delta_d l | l \in [0, 1, \ldots, N_d]\}$. The larger the configuration space, the more qubits needed to encode it, and so determining the effect of this discretization on solution quality is integral to the effective use of quantum annealing. To do so, we solve the deconflicting problem with departure delays only for various delay resolutions and upper bounds and compare the various optima to

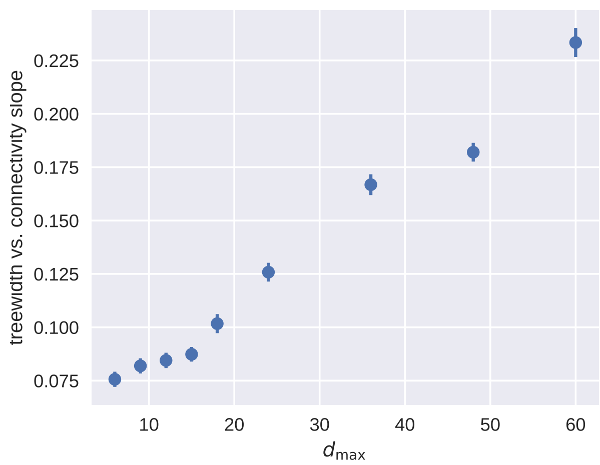

FIG. 7. Slope γ as a function of the maximum delay time.

the continuous problem without restrictions (other than non-negativity) on the delays.

We consider two sets of instances, $\mathcal{I}_{18}$ and $\mathcal{I}_{60}$. For $\mathcal{I}_{18}$, the exact optima are found by modeling the problem as a constraint satisfaction problem [9]; the largest instance in $\mathcal{I}_{18}$ has 50 flights and 104 potential conflicts.

The instances in $\mathcal{I}_{60}$ are much larger and harder; we solved them by mapping to QUBO (as described in the next section) and then using the Isoenergetic Cluster Method (a rejection-free cluster algorithm for spin glasses that greatly improves thermalization) [10], which has been shown to be one of the fastest classical heuristic to optimize QUBO problems [11]. Because ICM is a classical method, the penalty weights can be set arbitrarily large, ensuring that the desired constraints are satisfied. While not guaranteed to return the global optimum in general, for the sizes of instances to which we applied ICM the results are sufficiently well converged to conclude that the solution found is indeed globally optimal with exceedingly high probability.

Figure 8 shows the minimum total delay of a problem instance with 19 flights and 47 potential conflicts for various values of Δ_d and $d_{\max}$. With the exception of the small maximum delay $d_{\max} = 3$, the total delay of the solutions is nearly independent of the maximum delay. The total delay is non-decreasing with respect to the coarseness Δ_d of the discretization for a fixed maximum delay $d_{\max}$, and non-increasing with respect to $d_{\max}$ for a fixed Δ_d. Since the original data is discretized in time in units of 1 minute, $\Delta_d = 1$ yield the same result as a continuous variable with the same upper bound. Above some threshold value $d^0_{\max}$, further increasing the maximum delay does not decrease the minimum total delay. With one exception, we found that for all the investigated problem instances $d^0_{\max} \leq 6$ minutes (see Figure 8). Therefore we conclude, that a moderate maximum delay is sufficient even for larger problem instances. On the other hand,

