Optimal Seating Assignment in the COVID-19 Era via Quantum Computing

Optimal seating assignment in the COVID-19 era via Quantum Computing

Ilaria Gioda*, Davide Caputo†, Edoardo Fadda*, Daniele Manerba‡,
Blanca Silva Fernández†, and Roberto Tadei*

*Department of Control and Computer Engineering, Politecnico di Torino, 10129 Torino, Italy
Email: ilaria.gioda@studenti.polito.it, {edoardo.fadda, roberto.tadei}@polito.it

†Data Reply s.r.l., 10126 Torino, Italy
Email: {da.caputo, b.silvafernandez}@reply.it

‡Department of Information Engineering, Università degli Studi di Brescia, 25123 Brescia, Italy
Email: daniele.manerba@unibs.it

***Abstract*—In recent years, researchers have oriented their studies towards new technologies based on quantum physics that should allow the resolution of complex problems currently considered to be intractable. This new research area is called Quantum Computing. What makes Quantum Computing so attractive is the particular way with which quantum technology operates and the great potential it can offer to solve real-world problems. This work focuses on solving combinatorial optimization problems, specifically assignment problems, by exploiting this novel computational approach. A case-study, denoted as the Seating Arrangement Optimization problem, is considered. It is modeled through the Quadratic Unconstrained Binary Optimization (QUBO) paradigm and solved through two tools made available by the *D-Wave Systems* company, QBSolv and a quantum-classical hybrid system. The obtained experimental results are compared in terms of solution quality and computational efficiency.**

I. Introduction

Combinatorial Optimization (CO) is one of the most studied research fields in the area of optimization. The application of this research area extends to many sectors and more and more researches are active to model and solve effectively and efficiently the problems belonging to this category. Among others, one of the most recent and innovative modelling approaches allowing to formulate a CO problem is the so-called Quadratic Unconstrained Binary Optimization (QUBO) paradigm.

During the last few decades, many studies have been carried out regarding this mathematical formulation, also known as Unconstrained Binary Quadratic Programming (UBQP). An extensive survey on UBQP researches can be found in [14], in which the authors group in chronological order a good percentage of works addressing UBQP problems up to 2014. Given the potential of this quadratic formulation, Anthony et al. in [2] focus on the quadratization of arbitrary functions, by reporting a systematic study on the lower and upper bounds on the number of additional auxiliary variables needed to trace back to the quadratic pseudo-Boolean optimization case.

The literature contains several works that are dedicated on the formulation of traditional problems in the QUBO form to understand the benefits of this formulation; many of them try to model problems directly in the QUBO form, while others provide for the re-cast of already studied and implemented problems from their original formulations to this novel form. The most famous case in literature is the one conducted by Lucas [15]. He studies how to rewrite many well-known problems from their classical form to the QUBO and Ising one (an equivalent formulation, mainly used in physics). Another example is proposed by Alidaee et al. in [1], where the authors show how the QUBO paradigm can effectively be used to formulate and solve set packing problems. Moreover, the authors of [19] propose the formulation of the TSP with Time Windows problem in the QUBO version.

Subsequently, researches have been oriented towards the identification of efficient methods for solving problems written in this formulation. For instance, Wang et al. in [22] propose the first two tabu search with path relinking algorithms for solving UBQP problems by generating new solutions through the exploration of trajectories that connect high-quality solutions. Among the various approaches for solving combinatorial optimization problems in the QUBO form, in recent years researchers have begun to be oriented towards a new computational frontier, as the *Quantum Computing*. Borowski et al. in [5] outline four quantum annealing-based algorithms to solve Vehicle Routing (VRP) and Capacitated Vehicle Routing (CVRP) problems and compare their performance with well-known classical algorithms. A more practical case-study is Volkswagen's Traffic Flow Optimization [18], which deals with a real-world application for managing and minimizing congestions on-road segments within a particular city. The authors of this work, by considering the number of cars and their routes, formulated the problem as QUBO and solved it using quantum annealing technologies.

This paper focuses on the analysis of this new computational approach, specifically for the resolution of assignment problems. We analyzed the Seating Arrangement Optimization problem as a case-study, which was first formulated as a QUBO problem and then solved through the use of some tools made available by *D-Wave Systems*, a Canadian company specializing in quantum computing.

The remaining part of this paper is organized as follows. Section II outlines the basic concepts of the novel computational approach of quantum computing, and, specifically, introduces the quantum annealing. Section III reports details of the quantum technologies offered by D-Wave Systems to solve combinatorial optimization problems. First, the Quadratic Unconstrained Binary Optimization (QUBO) paradigm is described and, then, the dedicated solvers for problems in this particular form are shown. Section IV presents the case-study considered, the Seating Arrangement Optimization problem. The problem is first described and modeled as a quadratic problem, then an equivalent QUBO formulation is derived. Section V describes and compares the computational results of the experimental analysis. Section VI provides conclusions and a brief discussion on possible future works.

II. Quantum Computing approach

In the recent years, research has been directed towards studying new technologies that enable a new computational approach for solving complex problems from different areas.

