Optimization and Engineering
https://doi.org/10.1007/s11081-021-09662-8

RESEARCH ARTICLE

Optimization of a refinery scheduling process with column generation and a quantum annealer

J. Ossorio-Castillo[1] · **F. Pena-Brage**[1,2]

Received: 22 July 2020 / Revised: 21 April 2021 / Accepted: 3 July 2021

Abstract
This study focuses on the optimization of a refinery scheduling process with the help of an adiabatic quantum computer, and more concretely one of the quantum annealers developed by D-Wave Systems. We present an algorithm for finding a global optimal solution of a MILP that leans on a solver for QUBO problems, and apply it to various possible cases of refinery scheduling optimization. We analyze the inconveniences found during the whole process, whether due to the heuristic nature of D-Wave or the implications of reducing a MILP to QUBO, and present some experimental results.

Keywords Refinery scheduling · Quantum annealing · Mixed-integer programming · Column generation

1 Introduction

Since the first quantum algorithms, the range of problems where quantum computers can be applied have grown over time. Efforts have progressed in two fronts: to design algorithms that solve practical problems and to have operational quantum machines. In the first front, previous works on this matter that aimed to solve real-life problems with the help of a quantum annealer include Bauckhage et al. (2020), Calude and Dinneen (2017) and Venturelli et al. (2015). These articles consider a wide range of problems: the broadcast time problem, the job-shop scheduling problem and the

Part of this research was developed as an activity in the Joint Research Unit Repsol-ITMATI (Code File: IN853A 2014/03) which is funded by FEDER, the Galician Agency for Innovation (GAIN) and the Ministry of Economy and Competitiveness in the framework of the Spanish Strategy for Innovation in Galicia.

✉ F. Pena-Brage
fran.pena@usc.es

[1] Instituto Tecnolóxico de Matemática Industrial (ITMATI), Santiago de Compostela, Spain

[2] Univ. de Santiago de Compostela, Santiago de Compostela, Spain

Published online: 13 July 2021

max-sum diversification, respectively. However, all of them solely have binary variables in their formulations.

In the second front, the possibility of exploiting the advantages of quantum computation over classical computers has begun to take form. The standard gate-based quantum computing is more natural for computer scientists and is the one used in most textbooks (Nannicini 2020). Adiabatic quantum computers are equivalent and they are well suited for optimization problems (Aharonov et al. 2008). Various companies, and more especially D-Wave Systems (D-Wave 2016), have been researching a way to physically implement these new computing paradigms prophesied by Feynman (1982) and Born and Fock (1928). Although the debate of whether the D-Wave machine really exploits quantum phenomena continues generating controversy, the fact that these supercomputers are especially predisposed to solve quadratic optimization problems with binary variables is indisputable (Calude and Dinneen 2017; Syrichas and Crispin 2017; Venturelli et al. 2015).

Our aim is to take a problem of industrial interest and to follow the whole process of adapt it in order to solve it with a quantum annealer. We have choosen the scheduling of the arrival of vessels to the harbor in an oil refinery, the unloading of its contents into tanks, and also to get the most of them during the whole refining process. This classical problem in the oil and gas industry presents the advantage that its simpler version as mixed-integer linear problem (MILP) still retains the practical meaning (Lee et al. 1996). Surveys of representative works on optimization in the oil and gas industry can be found in Khor and Varvarezos (2017) and extensively in Furman et al. (2017). Mathematically, this kind of problems usually take a huge amount of constraints and variables (Karuppiah et al. 2008; Lee et al. 1996; Mouret 2010), a feature that clearly complicates the finding of the global solution within a reasonable amount of time.

The problem of finding the global optimum of a MILP is an NP-Hard problem, and appears in multiple scenarios and applications. Our contribution and the main objective behind this work is to solve a specific MILP, the aforementioned scheduling problem, defined by binary and continuous variables with the help of a quantum annealer developed by D-Wave (see D-Wave (2016) for more details). To take advantage of the capacity of such computer, a decomposition technique is presented in this work. The approach we have developed consists of a combination of column-generation algorithms such as the Dantzig–Wolfe decomposition (Dantzig and Wolfe 1960) and a branch-and-price method that correctly obtains an integer solution for the binary variables (Gamrath 2010). It is precisely the part that calculates the new columns the one that requires the solution of a binary linear problem, known as BLP, ZOLP or 0-1LP, where we can take advantage of the capacity of such quantum annealer. Although the refinery problem formulations, the column generation and the decomposition techniques present in this work are not novel, our contribution resides in using those techniques for separating the real part from the binary part of those problems and solving the latter one with a quantum annealer. Other hybrid methods that exploit the complementary strengths of quantum and classical computers can be consulted in Ajagekar et al. (2020) and Ajagekar (2020).

The embedding of the binary problem into D-Wave requires transforming it into a Quadratic Unconstrained problem. However, it cannot be directly solved even in

that case, since the available number of qubits is quite limited and the topology of the Chimera graph is rather specific. In this work we study the use the Qbsolv library (Booth et al. 2017; D-Wave Systems Inc. 2017) to overcome this difficulty and its extra cost in iterations. Due to the requirement of having a global solution in several steps of the algorithm, we have also checked that the solution has this property. Finally, we have tested the program with reference problems within the field of oil and gas.

This paper is organized as follows: in Sect. 2, we present the scheduling problem from an applied point of view. In Sect. 3, we describe the column generation technique that best suits our goal, and the branch-and-price method that completes the algorithm to decompose our problem in the mixed and binary parts. In Sect. 4, we describe more specifically how to solve the BLP part of the algorithm with a quantum annealer. Finally, in Sect. 5 we present some experimental results: we solve a small refinery problem with an actual quantum annealer, the D-Wave 2X processor based in the University of Southern California.

2 Overview of a refinery scheduling process

First, we introduce the main mathematical problem behind the optimization of the refinery scheduling process, which can generally be modeled as a MILP. As the main objective of this work consists in optimizing the scheduling process of a refinery with the help of a quantum annealer, we have to separate the real-valued variables of the problem from the integer ones. For this purpose we shall use the Dantzig–Wolfe decomposition, which we explain in the next section.