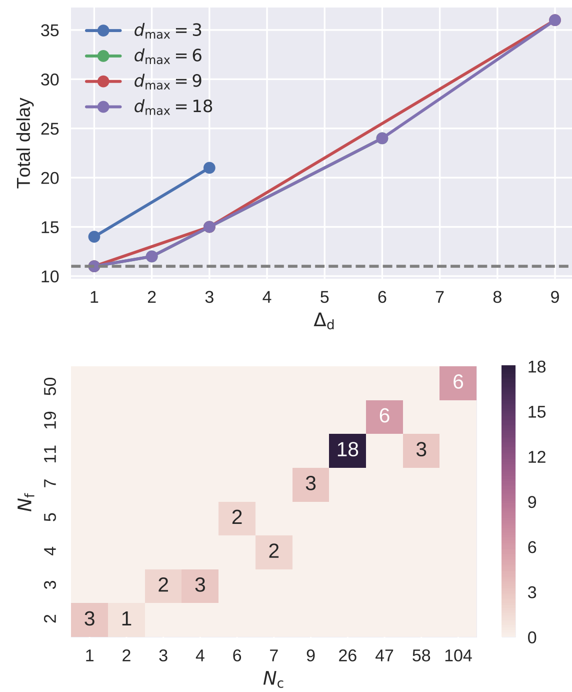

FIG. 8. Top: Minimum total delay of a problem instance from $\mathcal{I}_{18}$ with 19 flights and 47 conflicts for various values of Δ_d and $d_{\max}$. Bottom: Minimum $d_{\max}$ necessary to obtain same optimum as that without bounding the delay for various instances in $\mathcal{I}_{18}$.

the delay discretization should be as fine as possible to obtain a high quality solutions.

Figure (9) shows the dependence of the total delay time optimized by ICM on the delay discretization Δ_d for various problem instances extracted from the connected components of the conflict graph. Results are for maximum delay of 60 minutes. As expected, the total delay decreases by decreasing Δ_d. This is consistent with the idea that smaller Δ_d allows a finer optimization of the delays of the flights.

In Figure (10) we show the optimal delay time found by ICM as a function of the number of the flights in the connected components. Results are for a maximum delay of 60 minutes. Unfortunately, ICM was unable to optimize connected components with more than 12 flights. This can be explained by recalling that ICM works the best for almost-planar problem while the its performance quickly decreases for fully-connected problems. Indeed, as shown in Section III, the underlying graph of connected components look more like a fully-connected graph rather than a tree graph by increasing the number of flights inside the connected component.

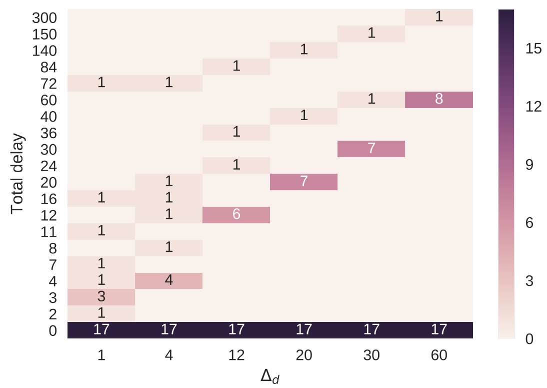

FIG. 9. Total delay in dependence of the discretization parameter Δ_d for 9 different problem instances from $\mathcal{I}_{60}$ with up to in 12 flights and 25 conflicts.

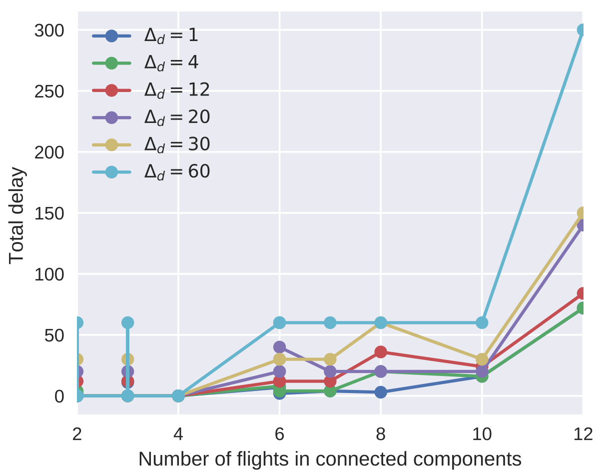

FIG. 10. . Optimal total delay found by using the Isoenergetic Cluster Method (ICM) at fixed time step Δ_d as a function of numbers of flight within each connected component. ICM was unable to find solutions for connected component with more than 12 flights.

V. MAPPING TO QUBO

In this section, we describe how to map to QUBO from the deconflicting problem restricted to only departure delays; a more general mapping is found in the appendix.