Quantum computing is an innovative research field for processing information and algorithms based on a technology that exploits quantum physics instead of the classical one. The fundamental element on which quantum technologies are based is the quantum bit, or *qubit*. Like a classical computer bit, a qubit can assume one of the two binary values 0 or 1, but the novelty of this element lies in the fact that, during the elaboration process, the qubit finds itself in a third further state, called *superposition*. In this part of the process, it can assume any state, but its final value is established only during the measurement process, which determines its collapse to one of the classical states. Nowadays, several companies such as *Google*, *IBM*, *NASA*, and *Rigetti* are involved worldwide in the quantum computing research field. They have been building and, only in recent years, granting access to quantum technology for the resolution of small-to-medium-sized algorithms. In the next few years, quantum computers are expected to surpass the traditional ones in solving classical complex algorithms, which are now intractable or requiring too much time to be executed.

Research has been conducted in several directions, as the technologies are different and therefore require different paradigms. However, we can identify two main categories of quantum computers, which differ in the type of structure and the applications for which they are designed [16]. The first category includes universal or gate-model quantum computers. The systems belonging to this class of quantum machines are equipped with a particular circuitry that manipulates the qubits' state. Due to the many limitations concerning the technology they are built with, universal quantum computers offers only several qubits of the order of dozens, and currently, they are used to solve small instances of problems in various areas such as machine learning [20] and chemistry [17]. The second type of quantum computers, named quantum annealers, is equipped with hardware with a higher number of qubits (in the order of thousands) and combinatorial optimization is the most important type of elaboration they can manage.

Quantum annealers owe their name to the process called *quantum annealing* and that physically takes place within them and which, by exploiting the intrinsic effects of quantum physics, can be used to solve a specific combinatorial optimization problem. In detail, quantum annealing is a heuristic way to solve NP-hard problems by finding the global minimum of a function with many local minima describing a physical system's energy. The search process is based on quantum fluctuations. Quantum annealing exploits the physical concept that everything in nature tends to evolve towards the equilibrium. Since quantum physics follows this reasoning, the quantum process is used to find equilibrium states corresponding to optimal solutions for the problem under observation.

The quantum annealing and the quantum systems implementing it are based on characteristics that make their computation principle unique and innovative. As previously introduced, the first and most important is the qubit. At the beginning of the annealing process, each qubit finds itself in the superposition state, and its value can assume with equal probability the 0 or 1 state. This probability can be modified and directed towards one of the two states following an external magnetic field, called *bias*. The second important concept exploited in quantum annealing is the *entanglement*, i.e., a linking process that involves qubits, which do not work alone but take part of a more complex process. Entanglement binds qubits together, making the state of each qubit dependent from one state of some other qubits. At the hardware level, this linking is done through a procedure called a *coupling*. The purpose of this procedure is to create a sort of single entity formed by pairs of qubits so that these two assume either the same value (both 0 or 1) or the opposite one (one 0 and the other 1). Therefore, this unique object has four states (00, 01, 10, 11), whose energies depend on the bias of the two single qubits that compose it and the coupling strength between them.

What a user of a quantum computer is allowed to do is programming the biases and the couplers strengths in such a way as to create an energy landscape which then allows to be sure to get a solution for the optimization problem when reaching the equilibrium during the annealing process. Thanks to this way of acting, quantum annealers' technologies are explicitly designed to solve complex combinatorial optimization problems.

III. Quantum Computing solvers

The leader company that work with quantum annealers is *D-Wave Systems Inc.*. In particular, this organization deals with building and studying quantum technologies and, for some years, has allowed external people to use their quantum annealers to solve specific commercial problems, especially combinatorial optimization ones. In this section, we show the particular type of problems that quantum annealers can solve (QUBO problems) and, in particular, the tools that the quantum company makes available for their resolution.

A. Quadratic Unconstrained Binary Optimization (QUBO)

Quantum annealers are designed to solve complex combinatorial optimization problems in a particular formulation, the Quadratic Unconstrained Binary Optimization (QUBO) one. The goal of a QUBO model is to find an optimal solution by minimizing an objective function in the form

$$\min \sum_{i \in N} Q_{i,i} x_i + \sum_{i,j \in N | i \neq j} Q_{i,j} x_i x_j \tag{1}$$

where

- N is the set of indexes associated with a variable;
- x_i is the i-th binary variable of the problem;
- $Q_{i,i}$ is the diagonal (or linear) cost coefficient for the i-th variable;
- $Q_{i,j}$ is the off-diagonal (or quadratic) cost coefficient for the association of the i-th variable with the j-th one.

The first characteristic of a QUBO problem is the kind of variables that describe it. They are binary and appear in the objective function both in the linear form, i.e., individually x_i, and in the quadratic form, i.e., one multiplied by the other $x_i x_j$. Alternatively, since for binary variables it holds $x_i = x_i^2$ for any $i \in N$, a QUBO problem can be described in matricial notation as follows

$$\min_{\mathbf{x} \in \{0,1\}^{|N|}} \mathbf{x}^T Q \mathbf{x} \tag{2}$$

where $\mathbf{x}$ is a column vector of binary variables of size $|N|$ and Q an upper-triangular $|N| \times |N|$ matrix, called QUBO matrix.