The refinery model has to take into account the variables and parameters involved in the refinery scheduling operations, such as the unloading of the vessels and the charging and storing into the tanks. It also has to define the sets that include all physical units: vessels, tanks, resources, etc. The model we have used as a basis for our algorithm can be found in Lee et al. (1996). This is a classical problem of inventory management of a refinery that imports several types of crude oil which are delivered by different vessels. The problem involves bilinear equations due to mixing operations. However, the linearity in the form of a MILP is maintained by replacing bilinear terms with individual component flows. This exact linear reformulation is possible since this scheduling system involves only mixing operation without splitting operation. More details about this problem and other possible reformulations can be seen in Mouret (2010) and Vyskocil and Djidjev (2019).

The scheduling problem consists of a multistage system composed of N_V vessels, N_S storage tanks, N_C charging tanks and N_D distillation units, with N_C key components of crude oil, as illustrated in Fig. 1.

For $1 \leq \nu \leq N_V$, the νth ship arrives at time $T_{A,\nu}$ with a volume of crude at initial time of $V_{V,\nu,0}$. The cost of unloading a vessel per time unit is C_U, and the cost of waiting in the sea per time unit is C_S.

For $1 \leq i \leq N_S$, the ith storage tank has a volume of crude at initial time of $V_{S,i,0}$ and its load can vary from a minimum of $V_{S,i,min}$ to a maximum of $V_{S,i,max}$, whereas the crude transferred from the νth ship to the ith storage tank can vary

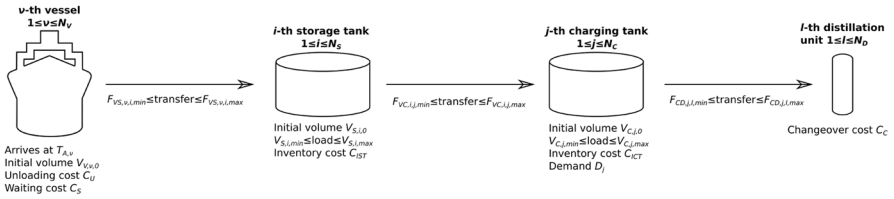

Fig. 1 Constant parameters considered in the scheduling problem

from $F_{VS,v,i,min}$ to $F_{VS,v,i,max}$. The inventory cost for the storage tanks per unit of time and volume is C_{IST}.

For $1 \leq j \leq N_C$, the mixed crude in the jth charging tank has an initial volume of $V_{C,j,0}$ and it can vary from $V_{C,j,min}$ to $V_{C,j,max}$, having a demand of D_j. The crude transferred from the ith storage tank can vary from $F_{SC,i,j,min}$ to $F_{SC,i,j,max}$. The inventory cost for the charging tanks per unit of time and volume is C_{ICT}.

Finally, for $1 \leq l \leq N_D$, the crude transferred from the jth charging tank to the lth distillation unit can vary from $F_{CD,j,l,min}$ to $F_{CD,j,l,max}$. In a distillation unit the changeover of crude from a charging tank to another has a cost $C_C \geq 0$.

Our problem considers a scheduling horizon discretized in S time units. Following the previous index notations and for each time unit t, $1 \leq t \leq S$, some continuous variables must be determined (see Fig. 2). For the vth ship, we need to know the time point when the unloading starts $t_{U,v}$ and ends $t_{D,v}$. We also have to find out at time t,

- the volume $v_{V,v,t}$ that the vessel has,
- whether or not the vessel is unloading crude, $0 \leq x_{W,v,t} \leq 1$,
- the volumetric flow rate of crude $f_{VS,v,i,t}$ from the vth vessel to the ith storage tank,
- the volume of crude $v_{S,i,t}$ and the concentration of the kth component $p_{S,i,k}$ in it,
- the volumetric flow rate of crude $f_{VC,i,j,t}$ and the kth component $f_{SC,i,j,k,t}$ from this tank to the jth charging tank,
- the volume of crude $v_{C,j,t}$ and the volume of the kth component $w_{C,j,k,t}$ in it,
- the volumetric flow rate of crude $f_{VD,j,l,t}$ and
- the kth component $f_{CD,j,l,k,t}$ from this tank to the lth distillation unit, and
- whether or not there is a transition from the jth charging tank to the j'th one, $0 \leq z_{j,j',l,t} \leq 1$.

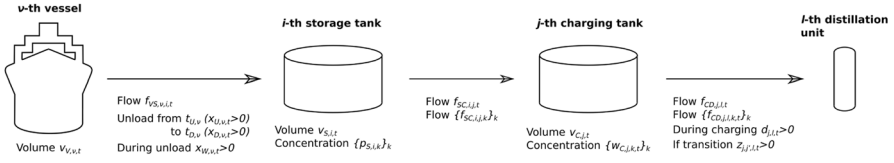

Fig. 2 Variables considered in the scheduling problem

Besides, we must find out several binary variables:

- $x_{U,v,t}$ and $x_{D,v,t}$, with value 1 when the vth vessel starts and completes unloading at time t, respectively.
- $d_{j,t,l}$, with value 1 when the jth charging tank is charging into the lth distillation unit at time t.

Finally, our scheduling problem is written in terms of the following cost minimization problem.