A. Binary encoding

Having suitably discretized the configuration space, we must then encode it into binary-valued variables. The value of d_i is encoded in $N_d + 1$ variables $d_{i,0}, \ldots, d_{i,N_d+1} \in \{0, 1\}$ using a one-hot encoding:

$$d_{i,\alpha} = \begin{cases} 1, & d_i = \alpha, \\ 0, & d_i \neq \alpha; \end{cases} \qquad d_i = \Delta_d \sum_{l=0}^{N_d} d_{i,l}. \tag{11}$$

To enforce this encoding, we add the penalty function

$$f_{\text{encoding}} = \lambda_{\text{encoding}} \sum_{i=1}^{n} \left(\sum_{l=0}^{N_d} d_{i,l} - 1 \right)^2, \tag{12}$$

where $\lambda_{\text{encoding}}$ is a penalty weight sufficiently large to ensure that any cost minimizing state satisfies $f_{\text{encoding}} = 0$. In terms of these binary variables, the cost function is

$$f_{\text{delay}} = \Delta_d \sum_{i=1}^{n} \sum_{l=0}^{N_d} l d_{i,l}, \tag{13}$$

Lastly, actualized conflicts are penalized by

$$f_{\text{conflict}} = \lambda_{\text{conflict}} \sum_{k} \sum_{\substack{l,l' | \Delta_d(l-l') \in D_k \\ i,j \in I_k | i<j}} d_{i,l} d_{j,l'}, \tag{14}$$

where again $\lambda_{\text{conflict}}$ is a sufficiently large penalty weight. The overall cost function to be minimized is

$$f = f_{\text{encoding}} + f_{\text{delay}} + f_{\text{conflict}}. \tag{15}$$

B. Softening the constraints

In the QUBO formalism, there are no hard constraints; thus the use of penalty functions in the previous section. For sufficiently large penalty weights, the optimum will satisfy the desired constraints. However, precision is a limited resource in quantum annealing [?]; therefore, we would like to determine the smallest sufficient penalty weights.

For a given instance, we say that a pair of penalty weights $(\lambda_{\text{conflict}}, \lambda_{\text{encoding}})$ is valid if the minimum of the total cost function satisfies both the conflict and encoding constraints when using those weights. Figure 11 shows the phase space of these penalty weights for a single instance with 7 flights and 9 conflicts. The box-like boundary between valid and invalid penalty weights suggests that the validity of the two penalty weights is independent; this box-like boundary is found for all of our instances with up to 7 flights and 9 conflicts.

VI. QUANTUM ANNEALING

In this section we report on our efforts to solve problem instances from the departure delay model from Section V with a D-Wave 2X quantum annealer. We restricted ourselves to instances with $d_{\max} = D_{\max} = 18$ and $\Delta_d \in \{3, 6, 9\}$.

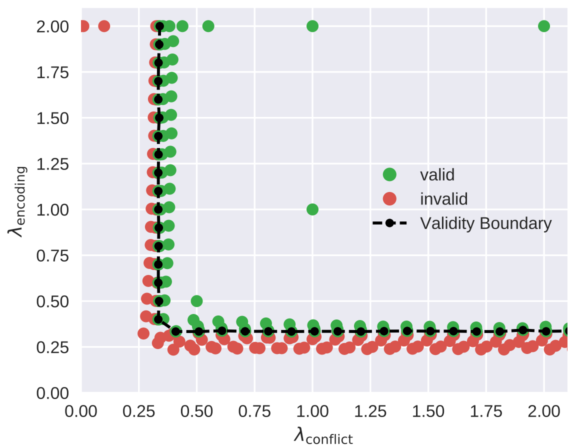

FIG. 11. Validity of exact solution to a QUBO extracted from a problem instance with $N_f = 7$ flights and $N_c = 9$ conflicts in dependence on the choice of the penalty weights, $\lambda_{\text{encoding}}$ and $\lambda_{\text{conflict}}$. Here, $\Delta_t = 6$ and $d_{\max} = 18$.