Clearly, not all the optimization problems come in this form. However, many of them can be rewritten as a QUBO model. The constraints identified for the problem must be readjusted and converted into penalties to form the actual objective function (1) that has to be minimized. Specifically, as in classical Lagrangean relaxations, the purpose of these penalties is to prevent the optimizer from choosing solutions that violate the constraints. They involve the addition of a positive quantity, therefore not favorable to the minimization objective in the case of infeasible solutions [13]. Some standard ways of creating this translation from classical constraints can be found in [13] and are reported in Table I. Starting

TABLE I
SOME KNOWN PENALTY TERMS. ALL THE VARIABLES ARE SUPPOSED TO BE BINARY AND p IS A POSITIVE SCALAR VALUE (FROM [13])

Classical Constraint	**Equivalent Penalty**
$x + y \leq 1$	$p(xy)$
$x + y \geq 1$	$p(1 - x - y + xy)$
$x + y = 1$	$p(1 - x - y + 2xy)$
$x \leq y$	$p(x - xy)$
$x_1 + x_2 + x_3 \leq 1$	$p(x_1x_2 + x_1x_3 + x_2x_3)$
$x = y$	$p(x + y - 2xy)$

from the obtained objective function, it is then possible to construct the QUBO matrix Q. This matrix maintains, in the form of constant values, the numerical relationships between all variables, also originating from the constraints to which the problem must be subjected. This element is also essential at the computational level since its dimensions impact the performance of the problem itself in terms of quality and solution time (see, e.g., [6] or [3]). The QUBO matrix comes in the form of a square $|N| \times |N|$ matrix and can be found both as symmetric and as upper-triangular matrix [13] (in the latter case, the second summation of (1) must be restricted to $i < j$).

Interesting enough, it is possible to identify a direct correspondence between the physical implementation of a *Quantum Processing Unit* (QPU), the hardware processor of a quantum annealer, and a QUBO problem. Given a generic QUBO model, each binary variable is represented into the QPU by a qubit, whose possible states are called *spin up* and *spin down*, while the linear and quadratic coefficients $Q_{i,i}$, and $Q_{i,j}$ correspond respectively to the biases and coupling strengths on a QPU. Despite this correspondence, usually, QUBO problems cannot be directly mapped to the QPU due to the physical interconnections between qubits and the number of qubits available on the hardware (in general, much smaller than the actual number of binary variables in a realistic-size problem). So, in practice, they undergo a translation process called *Minor Embedding*, which allows the mapping of each logical variable of the problem into a chain of physical qubits.

B. QBSolv

Over the last few years, *D-Wave Systems Inc.* has made available useful tools involving quantum technologies to solve optimization problems written in the QUBO formulation. In particular, it has made available a software development kit, the *Ocean SDK*, containing a series of open-source Python tools for solving hard problems with quantum computers [9]. One of these tools, useful in our case, is *QBSolv* [10]. QBSolv is an open-source solver released in January 2017, which runs on the CPU just like the traditional solvers. Its goal is to solve large QUBO problems with high connectivity. The solver strategy consists in partitioning significantly-large QUBO problems into smaller components and applying a specified sampling method (the classical Tabu Search algorithm, by default) independently to each of these pieces to find the minimum value required for the optimization. Further technical details on QBSolv can be found in [4].

C. D-Wave Quantum-Classical Hybrid solver

Another important tool made available by *D-Wave Systems Inc.* allows to submit and solve a problem modeled as QUBO on a remote quantum computer. To do this, in 2018, the computing company made available to users a cloud service, the *D-Wave's Leap*, and a set of Python APIs, the Solver API (SAPI), that allow any developer to access and submit any problem to the *D-Wave Quantum System*.

At the time of writing this paper, there are two types of solvers made available by the D-Wave Systems company: a solver that directly uses a QPU (to be chosen between an Advantage system and a D-Wave 2000Q lower-noise system)

[11] and one that exploits a quantum-classical hybrid system [7]. Both types of sampler accept as input a so-called Binary Quadratic Model (BQM), a specific format that can be constructed starting from the Q matrix of a common QUBO problem. Since quantum technologies are still growing due to the noise and the limited number of qubits available, using a hybrid system can be required for solving realistic and large problems. Specifically, a quantum-classical hybrid system uses a computational approach that exploits classical and quantum resources. It analyzes and subdivides significant BQM problems using classical technology to break them up into sub-problems and establish which of them should be solved with classical algorithms and can be effectively performed on a QPU by interacting directly with it for their resolution [21].

IV. A case-study: The Seating Arrangement Optimization Problem

The considered case-study focuses on passenger transport on high-speed trains considering the Italian Government's new regulations on social distancing due to the COVID-19 pandemic.