Problem (SP): Find the value of the previous set of variables that optimizes with respect to $t_{U,v}$, $t_{D,v}$, $v_{S,i,t}$, $v_{C,j,t}$ and $z_{j,j',l,t}$ the following MILP:

$$\begin{aligned}\min\, & C_U \sum_{v=1}^{N_V}(t_{D,v} - t_{U,v}) + C_S \sum_{v=1}^{N_V}(t_{U,v} - T_{A,v}) + C_{IST} \sum_{i=1}^{N_S}\sum_{t=1}^{S}\left(\frac{v_{S,i,t} - v_{S,i,t-1}}{2}\right) \\ & + C_{ICT} \sum_{j=1}^{N_C}\sum_{t=1}^{S}\left(\frac{v_{C,j,t} - v_{C,j,t-1}}{2}\right) + \sum_{t=1}^{S}\sum_{j=1}^{N_C}\sum_{j'=1}^{N_C}\sum_{l=1}^{N_D} C_C z_{j,j',l,t},\end{aligned} \tag{1}$$

and, for $v = 1, ..., N_V$, $i = 1, ..., N_S$, $j = 1, ..., N_C$, $l = 1, ..., N_D$ and $t = 1, ..., S$, subject to the following constraints over binary variables:

$$\sum_{t=1}^{S} x_{D,v,t} = 1, \tag{2a}$$

$$\sum_{l=1}^{N_D} d_{j,l,t} \leq 1, \tag{2b}$$

$$\sum_{j=1}^{N_C} d_{j,l,t} = 1, \tag{2c}$$

subject to the constraints over continuous variables:

$$t_{D,v} \geq T_{A,v}, \tag{3a}$$

$$t_{D,v} - t_{U,v} \geq \frac{V_{V,v,0}}{\max_i F_{VS,v,i,max}}, \tag{3b}$$

$$t_{U,v+1} \geq T_{D,v}, \tag{3c}$$

$$v_{V,v,t} = V_{V,v,0} - \sum_{i=1}^{N_S}\sum_{m=1}^{t} f_{VS,v,i,t}, \tag{3d}$$

$$v_{V,v,t} = V_{V,v,0} - \sum_{i=1}^{N_S} \sum_{m=1}^{t} f_{VS,v,i,t}, \tag{3e}$$

$$F_{VS,v,i,min}\, x_{W,v,t} \leq f_{VS,v,i,t} \leq F_{VS,v,i,max}\, x_{W,v,t}, \tag{3f}$$

$$\sum_{i=1}^{N_S} \sum_{t=1}^{S} f_{VS,v,i,t} = V_{V,v,0} \tag{3g}$$

$$v_{S,i,t} = V_{IS,i} + \sum_{v=1}^{N_V} \sum_{m=1}^{t} f_{VS,v,i,m} - \sum_{j=1}^{N_C} \sum_{m=1}^{t} f_{SC,i,j,m}, \tag{3h}$$

$$V_{S,i,min} \leq v_{S,i,t} \leq V_{S,i,max}, \tag{3i}$$

$$v_{C,j,t} = V_{C,j,0} + \sum_{i=1}^{N_S} \sum_{m=1}^{t} f_{SC,i,j,m} = \sum_{l=1}^{N_D} \sum_{m=1}^{t} f_{CD,j,l,m}, \tag{3j}$$

$$V_{C,j,min} \leq v_{C,j,t} \leq V_{C,j,max}, \tag{3k}$$

$$\sum_{l=1}^{N_D} \sum_{t=1}^{S} f_{CD,j,l,t} = D_j, \tag{3l}$$

$$v_{C,j,t} = V_{C,j,0} + \sum_{m=1}^{t} (\sum_{i=1}^{N_S} f_{SC,i,j,m} - \sum_{t=1}^{N_D} f_{CD,j,l,m}), \tag{3m}$$

$$f_{SC,i,j,k,t} = f_{SC,i,j,t} \cdot p_{S,i,k}, \tag{3n}$$

$$f_{CD,j,l,t}\, p_{S,j,k,min} \leq f_{CD,j,l,k,t} \leq f_{CD,j,l,t}\, p_{S,j,k,max}, \tag{3o}$$

$$v_{C,j,t}\, p_{S,j,k,min} \leq w_{C,j,k,t} \leq v_{C,j,t}\, p_{S,j,k,max}, \tag{3p}$$

$$0 \leq z_{j,j',l,t} \leq 1, \tag{3q}$$

$$z_{j,j',l,t} \geq d_{j',l,t} + d_{j,l,t-1} - 1, \tag{3r}$$

and subject to the constraints over mixed variables:

$$\sum_{t=1}^{S} t\, x_{D,v,t} = t_{D,v}, \tag{4a}$$

$$\sum_{t=1}^{S} t\, x_{U,v,t} = t_{U,v}, \tag{4b}$$

$$x_{W,v,t} \leq \sum_{m=1}^{t} x_{U,v,m}, \tag{4c}$$

$$x_{W,v,t} \leq \sum_{m=t}^{S} x_{D,v,t}, \tag{4d}$$

$$F_{SC,i,j,min}\left(1 - \sum_{l=1}^{N_D} d_{j,l,t}\right) \leq f_{SC,i,j,t} \leq F_{SC,i,j,max}\left(1 - \sum_{l=1}^{N_D} d_{j,l,t}\right), \tag{4e}$$

$$F_{CD,j,l,min} d_{j,l,t} \leq f_{CD,j,l,t} \leq F_{CD,j,l,max} d_{j,l,t}. \tag{4f}$$

3 Column generation

In this section, we show how to decompose the previous problem—or more generally, any MILP problem of its type—into its real and its binary part. We apply the Dantzig–Wolfe decomposition (for a more detailed explanation of this decomposition, see Chvatal 1983 or Dantzig and Wolfe 1960), and then apply to it a column generation algorithm. The algorithm evolves around a master problem, namely (MP), which is updated in every iteration. Both iterations and the stopping criteria depend on two subproblems, (SP_1) and (SP_2), which are also updated during every iteration.