A. Embedding

In order to make a QUBO amenable for a D-Wave 2X quantum annealer, it has to obey certain hardware constraints. For instance the connections between the binary variables are restricted to the so called Chimera graph [3] However, it is possible to map every QUBO to another QUBO which obeys the Chimera architecture while increasing the number of binary variables used by a so called minor-embedding technique [12]

Δ_d	3	6	9
Number of flights N_f	13	19	50
Number of conflicts N_c	27	47	104
Number of logical qubits	91	76	150
Average number of physical qubits	631	395	543

TABLE I. Parameters of the largest embeddable instances for the D-Wave 2X

We found, that non of the instances were suitable for direct calculation on the D-Wave machine. Therefore we used D-Wave's heuristic embedding algorithm [13] to embed instances with up to $N_f = 50$ and $N_c = 104$ depending on discretization (cf. Table I). We used up to 5 different embeddings for each QUBO instance. In figure 12 on can see the dependence of the number of physical qubits on the number of logical qubits.

B. Success Probability

In order to investigate the performance of the D-Wave 2X machine, we compared its results to the ones of an exact solver. We used an exact Max-SAT solver [?]

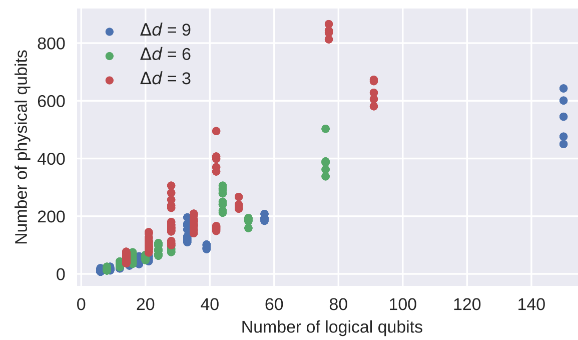

FIG. 12. Number of physical qubits versus the number of logical qubits after embedding of QUBO instances for the departure delay model.

after we mapped the QUBOs to Max-SAT [?]. For each QUBO instance, we ran the annealing process $N_r = 10000$ times. The success probability is then given by the ratio of the number of annealing solutions which are equal to exact solution and the N_r.

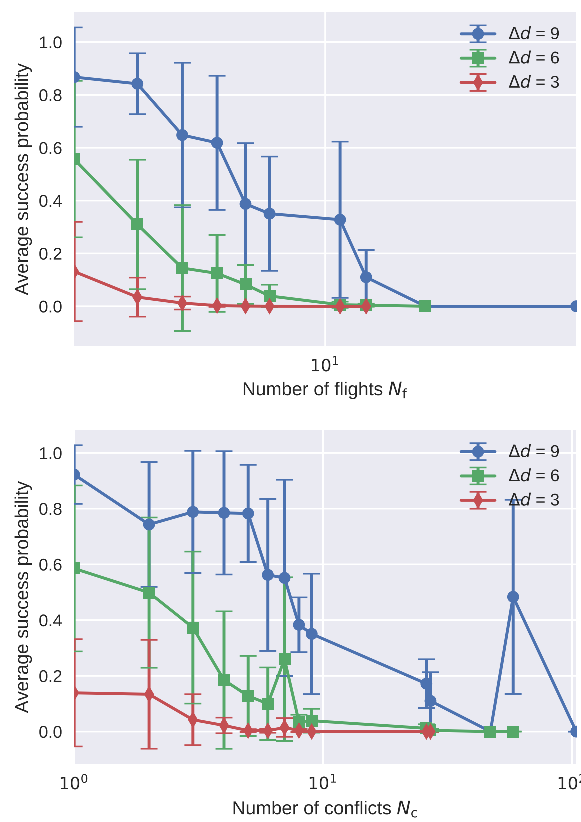

FIG. 13. Average success probability for QUBO instances in dependence of the number of flights N_f and the number of conflicts N_c. The error bars indicate the standard deviation. We used 10000 annealing runs for each instance and penalty weights $\lambda = \lambda_{\text{conflict}} = \lambda_{\text{unique}} \in \{0.5, 1, 2\}$.

In figure 13 the dependence of the average success probability on the number of flights and the number of conflicts is shown. The average was taken over all successfully embedded QUBO instances with the same number of flights and number of conflicts, respectively. On can see, that the success probability decreases for larger problem instances as well as for finer discretizations. This is a result of the limited precision in the specification of a QUBO on the D-Wave 2X machine [**?**].