The railway companies have currently adapted their passenger positioning strategies by embracing a seating arrangement as a "checkerboard" pattern, i.e., with the allocation of passengers to alternate seats, to counter the spread of the COVID-19 virus. Nevertheless, with the adoption of this strategy, the filling capacity of the wagons has dropped to the 50% of the total capacity, leading to a drastic reduction in the high-speed rail operators' earnings due to the mismatch between the costs necessary for the activation of the railway transport lines and the revenues obtained from ticket sales. From now on, we will denote the examined case study as the Seating Arrangement Optimization problem.

In the research conducted in this paper, we investigate a new seat allocation criterion for allocating people on high-speed trains by exploiting the social relationships between them. We aim to maximize the number of transported passengers, still complying with the rules for protecting the customers' health.

The problem has been modeled first as a non-linear program, then through the novel QUBO formulation, and finally solved using the tools provided by *D-Wave Systems*, i.e., the QBSolv and the Quantum-Classical Hybrid solver.

A. Problem description

The objective of the Seating Arrangement Optimization problem is to fill the train wagon as much as possible within the restrictions on social distancing due to the COVID-19 health emergency. Still, it aims to maximize the number of passengers belonging to the same family or living group in adjacent seats. Although the focus of the problem can be extended to the entire train, the study refers to only one wagon. Then, for a multi-wagon train the procedure will be run for each wagon separately. Furthermore, we assumed a static situation: just one train segment, i.e., trip between two adjacent stations, is considered so that the number of passengers and their social relationships are known beforehand without any changes during the travel.

Some fundamental elements characterize the Seating Arrangement Optimization problem. A set of passengers that has to be transported on a high-speed train is given. During the ticket reservation procedure, each passenger is associated with a unique identifier, the booking ID, which can be shared or not with other passengers. The important assumption of the problem is that people with the same booking ID belong to the same family or living group. This condition, therefore, assumes they can be excluded from the social distancing impositions prescribed by the regulations against the spread of the COVID-19 virus. A high-speed train's wagon is then considered. The wagon has a certain number of seats. Each seat is represented by a pair of coordinates, a row and a column number, which collocate it into a sort of grid. A graphic representation of a typical high-speed train's wagon can be seen in Fig. 1.

When modeling the problem, it is necessary to consider the following constraints:

- allocation of one and only one seat to each one of the considered passengers (avoid that a passenger has more than one seat assigned to him);
- allocation of one passenger at most to each seat (avoid different passengers being assigned the same seat);
- allocation of not adjacent (in front/behind/left/right) seats to people belonging to different families (identified by different booking IDs).

B. Mathematical programming formulation

In this section, we provide a natural non-linear programming formulation of the Seating Arrangement Optimization problem in order to formally define it.

Let us consider the following sets and parameters:

- $R = \{1, 2, \ldots, r_{max}\}$: set of seats row numbers;
- $C = \{1, 2, \ldots, c_{max}\}$: set of seats column numbers;
- K: set of booking IDs;
- n_k: number of passengers with the same booking ID $k \in K$.

Moreover, let us define the variable

$$x_{(r,c),k} := \begin{cases} 1 & \text{if a passenger with booking ID } k \text{ is} \\ & \text{assigned to seat with row and column } (r,c) \\ 0 & \text{otherwise} \end{cases}$$

for each row $r \in R$, column $c \in C$, and booking ID $k \in K$.

Then, a natural quadratic programming model for the problem can be stated as:

$$\max \sum_{k} \sum_{(r,c)} x_{(r,c),k} \cdot x_{(r+1,c),k} + \\ + \sum_{k} \sum_{(r,c)} x_{(r,c),k} \cdot x_{(r,c+1),k} \quad (3)$$

subject to

$$\sum_{(r,c)} x_{(r,c),k} = n_k, \quad k \in K \quad (4)$$

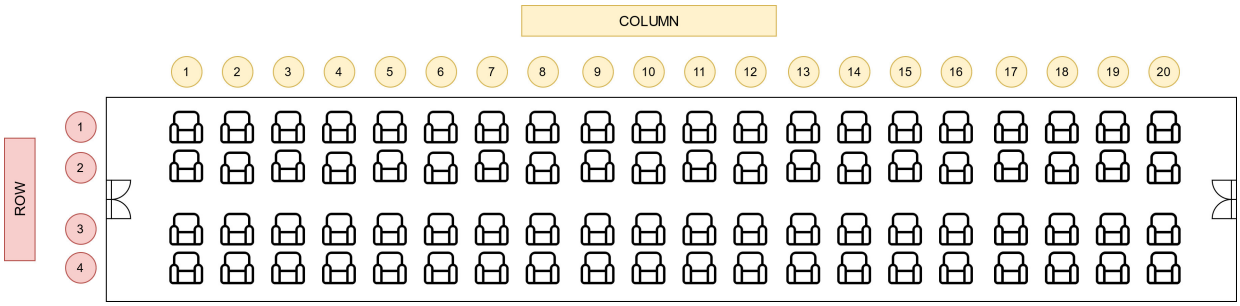

Fig. 1. Graphical representation of seats with row and column numbers inside a high-speed train's wagon

