




A QUBO Model to the Tail Assignment Problem

Luis N. Martins¹^a, Ana Paula Rocha²^b and Antonio J. M. Castro²^c

¹Department of Informatics Engineering (DEI), Faculty of Engineering (FEUP), University of Porto, Porto, Portugal

²Artificial Intelligence and Computer Science Lab (LIACC), University of Porto, Porto, Portugal


Keywords: Tail Assignment Problem, Quantum Annealing, Scheduling Problem.


Abstract: Tail Assignment is the problem of allocating individual aircraft to a set of flights subject to multiple constraints while optimising an objective function, such as operational costs. Given the enormous amount of possibilities and constraints involved, this problem has been a case study over the last decade. Many solutions have emerged using classical computing, but with limitations. Quantum Annealing (QA) is a heuristic technique to solve combinatorial optimisation problems by finding global minimum energy levels over an energy landscape using quantum mechanics. In this study, Tail Assignment Problem was framed as a Quadratic Unconstrained Binary Optimisation (QUBO) model and was solved using a classical and two hybrid solvers. The considered hybrid solvers made use of the D-Wave 2000Q quantum annealer. Tests were run based on extractions from real-world data, analysing, empirically, the performance of the implementation in terms of quality (i.e., the lowest operational costs) of the obtained solutions. We concluded that, for the considered datasets, there was a higher probability of obtaining better solutions for this problem using one of the hybrid solvers when compared with a classical heuristic algorithm such as Simulated Annealing (SA).


1 INTRODUCTION

Due to a large number of routes, aircraft and crew, scheduling and operational management are the most complex and challenging tasks that airlines need to face every day. Airline scheduling is a complex process composed of a sequence of tasks, starting from defining which airports and routes the airline company will operate, until the moment an individual aircraft and crew members are assigned to each specific flight. The Tail Assignment Problem is part of this process and aims to efficiently allocate individual aircraft (also called tails) to flights minimising a certain objective function, such as operational costs, while ensuring some constraints (Grönkvist, 2005). Quantum Annealing (QA) is a heuristic technique used to find a global minimum of an objective function defined over an energy landscape using quantum mechanics (Kadowaki and Nishimori, 1998; Ray et al., 1989). It operates by taking advantage of the natural evolution of a quantum system changing an Hamiltonian representing the initial quantum state to a final state representing the minimum final value of the op-

timisation problem it aims to solve (Albash and Lidar, 2018). Such technique has been used to solve complex optimisation problems with a very large set of possibilities aiming to find global minimum solutions (Neukart et al., 2017; Ikeda et al., 2019). Given the definition and complexity of finding a proper solution to the Tail Assignment Problem, as well as the good results when applying QA to optimisation problems, the main goal of this study is to verify the possibility of solving the Tail Assignment Problem, considering the operational restrictions and the operational costs, using QA in order to evaluate the usefulness of this kind of technique in such a complex domain. The rest of this paper is organized as follows, Section 2 presents some literature review concerning the Tail Assignment problem and Quantum Annealing approaches to scheduling problems. Section 3 presents the definition of the problem, including the constraints to be considered as well as the optimisation criteria. Section 4 details the modelling of the problem to be solved using a quantum annealing approach. Experiments and results are discussed in Section 5. Finally, conclusions and future work are presented in Section 6.

^a <https://orcid.org/0000-0001-5032-049X>

^b <https://orcid.org/0000-0002-8129-9758>

^c <https://orcid.org/0000-0001-8121-1974>

2 RELATED WORK

The Tail Assignment Problem has been studied over the last decade with multiple approaches being presented (Grönkvist, 2005; Ruther et al., 2017; Froyland et al., 2013; Montoito, 2016; Yadav, 2017).

In (Grönkvist, 2005) the authors studied an hybrid approach, using constraint programming and a branch-and-price algorithm. By defining the Tail Assignment Problem as a set partitioning problem based on pre-calculated routes, it starts by modelling the problem as a Constraint Satisfaction Problem (CSP). Such modelling is important for generating an initial feasible solution that is later optimised using a branch-and-price algorithm solving the problem for a fixed period of time and taking into account some specific activities and irregular schedules. The combination of both algorithms allows improving initial solutions that respect operational constraints such as maintenance, turnaround times and pre-assigned activities.