VII. CONCLUSIONS

In this paper, we propose a novel qubo mapping for a simplified version of one the NASA critical missions: the Air Traffic Management (ATM), i.e. the problem to find minimal modifications of wind-optimal trajectories to avoid conflicts between flights. In our study, we considered the actual wind-optimal trajectories for transatlantic flights (NAT) on July 29, 2012. Given the large number of flights, it is not a viable solution to directly map the optimal-wind trajectories in a qubo model. To avoid this bottleneck, our modified version of the ATM problem is based on the main assumption that flight maneuvers to avoid conflicts only locally modify the optimal-wind trajectories, with the net effect to introduce "delays" to the flights. Therefore, optimal-wind trajectories can be "hard encoded" in our qubo formulation of the ATM model, leaving the flights delays as the only variables to optimize. Nevertheless, as explained in Appendix 2, our method is enough general to potentially include the effect of maneuvers as well.

As part of our study, we also introduce a novel "preprocessing" algorithm to eliminate non-potential conflicts that, given a maximum delay, can never occur. This novel approach is not only important to greatly reduce the number of potential conflicts (as shown in Section XX), but it also gives an important indication of the underlying topology the conflict graph. Indeed, we have discovered that most of the flights have very few conflicts while there are few flights that have conflicts in a non trivial way. The latter set of flights represent the hardest part of the ATM problem to optimize. We want to stress that the proposed pre-processing algorithm is general and can be succesfully applied for the original ATM problem as well, aiding the already existing software to improve both the speed and quality to find optimal modification of the wind-optimal trajectories.

Finally, we have analyzed the performance of both classical and quantum heuristics in solving the our qubo model where only delays at the departure are allowed. Results show that xxx [what can we say here guys?]

In conclusion, we present one of the first attempt to model the Air Traffic Management problem onto a qubo problem.

VIII. ACKNOWLEDGEMENTS

Appendix A: General QUBO mapping

In this section we describe a mapping to QUBO of a more general version of the deconflicting problem than that covered in the main text.

1. Alternative encodings

In the mappings describe both in the main text and the appendix, we use a one-hot encoding to encode a variable. This is best for the specific mappings we described, but in variants an alternative may be better. Say we have a variable x that we want to allow to have variables from finite set $W = \{w_1, w_2, \ldots, w_m\}$. The one-hot encoding has m bits $(x_i)_{i=1}^m$ such that $x = \sum_{i=1}^m w_i x_i$ and $\sum_{i=1}^m x_i = 1$. While we focus on the case in which $W = \{0, 1, \ldots, m-1\}$, our methods are not dependent on that being case, and in particular can address non-uniform sets of values, say if via clever preprocessing it can be determined that such a set would be sufficient. An alternative encoding would remove the requirement that exactly one of the bits is one. The variable x would still be encoded as $x = \sum_{i=1}^m w_i x_i$, but without the one-hot constraint can take on values in $\{\sum_i b_i w_i | b_i \in \{0, 1\}\}$. In particular, this encompasses the unary encoding in which $w_i = 1$ for all i and thus $x \in [0, m]$, as well as the binary encoding $w_i = 2^{i-1}$ for which $x \in [0, 2^m - 1]$. The latter has the advantage of requiring much fewer qubits, but at the cost of similarly increased precision. The former requires the same number of qubits as the one-hot encoding we use, and even has the benefit of minimal precision, but does not allow for quadratic constraints that penalize certain pairs of values of variables, e.g. $d_i - d_j \neq B_k$, without the use of ancillary bits. In models in which the bits x_i only appear in the sum $\sum_i x_i$, it is actually preferable to use the unary encoding to improve the precision requirements. We stick to the one-hot encoding for simplicity, but in practice the unary encoding should be used when possible.

To make the expressions more concise, we define the generalized encoding penalty function

$$f_{\text{encoding}}\left(\{X_i\}_i\right) = \lambda_{\text{encoding}} \sum_i \left(\sum_{x \in X_i} x - 1 \right)^2 \quad \text{(A1)}$$

that enforces the constraint that exactly one bit x is one for each set of bits X_i.