$$\sum_{k} x_{(r,c),k} \leq 1, \quad r \in R,\ c \in C \tag{5}$$

$$x_{(r,c),k} \cdot x_{(r+1,c),k'} = 0, \quad r \in R \setminus \{r_{max}\},\ c \in C,\ k, k' \in K, k \neq k' \tag{6}$$

$$x_{(r,c),k} \cdot x_{(r,c+1),k'} = 0, \quad r \in R,\ c \in C \setminus \{c_{max}\},\ k, k' \in K, k \neq k' \tag{7}$$

$$x_{(r,c),k} \in \{0,1\}, \quad r \in R,\ c \in C,\ k \in K. \tag{8}$$

The objective function (3) maximizes the number of passengers with same booking ID assigned to adjacent seats. Constraints (4) state that each passenger with a certain booking ID is assigned to one seat, while constraints (5) state that each seat is assigned to at most one passenger with a certain booking ID. Constraints (6) ensure that two seats, one next to the other (in the same column), are not assigned to passengers with different booking IDs, while constraints (7) ensure that two seats, one in front of the other (in the same row), are not assigned to passengers with different booking IDs. Finally, binary conditions on the variables are stated in (8).

C. QUBO formulation

Since the QUBO paradigm asks for an unconstrained model, as the one in (2), the constraints (4)-(7) and the cost function (3) are relaxed and aggregated into a single objective function through non-negative parameters λ's, to be calibrated (see later). In particular, we chose to set these parametric coefficients as numerical, and to associate each of them with a specific group of constraints presented in model (3)–(8). We decided to adopt this modeling choice in order to minimize the number of λ parameters needed, as they represent a non-negligible obstacle during the model calibration.

To do this relaxation, we built a penalty term for each of the identified constraints by following the approach from [13]. Hence, a QUBO formulation for the Seating Arrangement Optimization problem becomes:

$$\min \lambda_A H_A + \lambda_B H_B + \lambda_C H_C + \lambda_D H_D - H_E \tag{9}$$

where

- the penalty term associated with constraints (4) is
$$H_A = \sum_{k} (n_k - \sum_{(r,c)} x_{(r,c),k})^2$$
- the penalty term associated with constraints (5) is
$$H_B = \sum_{(r,c)} \sum_{k,k'} x_{(r,c),k} \cdot x_{(r,c),k'}$$
- the penalty term associated with constraints (6) is
$$H_C = \sum_{(r,c)} \sum_{k,k'} x_{(r,c),k} \cdot x_{(r+1,c),k'}$$
- the penalty term associated with constraints (7) is
$$H_D = \sum_{(r,c)} \sum_{k,k'} x_{(r,c),k} \cdot x_{(r,c+1),k'}$$
- the penalty term associated with the objective function (3) is
$$H_E = \sum_{k} \sum_{(r,c)} x_{(r,c),k} \cdot x_{(r+1,c),k} + \sum_{k} \sum_{(r,c)} x_{(r,c),k} \cdot x_{(r,c+1),k}$$

Note that, unlike the other penalties, a squaring has been introduced in H_A as it is necessary to be able to grasp the relationship between the values assumed by different variables within the solution.

Starting from (9), the Q matrix of model (2) has been derived. To do that, we need to identify the relationship between the problem's variables, i.e., the multiplicative coefficients of the equation (1). First of all, the single QUBO terms of the function (9) need to be expanded. Then, after the coefficients have been found, they are multiplied by the parametric coefficients λ_A, λ_B, λ_C and λ_D, whose purpose is to give more or less weight to each QUBO penalty such that the constraints are imposed when searching for the solution.

D. Parametric coefficients calibration

We now provide some indications on how the calibration of the numerical parametric coefficients λ_A, λ_B, λ_C, and λ_D has been carried out.

Expressed as pseudo-code, the high-level steps that has been followed can be found in Algorithm 1.

Algorithm 1 λ's coefficients calibration

Initialize λ_A, λ_B, λ_C, and λ_D with small real random values and choose a small-sized instance of the problem;
repeat
  repeat
    Run the code, get a particular solution from the solver (QBSolv) and check it
    if infeasible solution, i.e., solution that does not respect at least one of the constraints (4)-(7) **then**
      Check the solution's energy
      if the solution's energy > the energy of a feasible solution already found **then**
        The solver has failed to find the solution with the lowest energy
      else if the solution's energy < the energy of a feasible solution already found **then**
        The current configuration of the λ parameters is not working
        Increase the λ parameter for which the solution violates the associated constraint
      end if
    end if
  until only feasible solutions for the current instance of the problem are found
  Increase the size of the considered instance
until only feasible solutions are found

V. Computational results

The model has been implemented via software through the Python programming language.