Following also a branch-and-price algorithm, in (Ruther et al., 2017) the authors present a solution for the Tail Assignment Problem together with the crew pairing. This approach is thought to be executed only few days before the operation, creating scheduled routes for a specific aircraft instead of generic ones.

In (Montoito, 2016) the author presents a solution for the Tail Assignment Problem using SA with an adaptive neighbourhood local search approach. Starting by obtaining an initial feasible solution using a First-In-First-Out (FIFO) approach, it made use of a SA algorithm to minimise the operational costs.

To solve problems using a QA approach, as for using D-wave 2000Q quantum annealer, multiple studies formulate it as a specific Binary Quadratic Model (BQM) called QUBO model. By doing this it is possible to set the problem aimed to be solved as the final Hamiltonian of the system Hamiltonian. In fact, a QUBO model may be expressed as in equation 1:

$$E_{qubo}(a_i, b_{i,j}; q_i) = \sum_{i=1} a_i q_i + \sum_{i < j} b_{i,j} q_i q_j \quad (1)$$

where q_i and q_j correspond to qubits, a_i represents the linear coefficients of the qubits and $b_{i,j}$ are quadratic coefficients representing the strength between each pair of connected (coupled) qubits.

Although presented here in a scalar notation, the biggest difference between both notations is that QUBO model can also be redefined using an upper-diagonal matrix whereas Ising model cannot.

In fact, a QUBO model can be expressed as a minimisation problem presented in equation 2 where x is a

vector of binary variables and Q is an upper-triangular matrix of constants that represent the biases and coupling strengths of the considered binary variable.

$$\text{minimise } y = x^T Q x \quad (2)$$

Due to the few research developments in this area, no studies were found applying QA to the Tail Assignment Problem. Therefore, a wider analysis was performed searching for another kind of problems that could have an implementation using this approach. In (Venturelli et al., 2016) the authors propose a solution for the Job Shop scheduling problem using a QA approach. They concluded that the usage of hybrid approaches and meta-heuristics were fruitful and should exist further investigation on such area. Furthermore, they also concluded that pruning unnecessary variables is an important step as it may highly reduce the size of the problem. Moreover, in (Stollenwerk et al., 2019) the authors implemented a flight gate assignment minimising the total time for transit passengers. Converting this NP-hard problem to a QUBO model, they tested how a quantum annealer, in this case a D-Wave 2000Q, could solve this problem. Due to hardware limitations, only a small portion of the real data was considered, concluding that extracting problem instances from data can lead to a QUBO model with distributed coefficients, reducing the success probability. Another study made use of a D-Wave 2000Q quantum annealer, but this time for solving the Nurse Scheduling Problem with hard constraints (Ikeda et al., 2019). It concluded that satisfiable solutions could be achieved using QA but only for small samples due to hardware constraints. Some other QUBO models were also defined for well-known problems such as the graph partitioning problem (Ushijima-Mwesigwa et al., 2017), the maximum clique problem (Chapuis et al., 2019), the traveling salesman problem (Papalitsas et al., 2019) or the minimum multi-cut problem (Cruz-Santos et al., 2019). Furthermore, in (Lucas, 2014) the author proposes a formulation for multiple well-known NP-hard and NP-complete problems, that can be run on a quantum annealer, and in (Tran et al., 2016) the author proposed a hybrid approach for multiple complex problems.

When taking into account other quantum approaches rather than QA, it is possible to find one study presenting a partial implementation of the Tail Assignment Problem using a Quantum Approximate Optimisation Algorithm (QAOA). In fact, in (Vikstål et al., 2019) the authors had as main goal to find a feasible solution considering some constraints. Due to the limited capabilities of the used quantum computer, a limited number of pre-calculated routes (us-

ing branch-and-price) was considered, ensuring that one of these routes was a solution with all the flights assigned. It also considered that all the aircraft would be used. Furthermore, it was able to find a solution satisfying all the constraints disregarding costs. Regarding the application of QA to the Tail Assignment Problem, no studies were found so the present study aims to fulfil such a gap.