2. Global trajectory modifications

Consider the case in which each trajectory can be modified by a departure delay and some parameterized spatial transformation, i.e. for each flight i there is a variable d_i and some parameter $\boldsymbol{\theta}_i$. For example, Rodionova et al. [2]

consider a single angle θ_i that determines a sinusoidal transformation of the trajectory. For the QUBO mapping, we require that these variables be allowed to take on values from some finite set, so that are QUBO variables are $\{d_{i,\alpha}\}$ and $\{\boldsymbol{\theta}_{i,\phi}\}$, where $d_{i,\alpha} = 1$ ($d_{i,\alpha}$) indicates that $d_i = \alpha$ ($d_i \neq \alpha$) and similarly for $\boldsymbol{\theta}_{i,\phi}$. For every pair of flights $i < j$, we can efficiently (in time and space polynomial in the size of the input) compute whether the corresponding trajectories conflict when modified according to d_i, d_j, $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$. Let $B_{i,j}$ be the set of values of $(d_i, \boldsymbol{\theta}_i, d_j, \boldsymbol{\theta}_j)$ such that the the modified trajectories conflict. Lastly, let $d_{(i,\alpha),(j,\beta)}$ indicate that $d_i = \alpha$ and $d_j = \beta$, and similarly for $\boldsymbol{\theta}_{(i,\phi),(j,\psi)}$. The overall cost function is

$$f_{\text{global}}\left(\left(d_{i,\alpha}\right)_{i,\alpha}\left(d_{(i,\alpha),(j,\beta)}\right)_{i,j,\alpha,\beta}\left(\boldsymbol{\theta}_{(i,\phi),(j,\psi)}\right)_{i,j,\phi,\psi}\right) = f_{\text{encoding}} + f_{\text{consistency}} + f_{\text{delay}} + f_{\text{conflict}}, \quad \text{(A2)}$$

where

$$f_{\text{encoding}}\left(\left\{\left\{d_{i,\alpha}\right\}_\alpha \cup \left\{\boldsymbol{\theta}_{i,\phi}\right\}_\phi\right\}_i\right) \quad \text{(A3)}$$

ensures that the values of d_i and $\boldsymbol{\theta}_i$ are uniquely encoded;

$$f_{\text{consistency}} = \lambda_{\text{consistency}}\left[\sum_{i<j,\alpha,\beta} s\left(d_{i,\alpha}, d_{j,\beta}, d_{(i,\alpha),(j,\beta)}\right) + \sum_{i<j,\phi,\psi} s\left(\boldsymbol{\theta}_{i,\phi}, \boldsymbol{\theta}_{j,\psi}, \boldsymbol{\theta}_{(i,\phi),(j,\psi)}\right)\right] \quad \text{(A4)}$$

ensures consistency between the values of $d_{i,\alpha}$, $d_{j,\beta}$, and $d_{(i,\alpha),(j,\beta)}$;

$$s(x, y, z) = 3z + xy - 2xz - 2yz \quad \text{(A5)}$$

is a non-negative penalty function that is zero if and only if $z = xy$;

$$f_{\text{delay}} = \sum_{i,\alpha} \alpha d_{i,\alpha} \quad \text{(A6)}$$

is the cost function to be minimized; and

$$f_{\text{conflict}} = \lambda_{\text{conflict}} \sum_{i<j} \sum_{(\alpha,\phi,\beta,\psi)\in B_{i,j}} d_{(i,\alpha),(j,\beta)} \boldsymbol{\theta}_{(i,\phi),(j,\psi)} \quad \text{(A7)}$$

penalize conflicts.