This section reports the results obtained by executing several instances of the Seating Arrangement Optimization problem written as QUBO using the two tools offered by D-Wave Systems, the QBSolv, and D-Wave Leap's cloud-based quantum-classical hybrid solver (from now on referred to as D-Wave Hybrid Solver). Initially, the problem size in terms of number of variables is reported. Then, the two solvers are compared in terms of optimal solutions and computational times.

An ad-hoc data-set containing simulated test instances about seats, passengers, and bookings was created for the performed experiments. The input that we provided to our QUBO model has been created based on indicative estimate of realistic data of a high-speed train. In particular, it was decided to use a wagon consisting of 80 seats, which have been placed in a 4×20 grid, made up of 4 horizontal (the rows) and 20 vertical (the columns) rows. Taking as a reference a credible number of passengers for a high-speed train, 1000 passengers have been created, but only a small subset of them was used for our restricted experimental analysis. In particular, for the experiments reported in Table II, the maximum number of people that has been tested is 52. Since it was necessary to associate a specific booking ID to each passenger, we decided to use 300 distinct booking IDs to make a reasonably homogeneous assignment. The final range of assigned booking IDs is as follows:

- 290 booking IDs
- minimum number of passengers with the same booking ID: 1
- maximum number of passengers with the same booking ID: 8

All the experiments have been carried out on a desktop computer with a 1.8 GHz Intel Core i7-8550U processor.

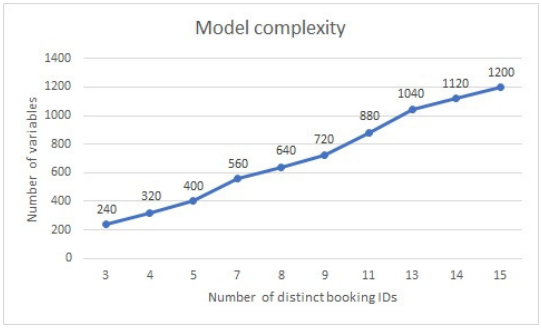

Fig. 2. Number of variables in the QUBO model

A. QUBO model size

The QUBO formulation involves a not-so-high number of variables, thus effectively modeling complex optimization problems. In Fig. 2 we can see the number of variables vs. the number of booking IDs of the considered problem instance. It can be noted that, by maintaining the number of available seats fixed at 80, the total number of variables increases almost linearly as the number of considered booking IDs grows.

B. Quality of the solutions

The quality of the *D-Wave Systems* solvers is now analyzed. The numerical results for the Seating Arrangement Optimization problem's various instances can be seen in Table II. For each problem instance ("Seats", "Passengers", "Distinct booking IDs" columns), the optimal solution with the minimum energy (i.e., the minimum value of the expression (9), reported in the "Total minimum energy" column) that each solver ("Solver" column) was able to find is indicated. Specifically, for each solution, the number of passengers allocated to seats inside the train wagon (fifth column) and the number of people with the same booking ID correctly assigned to adjacent seats (sixth column) are reported.

After having calibrated the λ parameters in model (9), by using the Python APIs of the Ocean SDK, the two solvers were used to solve different instances of the analyzed problem.

The two optimizers seem to perform well, most of the time reaching the goal of allocating people with the same booking ID to adjacent seats. Furthermore, an improvement compared to the passenger transport's current situation has been achieved. Both solvers manage to find at least an acceptable seating arrangement up to 15 booking IDs for a total of 51 passengers, bringing therefore to have a filling percentage of the seats up to 63,75% (instead of the classical 50%).

TABLE II
OPTIMAL SOLUTIONS OBTAINED BY RUNNING THE QUBO MODEL INSTANCES WITH THE *D-Wave Systems* SOLVERS

Seats	Passengers	Distinct booking IDs	Solver	Nb. of passengers with an assigned seat	Nb. of passengers with same booking ID assigned to adjacent seats	Total minimum energy
80	11	3	QBSolv	11	10	-464.300
			D-Wave Hybrid	10	11	-464.300
80	16	4	QBSolv	16	15	-682.800
			D-Wave Hybrid	16	15	-682.800
80	19	5	QBSolv	19	18	-762.000
			D-Wave Hybrid	19	18	-762.000
80	23	7	QBSolv	23	21	-849.400
			D-Wave Hybrid	23	21	-849.400
80	28	8	QBSolv	28	26	-1067.900
			D-Wave Hybrid	28	26	-1067.900
80	34	9	QBSolv	34	32	-1382.000
			D-Wave Hybrid	34	32	-1382.000
80	39	11	QBSolv	39	37	-1496.700
			D-Wave Hybrid	39	37	-1496.700
80	44	13	QBSolv	44	41	-1646.900
			D-Wave Hybrid	44	41	-1646.900
80	50	14	QBSolv	50	45	-1947.500
			D-Wave Hybrid	50	47	-1958.300
80	51	15	QBSolv	51	46	-1953.000
			D-Wave Hybrid	51	47	-1961.100

For most of the instances, the D-Wave Hybrid Solver finds solutions with the same energy as those found by QBSolv optimizer. This means that the solver running on the CPU performs well in terms of solution quality, even without quantum hardware usage. However, there are two cases, i.e., the ones corresponding to the instances with 14 and 15 distinct booking IDs (respectively 50 and 51 passengers) where D-Wave Hybrid Solver finds two lower energy and better solutions than those found by the QBSolv sampler.