Let us write first a generic form for the MILP we want to solve that includes the problem (SP). From now on, A matrices and $\mathbf{a}$ vectors will describe the equality constraints, while B matrices and $\mathbf{b}$ vectors will do the same with inequality constraints. Likewise, subindices b and r describe the coefficients associated with the binary and continuous variables respectively and $\mathbf{c}$ vectors define the objective function. The $\mathbf{x}$ and $\mathbf{z}$ vectors designate the binary and real variables, respectively. Expressed as a linear optimization problem, we have the following objective function

$$\min_{\mathbf{x},\mathbf{z}} \mathbf{c}_r \cdot \mathbf{x} + \mathbf{c}_b \cdot \mathbf{z}, \tag{5}$$

subject to the following constraints:

$$A_b\mathbf{z} = \mathbf{a}_b, \tag{6a}$$

$$B_b\mathbf{z} \leq \mathbf{b}_b, \tag{6b}$$

$$A_r\mathbf{x} = \mathbf{a}_r, \tag{6c}$$

$$B_r\mathbf{x} \leq \mathbf{b}_r, \tag{6d}$$

$$A_{mr}\mathbf{x} + A_{mb}\mathbf{z} = \mathbf{a}_m, \tag{6e}$$

$$B_{mr}\mathbf{x} + B_{mb}\mathbf{z} \leq \mathbf{b}_m, \tag{6f}$$

for $\mathbf{x} \in \mathbb{R}^{m_1}$, $\mathbf{z} \in \mathbb{Z}^{m_2}_{\geq 0}$, $\mathbf{a}_r, \mathbf{b}_r \in \mathbb{R}^{m_r}$, $\mathbf{a}_b, \mathbf{b}_b \in \mathbb{R}^{m_b}$, $\mathbf{a}_r, \mathbf{b}_r \in \mathbb{R}^{m_m}$, $A_r, B_r \in \mathbb{R}^{n_r \times m_r}$, $A_b, B_b \in \mathbb{R}^{n_b \times m_b}$, $A_m, B_m \in \mathbb{R}^{n \times m_m}$, with $n = n_r + n_b + n_m$ being the number of original constraints in the problem and $m = m_r + m_b$ the number of original variables.

We call this feasible point the first proposal, denoted by the vector $\mathbf{x}_{(1)}$ for the continuous variables and the vector $\mathbf{z}_{(1)}$ for the binary ones. From now on, we will denote by k and k' the number of continuous and binary proposals stored in every iteration. These proposals will be weighted respectively in the problem (MP) with the additional continuous variables λ_i and μ_j. The master problem can be defined in its generic form as follows:

Problem (MP):

$$\begin{aligned}
\min_{\lambda_i,\mu_j} \quad & \sum_{i=1}^{k}(\mathbf{c}_r \cdot \overline{\mathbf{x}}_{(i)})\lambda_i + \sum_{j=1}^{k'}(\mathbf{c}_b \cdot \overline{\mathbf{z}}_{(j)})\mu_j, \\
\text{subject to} \quad & \\
& \sum_{i=1}^{k}(A_{mr}\overline{\mathbf{x}}_{(i)})\lambda_i + \sum_{j=1}^{k'}(A_{mb}\overline{\mathbf{z}}_{(j)})\mu_j = \mathbf{a}_m, \\
& \sum_{i=1}^{k}(B_{mr}\overline{\mathbf{x}}_{(i)})\lambda_i + \sum_{j=1}^{k'}(B_{mb}\overline{\mathbf{z}}_{(j)})\mu_j \leq \mathbf{b}_m, \\
& \sum_{i=1}^{k}\lambda_i = 1, \sum_{j=1}^{k'}\mu_j = 1, \\
& \lambda_i, \mu_j \geq 0.
\end{aligned}$$

The previous linear and continuous problem can be solved efficiently with the help of a specialized global linear solver. The only information we need each time we solve problem (MP) is the dual solution of its constraints, which will be represented

from now on as the vector $\boldsymbol{\pi}$ for the equality constraints and the vector $\boldsymbol{\rho}$ for the inequality constraints. The value of these vectors will be updated each time we solve (MP).

It is in the next step of the algorithm where we can optionally use an adiabatic quantum computer, but it is also possible to accomplish it with the help of a global nonlinear solver. Two new subproblems are defined for this step:

Problem ($\mathbf{SP}_1$):

$$\begin{aligned} &\min_{\mathbf{x}} && (A_{mr}\boldsymbol{\pi} + B_{mr}\boldsymbol{\rho}) \cdot \mathbf{x}, \\ &\text{subject to} && \\ & && A_r\mathbf{x} = \mathbf{a}_r, \\ & && B_r\mathbf{x} \leq \mathbf{b}_r. \end{aligned}$$

Problem ($\mathbf{SP}_2$):

$$\begin{aligned} &\min_{\mathbf{z}} && (A_{mb}\boldsymbol{\pi} + B_{mb}\boldsymbol{\rho}) \cdot \mathbf{z}, \\ &\text{subject to} && \\ & && A_b\mathbf{z} = \mathbf{a}_b, \\ & && B_b\mathbf{z} \leq \mathbf{b}_b. \end{aligned}$$

As can be seen, both subproblems (SP_1) and (SP_2) depend on the dual solutions obtained from problem (MP) in order to define their objective functions. Problem (SP_1) is linear and continuous and can be solved again as problem (MP) with a LP solver. Problem (SP_2), however, have binary variables and therefore have a stronger complexity. Its resolution with the help of an quantum annealer will be explained in Sect. 4. For the time being, let us just assume that we have a black box that solves it globally.

Remark An Adiabatic Quantum Computing (AQC) algorithm (Farhi et al. 2001) is guaranteed to converge to the global optimum of problems but the tempering time might grow exponentially and the temperature has to be zero, while a Quantum Annealing (McGeoch and Wang 2013) process is a physical implementation of AQC (can be considered as a subcase) with finite temperature implementation and no deterministic convergence guarantees. Since our algorithm requires global convergence, an AQC would be suitable for it. However our results were obtained with a machine that implements a quantum annealing process, so we had to check that in this particular case the global solution was reached, as it is explained in Sect. 5.

The algorithm starts solving an instance of the original (SP_1) problem after dropping the objective function. This way, we will only need to get a feasible point for the problem, instead of one of its minima, a much easier achievement than its optimization counterpart. In order to complete this task we use the local nonlinear optimization solver Knitro (Byrd et al. 2006). See Fig. 3 for a general overview of the algorithm.