3. Local trajectory modifications

Alternatively, we can consider modifications to the trajectory only near conflicts. We describe a few special models and their mapping to QUBO, though many more such ways of doing so, and we leave a full accounting for future work.

a. *Exclusive avoidance*

Suppose for every conflict k and associated pair of flights $i < j$, there is a way for either flight to go around the trajectory of the other, introducing some delay $d_{i,k}$ to flight i or $d_{j,k}$ to flight j depending on which trajectory is changed. Let $a_k = a_{i,k} = 1$ ($a_{i,k} = 0$) indicate that flight i's trajectory is changed, and for convenience let $a_{j,k} = a_{i,k} - 1$, though only one (qu)bit will be used per conflict. Adding in the departure delay, we have the total cost function

$$f_{\text{exclusive}}\left(\left(d_{i,\alpha}\right)_{i,\alpha}, \left(a_k\right)_k\right) = f_{\text{delay}} + f_{\text{encoding}}, \quad \text{(A8)}$$

where

$$f_{\text{delay}} = \sum_i \left[\sum_\alpha \alpha d_{i,\alpha} + \sum_{k\in K_i} d_{i,k} a_{i,k}\right] \quad \text{(A9)}$$

and f_{encoding} is as in (12). This assumes that the trajectory modifications don't introduce potential conflicts with other flights; this assumption can be partially relaxed by adding penalty terms of the form $a_{i,k}a_{j,k'}$ or $d_{i,\alpha}a_{j,k}$ as appropriate.

b. *Flexible avoidance*

Exclusive *requires* that one or the other flight is delayed at each conflict. We can relax this by accounting for the fact that if the flights arriving at a potential conflict are already relatively delayed, the conflict could be passively avoided. Let $D_{k,\gamma} = 1$ ($D_{k,\gamma} = 0$) indicate that $D_k = \gamma$ ($D_k \neq \gamma$), where D_k is the difference in the accumulated delays at conflict k as defined in (7).

The total cost function is

$$f_{\text{flexible}}\left(\left(d_{i,\alpha}\right)_{i,\alpha}, \left(a_{i,k}\right)_{k,i\in I_k}, \left(D_{k,\gamma}\right)_{k,\gamma}\right) = f_{\text{encoding}} + f_{\text{delay}} + f_{\text{consistency}} + f_{\text{conflict}}, \quad \text{(A10)}$$

where the first term is

$$f_{\text{encoding}}\left(\left\{\left\{d_{i,\alpha}\right\}_\alpha\right\}_i \cup \left\{\left\{D_{k,\gamma}\right\}_\gamma\right\}_k\right); \quad \text{(A11)}$$

the consistency term is

$$f_{\text{consistency}} = \lambda_{\text{consistency}} \sum_k \left(D_{i,k} - D_{j,k} - \sum_\gamma \gamma D_{k,\gamma}\right)^2 \quad \text{(A12)}$$

using the notational variables

$$D_{i,k} = \sum_\alpha \alpha d_{i,\alpha} + \sum_{k'\in K_{i,k}} d_{i,k'} a_{i,k'}; \quad \text{(A13)}$$

f_{delay} is as in (A9) but where $a_{i,k}$ and $a_{j,k}$ are separate bits; and

$$f_{\text{conflict}} = \lambda_{\text{conflict}} \sum_k \sum_{\gamma\in B_k} \left[D_{k,\gamma}\left(1 - a_{i,k} - a_{j,k}\right) + 2a_{i,k}a_{j,k}\right] \quad \text{(A14)}$$

If we want to allow both flights to be delayed at conflict $a_{i,k} = a_{j,k} = 1$, we must introduce an ancillary bit a_k that indicates whether at least one flight is delayed at conflict k, adding

$$\lambda_{\text{consistency}} \sum_k \left[\left(a_{i,k} + a_{j,k}\right)\left(1 - 2a_k\right) + a_{i,k}a_{j,k}\right] \quad \text{(A15)}$$

to $f_{\text{consistency}}$, and replacing f_{conflict} with

$$\sum_k \sum_{\gamma \in B_k} D_{k,\gamma}(1 - a_k). \quad \text{(A16)}$$

c. Interstitial delays

In the interstitial-delay model, the local modifications are not made *at* conflicts but *between* them, and conflicts are only avoided via accumulated delays. That is, the delay $d_{i,k}$ introduced to flight i before reaching conflict k but after leaving the previous conflict $\max_{k' \in K_{i,k}} k'$. Unlike in the flexible avoidance model, $d_{i,k}$ is now a variable rather than a parameter, and we encode it using bits $d_{i,k,\delta}$.