3 PROBLEM STATEMENT

The Tail Assignment Problem consists of assigning individual aircraft to flights, while ensuring multiple constraints and minimising a certain objective function. Due to the complexity of this problem, in this paper it was considered a simplification of it, for illustration purposes. Therefore, we considered the following constraints and optimisation criteria.

3.1 Assignment Constraints

Assignment constraints are responsible for ensuring that no activities are left unassigned and that each activity is attributed only to one tail. Following this constraint, it is possible to guarantee that each and every flight is allocated exclusively to one aircraft.

3.2 Connection Constraints

Connection constraints are the most basic ones and can be as simple as ensuring that two activities may connect one another, i.e., can be operated in sequence. Such constraints can be thought as a graph where each node represents an activity and two connected nodes are representative of two activities that can be assigned to a same aircraft in sequence, taking into account the turnaround time, i.e., the minimum time needed for an aircraft to park in a gate between two consecutive activities.

3.3 Maintenance Constraints

Maintenance constraints ensure that each aircraft performs the required maintenance tasks. In fact, maintenance tasks can be divided in two major groups namely *Minor Maintenance Tasks* and *Pre-assigned Maintenance Tasks*. The first group happen every flight or up to once a week and take up to few hours. The second group of maintenance tasks are considered as long-term tasks and take from one day up to months to be completed, and happening more sparsely in time, are normally planned ahead in time. In this study we just take into account *Pre-assigned*

Maintenance Tasks, that specify the aircraft that has to perform it as well as the exact time and location it must happen.

3.4 Flight Restriction Constraints

Flight Restriction Constraints aim to ensure that each flight is operated by a valid tail. In fact, each flight may require a certain fleet, minimum number of seats, among others. In this study we considered that each flight has a minimum fleet required as well as a minimum number of seats corresponding to the expected demand of the flight. Therefore, tails belonging to a smaller fleet or not having the required number of seats were considered as being unfit to perform the flight.

3.5 Optimisation Criteria

The optimisation criteria is considered as the minimisation of an objective function (OF) which comprises the costs that represent the bigger impact on the schedule. Those are the costs related to the execution of the flights by the different tails. In fact, due to the variability of the tickets price and lack of information of the real tickets price, we decided to take into consideration only the biggest costs that really affect the sustainability and profitability of the airline companies. For a matter of simplicity, we divided them into three parts: *Ground costs*, *Flying costs*, and *Take-off/Landing costs*.

Therefore the objective function is defined as in equation 3, where F represents the set of all flights that must be performed.

$$OF = \sum_f^F (Ground\ costs_f + Flying\ costs_f + Takeoff/Landing\ costs_f) \quad (3)$$

4 TAIL ASSIGNMENT PROBLEM AS A QUBO MODEL

Modelling the Tail Assignment Problem as a QUBO model is set in three sequential steps, described below.

4.1 Formulate the Objective and Constraints

Starting by *Formulate the Objective and Constraints*, we specify and systematize the optimisation criteria and groups of constraints.

Objective. To minimise the considered operational costs of the schedule, i.e., to get a solution such that each flight is performed by the tail with the lowest execution cost.

The group of **constraints** to be considered are specified as follows:

- **Assignment Constraints:** each flight must only be performed by one tail;
- **Connection and Maintenance Constraints:** a tail must have a valid schedule regarding arrival and departure locations and times of activities. This group of constraints can be divided into three subgroups as follows:
 - **Impossible Pairing Activities:** activities that have no valid path between them and therefore cannot be performed by the same tail.
 - **Impossible Flights Due to Maintenance:** as maintenance is obligatory, tails cannot perform flights that have no valid path for all their maintenance tasks;
 - **Activity Path Consistency:** all consecutive activities performed by a tail must follow a valid path;
- **Flights Requirement Constraints:** since each flight has a pre-defined minimum fleet and a minimum number of seats required, a tail that meets such requirements is needed to perform the flight.

4.2 Redefine the Problem into a Binary Concept

To *Redefine the Problem to a Binary Concept* it is necessary to convert it from an optimisation problem into a decision problem. We state it as follows: "Should flight X be performed by tail Y ?". Each variable of the problem is then represented as the possible assignment of a certain flight to a specific tail.