After both subproblems are solved, it is time to check the first terminating condition of the algorithm. We have to examine the value of both objective functions from (SP_1) and (SP_2). If one or both of them are less than 0, the algorithm

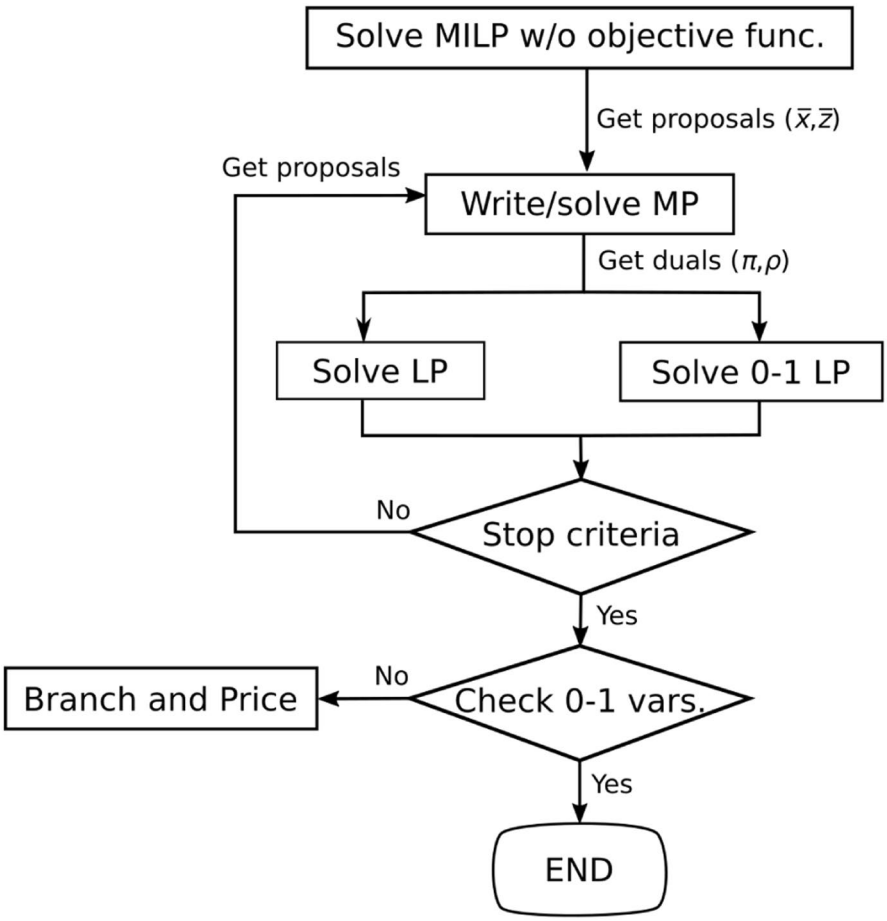

Fig. 3 Flowchart of the algorithm

comes back to the step where problem (MP) is solved and adds the solution for both subproblems (or only for the one whose objective function is less than 0) as proposals. In any other case, we rebuild the actual solution for the problem as a linear combination of the proposals and the last solution for the weights λ_i and μ_j, as shown in the next equations:

$$\mathbf{x}^* = \sum_{i=1}^{k} \lambda_i \bar{\mathbf{x}}_{(i)}, \tag{7}$$

$$\mathbf{z}^* = \sum_{j=1}^{k'} \mu_j \bar{\mathbf{z}}_{(j)}. \tag{8}$$

As a final termination for the algorithm, it is time to check if the solutions calculated for the binary variables in the previous step are in fact 0s and 1s. If the answer is negative for any of the variables, a branch-and-price scheme has to be run over the proposals obtained with the column generation algorithm. You can see an overall flowchart of the method described in this section in Fig. 3. More information about the branch-and-price algorithm can be found at Feillet (2010) and Gamrath (2010).

4 Resolution of an ILP as a QUBO

In order to solve an optimization problem with binary variables like (SP_2) using a quantum annealer, it is necessary first to transform it into an Ising Spin (IS) problem or a Quadratic Unconstrained Binary Optimization (QUBO) problem. Both types of problems are unconstrained, but a BLP usually has constraints in its formulation. A possible way of penalizing the constraints inside the objective function of the IS or QUBO problem is described in D-Wave (2016), with the objective of solving a generic BLP with a D-Wave machine.

We proceed to explain this transformation and its associated iterative method. First, let us suppose that we have a BLP in its generic form:

$$\begin{array}{ll} \min_{\mathbf{z}} & \mathbf{c}\cdot\mathbf{z}, \\ \text{subject to} & A\mathbf{z}=\mathbf{a}, \\ & B\mathbf{z}\le\mathbf{b}, \\ & z_i\in\{0,1\}. \end{array}$$

A possible way of penalizing the constraints $A\mathbf{z}=\mathbf{a}$ and $B\mathbf{z}\le\mathbf{b}$ in the objective function is to express them as quadratic penalties. Due to the binary nature of the variables, from now on we will only explain the transformation into QUBO form, although the IS counterpart is similar. Here, index i runs over the variables, whereas j runs over the equality constraints and k runs over the inequality constraints:

$$\min_{\mathbf{z}} \mathbf{c}\cdot\mathbf{z}+\sum_j p_j(\mathbf{A}_j\cdot\mathbf{z}-a_j)^2+\sum_k q_k(\mathbf{B}_k\cdot\mathbf{z}-b_k+\mathbf{2}\cdot\boldsymbol{\xi}_k)^2, \tag{9}$$

where $\mathbf{2}$ is the vector of powers of 2, i.e., $[2^l,\ldots,2,1]$ and $\boldsymbol{\xi}_k$ is the vector of bits of the slack variable ξ_k; that is, $\xi_k=\mathbf{2}\cdot\boldsymbol{\xi}_k$. Here l is the number of bits needed to code the slack variable. We have introduced new parameters, namely p_j and q_k associated with the weighting of the penalties. However, the value of these parameters cannot be calculated a priori. It is necessary to iterate over them until all constraints have been satisfied. As stated in D-Wave (2016), a possible way of achieving this would be as follows:

$$p_j^{(n+1)}=p_j^{(n)}+\alpha\,|\mathbf{A}_j\cdot\mathbf{z}^{(n)}-a_j|, \tag{10}$$

$$q_k^{(n+1)}=q_k^{(n)}+\alpha\,|\mathbf{B}_k\cdot\mathbf{z}^{(n)}-b_k+\mathbf{2}\cdot\boldsymbol{\xi}_k^{(n)}|. \tag{11}$$

with $\alpha>0$ representing the increase rate of the penalty weighs.