C. Efficiency of the solvers

The last comparison refers to the computational times of the two solvers.

In Table III, each row reports the times required by the QBSolv ("QBSolv" column) and by the D-Wave Hybrid ("LeapHybridSampler" column) optimizers to solve a specific problem instance formed by a certain number of seats, passengers and booking IDs ("Seats", "Passengers", "Distinct booking IDs" columns). As we can see in this table, the time increases as the size of the considered instance of the Seating Arrangement Optimization problem grows. Moreover, both solvers require a very limited amount of time (just some seconds) to solve the problem. The difference between them lies in the way in which they work. The first optimizer, QBSolv, works locally on the CPU. The second one, D-Wave Hybrid Solver, requires remote access via the Internet to a physically remote system shared between multiple users. For this reason, the usage of this type of systems provides for additional time-consuming steps. First of all, the internet latency required to access the remote machine, then the problem management time by the classic resources of the solver, and finally the wait for execution on QPU due to the presence of instructions of other users which must be run on the same quantum processor. More details on QPU access times can be found in the *D-Wave Systems* documentation [12].

Therefore, all these conditions lead to an overall time more significant than that employed by the solver, which uses the CPU locally. The real advantage of the system that uses quantum technology is the speed with which its quantum elaboration element works. The column of Table III called "QPU access time" [8] shows the actual QPU usage time. The QPU executes instructions in microseconds instead of seconds. Still, such computational advantage cannot be fully exploited yet since it is obscured by the additional time needed to resolve the problem on the remote system.

VI. CONCLUSIONS

In this paper, we have analyzed how combinatorial optimization problems can be effectively solved through quantum technology tools. Specifically, we aimed to investigate this innovative computation technique, quantum computing, and analyze the advantages and disadvantages that derive from it.

For this reason, a brief overview of quantum computing's branch of interest was exposed. First, quantum annealers, i.e., quantum systems suitable for solving optimization problems,

TABLE III
COMPARISON OF COMPUTATIONAL TIMES IN SECONDS OF THE D-WAVE SYSTEMS SOLVERS

Seats	**Passengers**	**Distinct booking IDs**	**QBSolv (D-Wave CPU solver)**	**LeapHybridSampler (D-Wave quantum-classical hybrid solver)**	**QPU access time (D-Wave quantum-classical hybrid solver)**
80	11	3	1.267	6.505	0.040579
80	16	4	1.567	7.792	0.041579
80	19	5	1.801	9.865	0.041494
80	23	7	4.561	15.776	0.042629
80	28	8	5.327	18.214	0.042623
80	34	9	5.976	18.525	0.042694
80	39	11	8.680	19.593	0.042617
80	44	13	10.560	20.819	0.042623
80	50	14	17.527	20.968	0.042624
80	51	15	18.848	22.041	0.042695

were presented. We then described quantum annealing, i.e., the process through which problems are solved on quantum annealers, its key elements, qubits, and entanglement's quantum property. We, therefore, presented the Quadratic Unconstrained Binary Optimization (QUBO) formulation. Moreover, we introduced two solvers offered by *D-Wave Systems* for solving this kind of problems on Central Processing Unit (CPU) and on quantum-classical hybrid remote systems which make use of a Quantum Processing Unit (QPU).

We then considered a specific case-study concerning the allocation of passengers to seats on high-speed trains with the recent hygiene and health regulations on social distancing due to the COVID-19 pandemic. We modeled the problem through the QUBO paradigm. Then, we compared the results obtained through the use of the *D-Wave Systems*' tools, QBSolv and Leap's cloud-based quantum-classical hybrid solvers on several instances of the analyzed problem.

In our future work, we intend to use the same case-study, the Seating Arrangement Optimization problem, to compare this novel computational approach to a more classical one. In particular, we aim to model the problem through a Mixed-Integer Linear Programming (MILP) formulation and to solve it through the use of a state-of-the-art solver, such as Gurobi or CPLEX. A noteworthy aspect that could also be explored in the future could be to set the λ coefficients as vector parameters, in order to be able to weigh each constraint in a different way. A last interesting development could be the study of the QUBO model with the introduction of an additional α parameter to determine the weight that the H_E term has within the objective function (9). With the introduction of this factor and with the addition of a multiplicative coefficient $(1-\alpha)$ in front of the remaining part of (9), we could study the relationships and the weights that the constraints and the objective function have within the QUBO model.

ACKNOWLEDGMENTS

This study was derived from a proposal and with the contribution of the Quantum Computing Team of Data Reply S.r.l., Torino (Italy).

REFERENCES

[1] Alidaee B., Kochenberger G., Lewis K., Lewis M., Wang H., *"A new approach for modeling and solving set packing problems"*, European Journal of Operational Research (2008), Volume 186, Issue 2, Pages 504-512, https://doi.org/10.1016/j.ejor.2006.12.068.