$$f_{\text{interstitial}}\left(\left(d_{i,\alpha}\right)_{i,\alpha}, \left(D_{i,k,\gamma}\right)_{i,k \in K_i,\gamma}\right) = f_{\text{encoding}} + f_{\text{consistency}} + f_{\text{conflict}} + f_{\text{delay}}, \quad \text{(A17)}$$

where

$$f_{\text{encoding}}\left(\left\{\left\{d_{i,\alpha}\right\}_\alpha\right\}_i \cup \bigcup_i \left\{\left\{D_{i,k,\gamma}\right\}_\gamma\right\}_{k \in K_i}\right), \quad \text{(A18)}$$

$$f_{\text{consistency}} = \sum_i \sum_{k,k' \in K_i | k' = \max K_{i,k}} \sum_{(\gamma,\gamma') \in B_{i,k}} D_{i,k,\gamma} D_{i,k',\gamma'}, \quad \text{(A19)}$$

$$f_{\text{conflict}} = \lambda_{\text{conflict}} \sum_{k=1}^{N_c} \sum_{(\gamma,\gamma') \in B_k} D_{i,k,\gamma} D_{j,k,\gamma'}, \quad \text{(A20)}$$

and

$$f_{\text{delay}} \sum_i \sum_\gamma D_{i,\max K_i,\gamma}. \quad \text{(A21)}$$

[1] H. K. Ng, B. Sridhar, and S. Grabbe, *Optimizing Aircraft Trajectories with Multiple Cruise Altitudes in the Presence of Winds*, Journal of Aerospace Information Systems **11**, 35 (2014), URL http://dx.doi.org/10.2514/1.I010084.

[2] O. Rodionova, D. Delahaye, B. Sridhar, and H. Ng., *Deconflicting wind-optimal aircraft trajectories in north atlantic oceanic airspace*, Proceedings of Advanced Aircraft Efficiency in a Global Air Transport System (AEGATS'16) Conference (2016).

[3] 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 Information Processing **14**, 1 (2015), ISSN 1573-1332, URL http://dx.doi.org/10.1007/s11128-014-0892-x.

[4] E. Rieffel, D. Venturelli, M. Do, I. Hen, and J. Frank, *Parametrized families of hard planning problems from phase transitions* (2014), URL https://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8614/8697.

[5] D. Venturelli, D. J. J. Marchand, and G. Rojo, *Quantum Annealing Implementation of Job-Shop Scheduling*, arXiv:1506.08479 [quant-ph] (2015), arXiv: 1506.08479, URL http://arxiv.org/abs/1506.08479.

[6] R. Albert, H. Jeong, and A.-L. Barabási, *Internet: Diameter of the world-wide web*, nature **401**, 130 (1999).

[7] U. Bertele and F. Brioschi, *Nonserial dynamic programming* (Academic Press, 1972).

[8] R. Halin, *S-functions for graphs*, Journal of geometry **8**, 171 (1976).

[9] E. Hebrard, E. O'Mahony, and B. O'Sullivan, *Constraint Programming and Combinatorial Optimisation in Numberjack* (Springer Berlin Heidelberg, Berlin, Heidelberg, 2010), pp. 181–185, ISBN 978-3-642-13520-0, URL http://dx.doi.org/10.1007/978-3-642-13520-0_22.

[10] Z. Zhu, A. J. Ochoa, and H. G. Katzgraber, *Efficient cluster algorithm for spin glasses in any space dimension*, Physical review letters **115**, 077201 (2015).

[11] S. Mandrà, Z. Zhu, W. Wang, A. Perdomo-Ortiz, and H. G. Katzgraber, *Strengths and weaknesses of weak-strong cluster problems: A detailed overview of state-of-the-art classical heuristics versus quantum approaches*, Physical Review A **94**, 022337 (2016).

[12] V. Choi, *[0804.4884] Minor-Embedding in Adiabatic Quantum Computation: I. The Parameter Setting Problem*, URL http://arxiv.org/abs/0804.4884.

[13] J. Cai, W. G. Macready, and A. Roy, *A practical heuristic for finding graph minors*, arXiv:1406.2741 [quant-ph] (2014), arXiv: 1406.2741, URL http://arxiv.org/abs/1406.2741.