Fig. 4 Resolution of an ILP as a QUBO ▶

Figure 4 shows the iterative process needed for solving an ILP as a QUBO. The dotted line contains the parts that have to be done by the quantum annealer, and represents a part of the algorithm that ideally should be a black box solved in just one iteration. However, as will be explained later, current technology does not allow for that when the size of the problem is large enough. The part corresponding to "Is global solution ensured with certain probability?" depends on the machine we are using: with an ideal adiabatic quantum computer, we should have a total assurance that the solution obtained is the global optimum; with a quantum annealer, however, we have to check with another solver (i.e., Baron) that the solution provided is the global optimum. We acknowledge that this is not ideal, but we expect that a quantum machine capable of obtaining a global optimum would be available in the future.

Note that decomposing the original refinery problem with the Dantzig–Wolfe algorithm rather than with a Benders' decomposition maintains the structure of the original constraints, thus keeping any possible advantage that the problem formulation could have at the time of embedding it into the Chimera graph. It remains to be seen if Benders' is a better choice for solving another different sort of MILP with a quantum annealer (Verstichel et al. 2015).

Regarding the embedding of the problem into D-Wave, at present the available number of qubits and couplings in current technology (1152 and 3360 respectively in the case of the D-Wave 2X processor), make non-viable the direct resolution of problems with a large number of variables and a huge density in its pairing ups derived from its constraints. The company have developed an open-source library called `Qbsolv` with the objective of palliating this weakness while hardware size limitations are reduced. With the help of this library, bigger problems can be partitioned and reduced into subproblems that can be solved by the D-Wave system you are using. The embedding problem is an NP-complete problem, but as the structure of our QUBO problem is maintained during the whole process, we can just solve this problem for the first iteration and reuse the solution. More information about `Qbsolv` can be found in Booth et al. (2017) and D-Wave Systems Inc. (2017).

5 Experimental results

The refinery scheduling problem (SP) can fit in the MILP generic problem just identifying its constraints over binary variables (2a)–(2c) with (6a)–(6b), the constraints over reals (3a)–(3r) with (6c)–(6d) and the constraints over mixed variables (4a)–(4f) with (6e)–(6f). Thus, we can use the afore-described algorithm with the help of a D-Wave machine to solve (SP).

We have run a program written in C++ that implements the algorithm and calls to D-Wave. A first check was conducted using a simple problem (P0) that corresponds to a minimal problem with three continuous variables and three binary variables. Then the program was tested with the MILP "multmip3" found in the AMPL

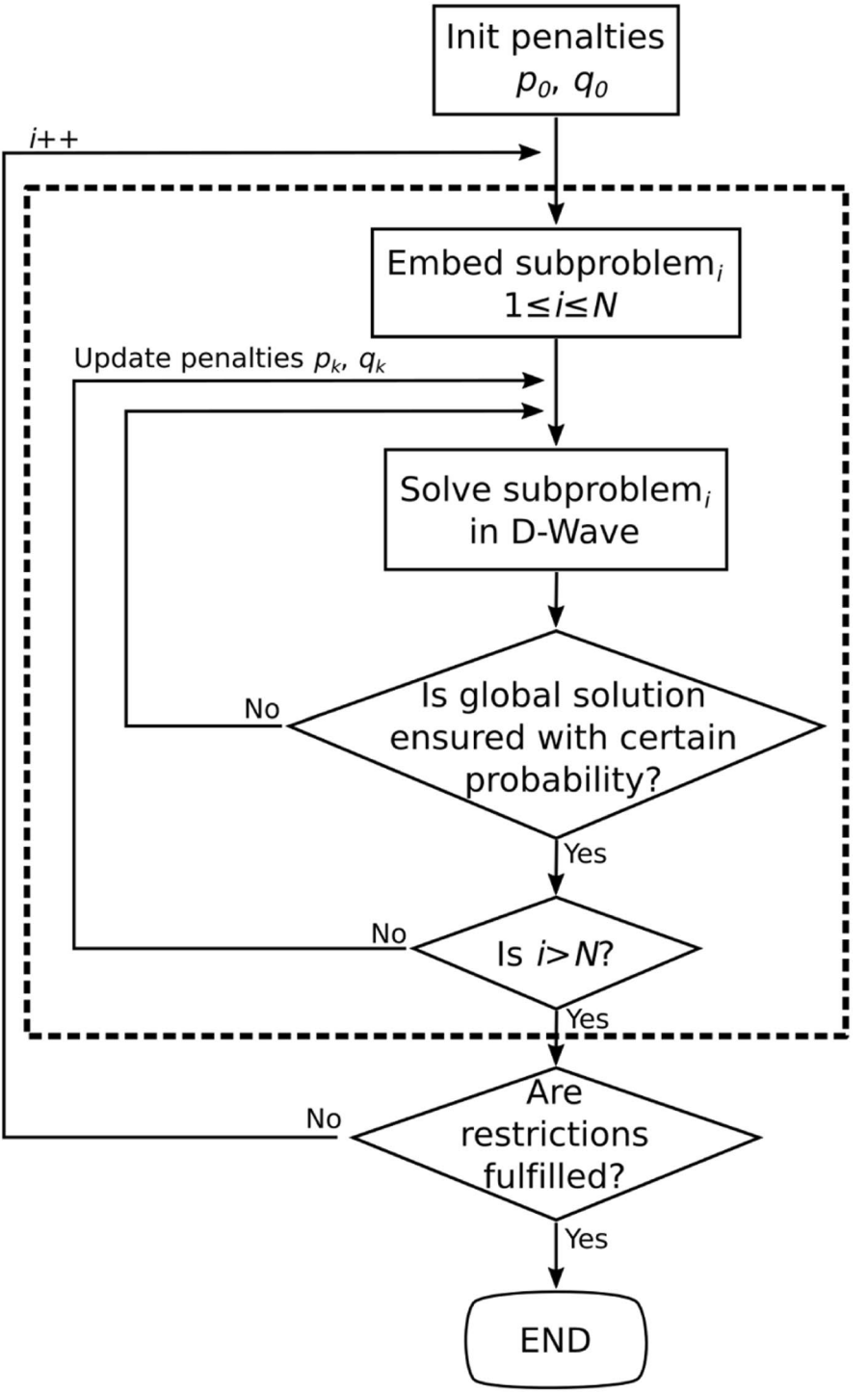
Init penalties
p_0, q_0
i++
Embed subproblem$_i$
$1 \leq i \leq N$
Update penalties p_k, q_k
Solve subproblem$_i$
in D-Wave
No
Is global solution ensured with certain probability?
Yes
No
Is $i > N$?
Yes
No
Are restrictions fulfilled?
Yes
END