Considering a set of F flights labeled as $f = 1, \dots, F$ and a set of T tails labeled as $t = 1, \dots, T$, we define the binary variables $q_{f,t}$ as the variables of the problem's domain.

4.3 Transform the Problem into a QUBO Model

The QUBO model is set as the sum of six different terms that represent both the constraints and the objective function as presented in equation 4.

$$H = \gamma H_A + \eta H_{B_1} + \lambda H_{B_2} + \tau H_{B_3} + \phi H_{B_4} + \psi H_C \quad (4)$$

From this equation $\gamma, \eta, \lambda, \tau, \phi$ and ψ represent positive real-valued numbers used to tune the relative importance of each term in the global QUBO model. H_A is related to the assignment constraints. On its turn, terms $H_{B_1}, H_{B_2}, H_{B_3}$ and H_{B_4} assure connection and maintenance constraints. Finally, H_C is the term responsible for setting a lower energy value to solutions with minimum value of the objective function. To assign values to the different tuning parameters, it is important to take into account that some of them may incorrectly affect the others. For example, the value of ψ must be chosen sufficiently small not to violate any of the constraints of the problem. We considered the following notation:

T : set of all tails

F : set of all flights

M : set of all maintenance tasks

$T_f \subset T$: subset of T that can perform flight f

$F_t \subset F$: subset of F that can be performed by a given tail t

$I_f \subset F_t$: subset of F_t that cannot be assigned together with a given flight f

$IC_f \subset F_t$: subset of F_t that is indirectly connected to a given f on the corresponding tail's connection network graph

$M_t \subset M$: subset of M that must be performed by a given tail t

4.3.1 Assignment Constraints

Since it is desirable that all flights are assigned to exactly one tail, a quadratic penalty is introduced for schedules not meeting such condition. The implementation of this constraint follows equation 5.

$$H_A = \sum_f \left(\sum_t q_{f,t} - 1 \right)^2 \quad (5)$$

Such constraint is set by defining a negative bias to each one of the individual variables $q_{f,t}$ and a positive coupling strength, for each pair of variables $\{q_{f,t}; q_{f,t'}\}$, where the latter corresponds to the assignment of a same flight to different tails.

4.3.2 Connection and Maintenance Constraints

As previously described, connection and maintenance constraints aim to guarantee that the obtained solution is valid regarding sequences of flights while ensuring all pre-defined maintenance tasks can be performed by the correspondent tail.

Impossible Pairing Activities. This subgroup of constraints is set to penalise the cases where two flights (f, f') that have no valid path between them are assigned simultaneously to the same tail (t). As defined in equation 6, it penalises variables that represent non-pairable flights.

$$H_{B_1} = \sum_t \sum_{f \in F_t} \sum_{f' \in I_f} q_{f,t} q_{f',t} \quad (6)$$

Activity Path Consistency. To ensure that the chosen schedule is valid regarding path consistency, an extra penalty is added on paths that are not valid. It can be divided in the following subgroups: *Path consistency between non-consecutive flights*, *Path consistency between maintenance tasks* and *Path consistency between flight and maintenance task*

Path Consistency between Non-consecutive Flights. To ensure path consistency between two non-consecutive flights, equation 7 penalises all pairs of indirectly connected flights (f, f') assigned to a tail (t) that do not have a valid path assigned to that same tail. It is done by penalizing cases where $p(f, f', t)$, which corresponds to the penalty function of the boolean expression $p_b(f, f', t)$ in equation 8, has a value different than 1.

$$H_{B_2} = \sum_t \sum_f \sum_{f' \in IC_f} (p(f, f', t) - 1)^2 \quad (7)$$

$$p_b(f, f', t) = ((q_{f,t} \wedge q_{f',t}) \oplus 1) \vee w \quad (8)$$

In equation 8, $w = q_{f_1,t} \vee q_{f_2,t} \vee \dots \vee q_{f_n,t}$ and $[f_1, \dots, f_n] \in PF$ where PF represents the set of flights that can be reached from flight f and are directly connected to flight f' , when both flights are assigned to tail t .