[2] Anthony, M., Boros, E., Crama, Y. et al. *"Quadratic reformulations of nonlinear binary optimization problems."* Math. Program. 162, 115–144 (2017). https://doi.org/10.1007/s10107-016-1032-4

[3] Asproni, L., Caputo, D., Silva, B. et al. Accuracy and minor embedding in subqubo decomposition with fully connected large problems: a case study about the number partitioning problem. Quantum Mach. Intell. 2, 4 (2020). https://doi.org/10.1007/s42484-020-00014-w

[4] Booth, M. & Reinhardt, S.P. *"Partitioning Optimization Problems for Hybrid Classical / Quantum Execution"*, Technical Report (2017)

[5] Borowski M. et al. (2020) *"New Hybrid Quantum Annealing Algorithms for Solving Vehicle Routing Problem."* In: Krzhizhanovskaya V. et al. (eds) Computational Science – ICCS 2020. ICCS 2020. Lecture Notes in Computer Science, vol 12142. Springer, Cham. https://doi.org/10.1007/978-3-030-50433-5_42

[6] Lewis, M. & Glover, F. *"Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis"*, Networks 70 (2017): 79-97, arXiv:1705.09844 [cs.AI]

[7] D-Wave Hybrid Solvers, D-Wave Documentation, https://docs.ocean.dwavesys.com/en/stable/overview/hybrid.html#dwave-hybrid-solvers

[8] D-Wave Leap's Hybrid Solvers' Timing Information, D-Wave Documentation, https://docs.dwavesys.com/docs/latest/timing_hybrid.html#timing-hybrid

[9] D-Wave Ocean SDK Documentation, https://docs.ocean.dwavesys.com/en/stable/getting_started.html

[10] D-Wave QBSolv function, docs: https://docs.ocean.dwavesys.com/projects/qbsolv/en/latest/index.html, source code: https://github.com/dwavesystems/qbsolv

[11] D-Wave Quantum Solvers, D-Wave Documentation, https://docs.ocean.dwavesys.com/en/stable/overview/qpu.html

[12] D-Wave Solver Computation Time, D-Wave Documentation, https://docs.dwavesys.com/docs/latest/doc_timing.html

[13] Glover, F., Kochenberger, G. & Du, Y. *"Quantum Bridge Analytics I: a tutorial on formulating and using QUBO models."* 4OR-Q J Oper Res 17, 335–371 (2019). https://doi.org/10.1007/s10288-019-00424-y

[14] Kochenberger, G., Hao, JK., Glover, F. et al. *"The unconstrained binary quadratic programming problem: A survey"*, Journal of Combinatorial Optimization (2014), 28, doi:10.1007/s10878-014-9734-0

[15] Lucas A. *"Ising formulations of many NP problems"*, Frontiers in Physics (2014), vol.2, p.5, doi: 10.3389/fphy.2014.00005

[16] Marchenkova A., *"What's the difference between quantum annealing and universal gate quantum computers?"*, https://medium.com/quantum-bits/what-s-the-difference-between-quantum-annealing-and-universal-gate-quantum-computers-c5e5099175a1

[17] Nam, Y., Chen, J., Pisenti, N.C., Wright, K., Delaney, C., Maslov, D., Brown, K., Allen, S., Amini, J., Apisdorf, J., Beck, K., Blinov, A., Chaplin, V., Chmielewski, M., Collins, C., Debnath, S., Ducore, A.M., Hudek, K., Keesan, M., Kreikemeier, S., Mizrahi, J., Solomon, P., Williams, M., Wong-Campos, J.D., Monroe, C., & Kim, J. *"Ground-state energy estimation of the water molecule on a trapped-ion quantum computer"*, npj Quantum Information (2019), vol.6, pag.1-6

[18] Neukart F., Compostella G., Seidel C., von Dollen D., Yarkoni S., Parney B. *"Traffic Flow Optimization Using a Quantum Annealer"*, Frontiers in ICT (2017), vol.4, doi:10.3389/fict.2017.00029

[19] Papalitsas, C.; Andronikos, T.; Giannakis, K.; Theocharopoulou, G.; Fanarioti, S. *"A QUBO Model for the Traveling Salesman Problem with Time Windows."* Algorithms 2019, 12, 224. https://doi.org/10.3390/a12110224

[20] Schuld, M., Sinayskiy, I., & Petruccione, F. *"An introduction to quantum machine learning"*, Contemporary Physics (2014), vol.56, pag.172 - 185

[21] Three Truths and the Advent of Hybrid Quantum Computing, https://medium.com/d-wave/three-truths-and-the-advent-of-hybrid-quantum-computing-1941ba46ff8c

[22] Wang Y., Lü Z., Glover F., Hao J., *"Path relinking for unconstrained binary quadratic programming"*, European Journal of Operational Research (2012), Volume 223, Issue 3, Pages 595-604, https://doi.org/10.1016/j.ejor.2012.07.012.