Table 1 Table of iterations and calls for every problem

Properties	Problems			
	P0	PM	P2	P4
Variables				
Real	3	102	546	1776
Binary	3	21	120	270
Total	6	123	786	2046
Constraints				
Real	1	29	218	534
Binary	1	3	6	6
Mixed	2	41	155	455
Total	4	73	379	995
Generated columns				
Real	4	247	~ 1000	~ 1000
Binary	2	74	37	50
Weighting iterations				
Per column	1	13	2	2
Total	2	962	74	100
Qbsolv iterations				
Per weighting	1	1	62	42
Per column	1	13	124	84
Total	2	962	4588	4200

documentation (AMPL Optimization Inc. 2017) and named here as (PM). Finally, we have tested the program with problems obtained from article (Lee et al. 1996), which is a reference within the field of oil and gas. Its "Problem 2" and "Problem 4", named here (P2) and (P4), gather all the possible difficulties inherent to a refinery scheduling problem. Although they may not have the size of a realistic one, their size force the program to divide them in treatable subproblems.

Table 1 summarizes the properties of the four problems in terms of number and type of variables and constraints. Besides, it also collects the following figures:

- The number of iterations for every part of the algorithm applied to them, including the number of columns generated for every problem (i.e. the iterations of the algorithm described in Sect. 3).
- The number of weighting iterations (i.e. the number of times we have to increment the α value in the penalizing constraints)
- The number of iterations of `Qbsolv`. The total number of calls refers to a quantum annealer like D-Wave 2X, but this final number depends on the number of qubits in the chip, presently around 1000; it does contain neither the iterations made by the D-Wave machine in order to ensure the correct pairing of mirror variables nor the internal solution of the problems made in order to maximize the probability of finding the global optimum.

As a preliminary conclusion, our algorithm seems to be especially predisposed to solve problems with the particular structure similar to (P2) and (P4), thanks to the proportion between binary, mixed and continuous constraints, and also the grouping of binary variables inside the binary constraints. We have also observed that the performance of `Qbsolv` is not linear in relation to the number of binary variables, as can be seen in Table 1, probably due to the heuristical nature of its algorithm.

The purely binary subproblems that have to be solved during the execution of our method acts as its bottleneck, and the responsibility of solving them could lie either with an specific solver for this kind of problems, or with an adiabatic quantum computer. In our case, we have studied the possibilities of adapting the formulation of such subproblems to those that can be solved by the D-Wave quantum annealer. For that, it is necessary to transform our subproblems from its BLP format into QUBO, equivalent to IS, as it was explained in Sect. 4. Transforming to QUBO or IS could be advantageous in a certain kind of problems, and as can be seen in Table 1, the number of iterations needed for enforcing the constraints is minimal in the case of the refinery problems.

As explained previously, in certain cases the embedding of those QUBO problems into the D-Wave machine could result in the use of mirror variables that represent the same variable in the original QUBO formulation, and that have to be connected inside the Chimera graph. In order to ensure that those variables end up with the same solution, it is mandatory to strengthen their connection modifying certain terms of the QUBO problem; namely, if the qubits i and j represent the same original variable, we have to add a certain *chain strength* weight to the term Q_{ij}, but this could increase drastically the number of calls to the D-Wave machine until all mirror variables match.

Concerning size limitations, if our QUBO problem has a total connectivity within its variables (i.e. the problem graph is complete), the heuristic algorithm present in the D-Wave libraries can only guarantee the success of the embedding process with a probability of 100% up to 33 variables in the case of D-Wave 2X (a dramatic reduction if we have into account that the chip has more than one thousand qubits, but this is in the worst case scenenario when the graph of the problem is complete; with sparse problems, a problem with much more variables could be embedabble in the D-Wave 2X). To that effect, D-Wave has developed an open-source library called `Qbsolv`, intended for solving problems that cannot be embedded into the current D-Wave chip (whether it is for size limitations or for connectivity density). The use of `Qbsolv` implies several loops in order to ensure a global optimum within a certain probability. The final number of calls to the machine depends, among other things, of the success of the heuristic used for the partitioning and embedding process. We have also noted the existence of a relationship between the number of binary variables involved in the strictly binary constrains, and also with the number of binary variables present in a certain constraint. In the case of the refinery scheduling problems, this translates to a dependency over the complexity of the first stages of the whole process (i.e. the arrival of the vessels and downloading of their contents).

The main problem we have found while using `Qbsolv` libraries or solving certain subproblems with the D-Wave machine is that finding the global optimum is not guaranteed in every case, a necessary condition for the correct execution of our algorithm. In order to avoid problems generated by these limitations, we have checked the solution obtained by `Qbsolv` or D-Wave with the nonlinear solver BARON (Sahinidis 2014; Tawarmalani and Sahinidis 2005), which guarantees a global solution. We have checked that in our problems the result of the algorithm was correct and, regarding the time performance, the CPU time reported by the `Qbsolv` tool for decomposing the original problem averages, in our case, between 500 ms and 2 s, and the estimation of the D-Wave 2X computation time for each of the subproblems (including the connection times) averages 300 ms in the P4 problem. These times are coherent with the value of 491 ms of McGeoch and Wang (2013) for an original D-wave Two and the range of 10–300 ms for a D-Wave 2000Q, presented in Chiscop et al. (2020).