Translating the desired boolean expression defined in 8 to a penalty function, we use the conversions defined in Table 1, where x_3 represents the output auxiliary variables that take the value of the relationship between the two inputs (x_1 and x_2). As previously presented, the first two penalty functions can be found in the literature, whereas the third penalty function was constructed for this study based on the penalty functions of the trivial boolean relationships.

Table 1: Boolean penalty functions for the QA approach.

Classical Constraint	Equivalent Penalty
$x_3 \Leftrightarrow x_1 \wedge x_2$	$P(x_1 x_2 - 2(x_1 + x_2)x_3 + 3x_3)$
$x_3 \Leftrightarrow x_1 \vee x_2$	$P(x_1 x_2 + (x_1 + x_2)(1 - 2x_3) + x_3)$
$x_3 \Leftrightarrow (x_1 \oplus 1) \vee x_2$	$P(-x_1 x_2 - x_3 + 2x_1 x_3 - 2x_2 x_3 - x_1 + 2x_2 + 1)$

Path Consistency between Maintenance Tasks. To guarantee a valid path between obligatory maintenance tasks, a new penalty is added for the cases where none of the flights that guarantee such path is assigned to the considered tail, as in equation 9.

$$H_{B_3} = \sum_t \sum_{m, m'}^{M_t} (g(m, m', t) - 1)^2 \quad (9)$$

In this equation $g(m, m', t)$ corresponds to the penalty function of the boolean expression $g_b(m, m', t)$ presented in equation 10

$$g_b(m, m', t) = (q_{f_1,t} \vee q_{f_2,t} \vee \dots \vee q_{f_n,t}) \wedge (q_{f'_1,t} \vee q_{f'_2,t} \vee \dots \vee q_{f'_n,t}) \quad (10)$$

where $[f_1, \dots, f_n] \in MPF$ and $[f'_1, \dots, f'_n] \in MNF$. On one hand, MPF represents the set of flights that are part of a valid path between the two maintenance tasks (m and m') and are directly connected to m on the tail's connection network graph. On the other hand, MNF represents the set of flights that are part of a valid path between two maintenance tasks (m and m') and are directly connected to m' on the corresponding tail's connection network graph. Setting boolean expression defined in equation 10 can be done using the penalty functions from Table 1.

Path Consistency between Flight and Maintenance Task. It is necessary to guarantee that any flight that is not directly connected to an existent maintenance task, is only assigned to the tail that must execute such maintenance task if the valid path between both activities is part of the same assignment. Equation 11 sets a penalty for solutions that do not follow such constraint,

$$H_{B_4} = \sum_t \sum_f \sum_m^{M_t} (r(f, m, t) - 1)^2 \quad (11)$$

where $r(f, m, t)$ corresponds to the penalty function of the boolean expression $r_b(f, m, t)$ presented in equation 12,

$$r_b(f, m, t) = (q_{f,t} \oplus 1) \vee (q_{f'_1,t} \vee q_{f'_2,t} \vee \dots \vee q_{f'_n,t}) \quad (12)$$

where $[f'_1, \dots, f'_n] \in MCF$, with MCF representing the set of flights that are directly connected to maintenance task m and have a valid path to flight f on the corresponding tail's connection network graph.

Analysing all constraints as a whole, some redundancies may appear. Therefore, to minimise the number of variables of the problem, while implementing

	2A	3A	4A	5A	6A	2B	3B	4B	5B	6B	7B	8B	2C	3C	4C	5C	6C	7C	8C
2A	-1																		
3A		1	1			2							2						
4A			1	1			2							2					
5A				1				2							2				
6A					1				2							2			
2B						-1													
3B							1	1					2						
4B								1	1					2					
5B									1	1					2				
6B										1	1					2			
7B											1	1					2		
8B												1	1					2	
2C													-1						
3C														1	1				1
4C															1	1			
5C																1	1		
6C																	1	1	
7C																		1	1
8C																			-1

Figure 1: Partial QUBO model matrix for the illustrative example: 2,A corresponds to the variable represented by the assignment of flight 2 to tail A. The colored entries correspond to the coefficients from Equations (5) and (6) and empty entries correspond to coefficient 0.

each constraint, it is necessary to verify whether or not that constraint is already assured.

Figure 1 shows a QUBO model of the illustrative problem for some of the constraints, in the form of an upper-triangular matrix. Furthermore, every coefficient is multiplied by the tuning parameter associated to the constraint it represents.

Objective Function. Each variable $q_{f,t}$ gets its bias increased depending on how smaller is the operational cost of the assignment it represents when compared with the maximum cost possible for performing flight f . Equation 13 represents the associated penalty function,

$$H_C = \sum_f \sum_t \frac{exccost_{f,t}}{maxcost_f} q_{f,t} \quad (13)$$

where $exccost_{f,t}$ represents the cost of performing flight f on tail t , and $maxcost_f$ represents the maximum cost of performing flight f in any of the possible tails.

Therefore for each flight, H_C will take the value of 1 for the most expensive assignment.

5 TESTS AND RESULTS

To verify and analyse the proposed implementation it was considered different sets of tests based on a real dataset. Such dataset was provided by *TAP Air Portugal*¹ and was composed by 1965 flights, 30 maintenance tasks to be performed by 42 different tails. In fact, the considered dataset belongs to a well-known airline company which usually operates in a hub-and-spoke network.

¹<https://www.flytap.com/>

As this initial dataset was too large to perform the desired tests, it was necessary to define some rules selecting partial datasets data while keeping the proportions of a real scenario. Therefore, this data was selected based on the fact that for the original dataset on average a tail should perform two flights per day. To perform the tests, two different subsets were considered. On the one hand, dataset A was composed by a group of 10 tails and a set of 266 flights. On the other hand, dataset B, a bit larger that included 442 flights.

In Table 2 we present the values of the tuning parameters used during the performed tests. Since no rules were possible to set on the values of these parameters, they were found empirically.

Table 2: Tuning parameters' values for QUBO model.

γ	η	λ	τ	ϕ	ψ
5	8.5	3	4	4	0.3

The main goal of the tests was to compare the solutions obtained by solving the proposed implementation using three different solvers, one classical and two hybrid, all part of D-wave's toolchain. The chosen classical solver was the *SimulatedAnnealingSampler*² whereas the hybrid solvers were *D-Wave Hybrid Solver Service (HSS)*³ and *KerberosSampler*⁴. Both hybrid solvers made use of a D-Wave 2000Q quantum annealer, composed by 2048 qubits, which QPU is organized in a Chimera topology.

For doing such analysis we run for each test case (i.e, each dataset and each solver), 10 shots to have a considerable accuracy on the obtained results.

5.1 Finding a Feasible Solution

As presented in Table 3, the probability of each one of the solvers to find a valid solution is high. However for the biggest dataset such probability becomes really low when solving the problem using *SASampler* and *Kerberos*.

Table 3: Probability of finding valid solutions.

Dataset	Prob. valid solutions		
	<i>SASampler</i>	<i>Kerberos</i>	<i>HSS</i>
A	1	1	0.8
B	0.2	0.3	1

²<https://docs.ocean.dwavesys.com/projects/neal>

³https://docs.dwavesys.com/docs/latest/doc_leap_hybrid.html

⁴<https://github.com/dwavesystems/dwave-hybrid/blob/master/hybrid/reference/kerberos.py>

5.2 Solutions' Distribution

Since the main goal of solving the Tail Assignment Problem is not just to get a valid solution but also a solution with minimum cost, it is also important to analyse the quality of the the obtained solutions from the three solvers. When considering complex problems, the possibility of finding a good but still non-optimal solution is relevant. Therefore, we analysed the distribution of the feasible solutions obtained for each one of the datasets when solved using the three solvers aforementioned.

Regarding dataset A, as presented in Figure 2, for *SASampler* and *Kerberos* some outliers were identified. In fact, the majority of the valid solutions obtained had similar OF costs and, therefore, the solution with minimum OF cost obtained for *SASampler* and the solution with maximum OF cost obtained for *Kerberos* were considered outliers. For *HSS*, the solutions found shown a higher variability with 25% of the solutions having an OF cost of less than 29.334.216 units of cost (UC). It is also relevant to note that, for *HSS*, the most expensive solutions found have an associated OF cost considerably higher when compared with the most expensive solution obtained by any of the other two solvers. Summing up, *HSS* has a bigger probability of finding cheaper solutions when compared with the other solvers used. Nonetheless, this solver also has some drawbacks as it found a big number of expensive solutions.