6 Conclusions and future work

In this paper we have described a possible technique for solving an optimization problem with continuous and binary variables, drawing upon a column generation scheme supported by the Dantzig–Wolfe decomposition. In this manner, thanks to breaking down the problem in its real and its binary parts, it is necessary to solve several subproblems, each of them of either a continuous or binary nature, instead of a mixed one. Although the algorithm can be used for any generic MILP, our interest resided in a certain type of optimization problems: the ones involved in the processes of a refinery, and more concretely in the ones associated with the correct scheduling of the arrival and discharge of vessels in the harbor.

In problems with a different structure, this algorithm could not be that beneficial and the average number of iterations in any part of the algorithm may explode, but the complexity in the worst case remains to be seen. A possible way of palliating this is to decompose the MILP in various binary subproblems, not just one, using Dantzig–Wolfe, maintaining a certain grouping of variables and thus improving the performance of the constraint enforcement process.

Other possible difficulties that may arise while solving our problems are related to the QUBO formulations. The Chimera graph present in the D-Wave chips have some serious limitations either related to the problem size or to the connectivity of its variables, proper of a technology that is currently in its early steps. In the meantime, in order to solve a QUBO problem with the D-Wave machine it is necessary to embed its graph into the Chimera graph, but this process could result in an increase in the number of calls to the quantum annealer, as explained in Sect. 4. Those limitations in current hardware may alter the perception of the possibilities of using this technique, but we hope that future advancements in quantum computers will make it more competitive.

Acknowledgements The authors would like to thank the Information Sciences Institute at the University of Southern California, for the possibility of using their D-Wave 2X machine, and especially Itay Hen for

all the help and support granted during the whole process. We would also like to express our gratitude to the former employees of Repsol that supported this work. Finally, we would like to thank the referees for their helpful comments.

References

Aharonov D, van Dam W, Kempe J, Landau Z, Lloyd S, Regev O (2008) Adiabatic quantum computation is equivalent to standard quantum computation. SIAM Rev 50(4):755–787

Ajagekar AS (2020) Quantum computing for process systems optimization and data analytics. Master's thesis, Cornell University, USA

Ajagekar A, Humble T, You F (2020) Quantum computing based hybrid solution strategies for large-scale discrete-continuous optimization Problems. Comput Chem Eng 132:106630

AMPL Optimization Inc. The AMPL book. Example files. http://ampl.com/resources/the-ampl-book/example-files/. Accessed 22 Nov 2017

Bauckhage C, Sifa R, Wrobel S (2020) Adiabatic quantum computing for max-sum diversification. In: Proceedings of the 2020 SIAM international conference on data mining, SIAM, pp 343–351

Booth M, Reinhardt SP, Roy A (2017) Partitioning optimization problems for hybrid classical/quantum execution. 14-1006A-A. Tech. rep., D-Wave Systems, Inc

Born M, Fock V (1928) Beweis des adiabatensatzes. Z Phys A Hadrons Nucl 51(3):165–180

Byrd RH, Nocedal J, Waltz RA (2006) Knitro: an integrated package for nonlinear optimization. Springer, Boston, pp 35–59

Calude CS, Dinneen MJ (2017) Solving the broadcast time problem using a D-wave quantum computer. In: Advances in unconventional computing, Springer, pp 439–453

Chiscop I, Nauta J, Veerman B, Phillipson F (2020) A hybrid solution method for the multi-service location set covering problem. In: International conference on computational science (ICCS)

Chvatal V (1983) Linear programming. Macmillan, New York

Dantzig GB, Wolfe P (1960) Decomposition principle for linear programs. Oper Res 8(1):101–111

D-Wave (2016) Programming with QUBOs. Release 2.3, 09-1002A-B. Tech. rep., D-Wave Systems Inc

D-Wave Systems Inc. (2017) Qbsolv 2.0.4. https://github.com/dwavesystems/qbsolv

Farhi E, Goldstone JS, Gutmann JL, Lundgren A, Preda D (2001) A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. Science 292(5516):472–475

Feillet D (2010) A tutorial on column generation and branch-and-price for vehicle routing problems. 4OR Q J Oper Res 8(4):407–424

Feynman RP (1982) Simulating physics with computers. Int J Theor Phys 21(6):467–488

Furman K, El-Bakry A, Song JH (2017) Optimization in the oil and gas industry. Optim Eng 18(1):1–2

Gamrath G (2010) Generic branch-cut-and-price. Master's thesis, Zuse Institute Berlin

Karuppiah R, Furman KC, Grossmann IE (2008) Global optimization for scheduling refinery crude oil operations. Comput Chem Eng 32(11):2745–2766

Khor C, Varvarezos D (2017) Petroleum refinery optimization. Optim Eng 18(4):943–989

Lee H, Pinto JM, Grossmann IE, Park S (1996) Mixed-integer linear programming model for refinery short-term scheduling of crude oil unloading with inventory management. Ind Eng Chem Res 35(5):1630–1641

McGeoch C, Wang C (2013) Experimental evaluation of an adiabatic quantum system for combinatorial optimization. In: CF '13: proceedings of the ACM international conference on computing frontiers, ACM, vol 23, pp 1–11

Mouret S (2010) Optimal scheduling of refinery crude-oil operations. Ph.D. thesis, Carnegie Mellon University

Nannicini G (2020) An introduction to quantum computing, without the physics. SIAM Rev 62(4):936–981

Sahinidis NV (2014) BARON 14.3.1: global optimization of mixed-integer nonlinear programs. User's manual

Syrichas A, Crispin A (2017) Large-scale vehicle routing problems: quantum annealing, tunings and results. Comput Oper Res 87:52–62

Tawarmalani M, Sahinidis NV (2005) A polyhedral branch-and-cut approach to global optimization. Math Program 103:225–249

Venturelli D, Marchand DJJ, Rojo G (2015) Quantum annealing implementation of job-shop scheduling. arXiv:1506.08479

Verstichel J, Kinable J, De Causmaecker P, Berghe GV (2015) A combinatorial benders' decomposition for the lock scheduling problem. Comput Oper Res 54:117–128

Vyskocil T, Djidjev H (2019) Embedding equality constraints of optimization problems into a quantum annealer. Algorithms 12(4):77