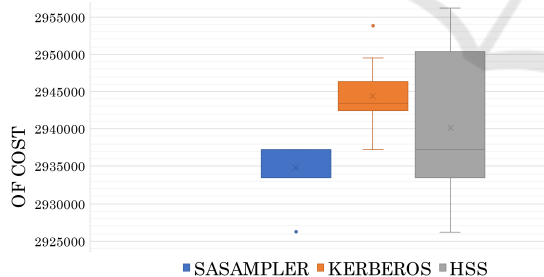


Figure 2: Box plot of the OF costs regarding the obtained solutions from the three solvers for dataset A.

Finally, regarding dataset B, we tried to verify if the same previous conclusions were also true for bigger datasets. As presented in Figure 3, for *SASampler* and *Kerberos*, the obtained solutions are highly concentrated in terms of OF costs indicating that their OF costs are similar. This may have happened because both of these solvers were only able to find a few valid solutions. Furthermore, for *SASampler*, the solutions found had lower OF costs when compared with the solutions found by *Kerberos*. Regarding *HSS*, it was able to find valid solutions for all the 10 shots. Fol-

lowing the same trend verified for the other datasets, *HSS* found solutions with a big range of OF costs. Although the most expensive solution found for dataset B was obtained by this solver, 50% of the solutions found had a lower OF cost when compared with any of the solutions obtained by *Kerberos*. Comparing *HSS* with *SASampler*, both of them were able to find the minimum cost solution. However, as previously analysed, *SASampler* was able to find a valid solution only 20% of the times. Thus, it may indicate that using *SASampler* for big datasets can lead to multiple unfeasible solutions.

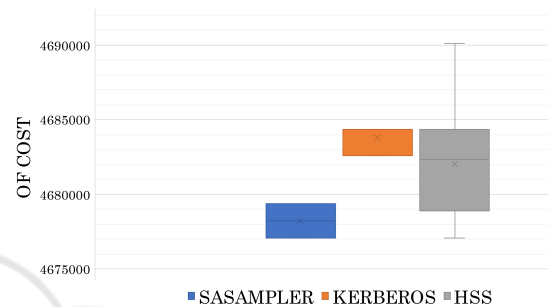


Figure 3: Box plot of the OF costs regarding the obtained solutions from the three solvers for dataset B.

Summing up, the results shown that when comparing the classical solver *SASampler* with the hybrid solvers *Kerberos* and *HSS* none of the solvers revealed being perfect for finding the best solutions. Nonetheless, they diverge on the performance when solving the problem considering different datasets. In fact, although all the solvers were able to find feasible solutions for all the datasets, none of them was able to find the solution with minimum total cost in the majority of the shots. *Kerberos* was the one that performed the worst, as the distribution of the results obtained were highly expensive for the majority of the runs and therefore were not satisfactory. *SASampler* revealed better results, being able to find the minimum energy solution at least once for the datasets. For bigger datasets, this solver revealed not to be so useful to solve this QUBO model as the majority of the solutions found were either unfeasible or considerably more expensive than the cheapest solution found. Finally, regarding *HSS*, it was the solver that performed the best overall. However, it is relevant to note that some of the solutions found by this solver were highly expensive which represent a drawback on the confidence on the usage of this solver to this problem.

6 CONCLUSIONS AND FUTURE WORK

We have presented a formulation of the Tail Assignment Problem as a QUBO model to be solved using QA based on the current commercial solution developed by D-Wave Systems. Our results demonstrate that using an hybrid solver for the proposed QUBO model may represent a considerable advantage in the probability of finding valid non-expensive solutions when compared with a classical solver. Nonetheless, it is relevant to note that, even though it may show encouraging results, quantum computation is in an early stage and, therefore, the current limitations do not allow scaling it to complete real-world datasets. Throughout this study, we opted to implement multiple simplifications to narrow the scope of the problem in analysis. A non-consideration of minor maintenance tasks is not realistic as they have to occur in a real scenario. Furthermore, a robust approach may be a pivotal achievement as flight delays are frequent and tight schedules can be significantly affected by that. Additionally, to understand the effectiveness in a deeper level of the proposed modelling technique, when applied to multiple solvers using different scenarios, it would be important to run more tests using different datasets. Finally, as *HSS* revealed to perform better for solving this problem than using only a classical algorithm such as SA, further studies on hybrid solvers could be relevant for a better understanding on the real advantage of using such technique to solve complex optimisation problems.

REFERENCES

- Albash, T. and Lidar, D. A. (2018). Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1).
- Chapuis, G., Djidjev, H., Hahn, G., and Rizk, G. (2019). Finding maximum cliques on the d-wave quantum annealer. *Journal of Signal Processing Systems*, 91(3):363–377.
- Cruz-Santos, W., Venegas-Andraca, S. E., and Lanzagorta, M. (2019). A QUBO Formulation of Minimum Multi-cut Problem Instances in Trees for D-Wave Quantum Annealers. *Scientific Reports*, 9(1):1–12.
- Froyland, G., Maher, S. J., and Wu, C. L. (2013). The recoverable robust tail assignment problem. *Transportation Science*, 48(3):351–372.
- Grönkvist, M. (2005). *The tail assignment problem*. PhD thesis, Chalmers University of Technology and Göteborg University.
- Ikeda, K., Nakamura, Y., and Humble, T. S. (2019). Application of quantum annealing to nurse scheduling problem. *Scientific Reports*, 9(1):12837.
- Kadowaki, T. and Nishimori, H. (1998). Quantum annealing in the transverse ising model. *Phys. Rev. E*, 58:5355–5363.
- Lucas, A. (2014). Ising formulations of many NP problems. *Frontiers in Physics*, 2(February):1–14.
- Montoito, F. (2016). Application of the Simulated Annealing with Adaptive Local Neighborhood Search to the Tail Assignment Problem The Case Study of TAP. Master's thesis, Instituto Superior Técnico, Universidade de Lisboa.
- Neukart, F., Compostella, G., Seidel, C., von Dollen, D., Yarkoni, S., and Parney, B. (2017). Traffic flow optimization using a quantum annealer. *Frontiers in ICT*, 4:29.
- Papalitsas, C., Andronikos, T., Giannakis, K., Theocharopoulou, G., and Fanarioti, S. (2019). A QUBO model for the traveling salesman problem with time windows. *Algorithms*, 12(11):1–18.
- Ray, P., Chakrabarti, B. K., and Chakrabarti, A. (1989). Sherrington-kirkpatrick model in a transverse field: Absence of replica symmetry breaking due to quantum fluctuations. *Phys. Rev. B*, 39:11828–11832.
- Ruther, S., Boland, N., Engineer, F. G., and Evans, I. (2017). Integrated aircraft routing, crew pairing, and tail assignment: Branch-and-price with many pricing problems. *Transportation Science*, 51(1):177–195.
- Stollenwerk, T., Lobe, E., and Jung, M. (2019). Flight Gate Assignment with a Quantum Annealer. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11413 LNCS:99–110.
- Tran, T. T., Do, M., Rieffel, E. G., Frank, J., Wang, Z., O’Gorman, B., Venturelli, D., and Beck, J. C. (2016). A hybrid quantum-classical approach to solving scheduling problems. In *SOCS*.
- Ushijima-Mwesigwa, H., Negre, C. F. A., and Mniszewski, S. M. (2017). Graph partitioning using quantum annealing on the d-wave system. In *Proceedings of the 2nd International Workshop on Post Moores Era Supercomputing*, page 22–29, New York, USA. Association for Computing Machinery.
- Venturelli, D., Marchand, D. J. J., and Rojo, G. (2016). Job Shop Scheduling Solver based on Quantum Annealing. arXiv:1506.08479 [quant-ph].
- Vikstål, P., Grönkvist, M., Svensson, M., Andersson, M., Johansson, G., and Ferrini, G. (2019). Applying the Quantum Approximate Optimization Algorithm to the Tail Assignment Problem. arXiv:1912.10499 [quant-ph].
- Yadav, K. K. (2017). An Effective and Efficient Solution to Tail Assignment Problem. *International Journal of Advanced Research in Computer Science and Software Engineering*, 7(4):378–386.