Maritime inventory routing problem as a quadratic unconstrained binary optimization problem

Stuart M. Harwood Dimitar Trenev Dimitri Papageorgiou Laurent White

April 22, 2019

This documentation is largely superseded by the publication [3]. That paper describes the formulations that are implemented with the accompanying code. However, for completeness, this document describes some of the basic structure of the maritime inventory routing problem (MIRP), as well as an early version of the "path-based" formulation of it and its transformation to a quadratic unconstrained binary optimization problem (QUBO).

1 Introduction

Inventory routing problems are a class of optimization problems often encountered in logistics and operations research. They entail the simultaneous decision of the routing of various delivery vehicles and management of inventory, with the goal of optimizing some objective (e.g. minimizing transportation cost). MIRPs are a subclass characterized by long travel times and large delivery amounts. MIRPs can be used to optimize global supply chains of commodity products. Because of the maritime aspect of the problem, we will use the terms "ships" to mean the vehicles, and "ports" to mean either the supply or demand nodes.

A general framework and library of instances, termed MIRPLib, was published in [5]. We will use this open test set as inspiration for the form of problem that we will transform. The assumptions behind the core model of MIRPLib are very broad and afford the flexibility to model an inhomogeneous fleet (e.g., ships of different sizes and traveling speeds), split deliveries (i.e. not emptying out the ship on delivery), or multiple ships servicing a port at the same time. However, we will simplify some of these assumptions to obtain the QUBO formulation.

To this end, we aim to use the general vehicle routing problem with time windows (VRPTW) formulation from [2]. The main reason for this is that the core problem in [2] is explicitly given as a form of set partitioning problem. This is a classic problem in discrete optimization, for which a mapping to a QUBO is easily obtained; see for instance [4, §4.1].

In the following sections, we will discuss the specific form of the VRPTW, the transformation to a QUBO, and finally the modifications needed to obtain a reasonable approximation of a MIRPLib problem as a VRPTW.

2 VRPTW as set partitioning problem

The setting of the VRPTW is on a graph with nodes $N \cup \{d\}$ and directed arcs A. One node d is a special "depot" node that is treated differently; otherwise, all other nodes i are customers associated with which are an amount of product demanded q_i which must be delivered in a time window $[a_i, b_i]$. We allow a vehicle to arrive early and wait, but it cannot arrive late (i.e. after

b_i). Each customer is serviced exactly once (i.e. a vehicle cannot fulfill part of the demand, and another one fulfill the rest). Each arc $(i,j) \in A$ has an associated cost $c_{i,j}$ and travel time $t_{i,j}$. A homogeneous fleet of vehicles is available. Each vehicle has capacity (maximum load size) Q. An interesting feature of this formulation is that the number of vehicles available is not explicitly constrained; thus it also combines elements of a fleet-sizing problem. However, through careful construction of the graph, a maximum number of vehicles may be enforced (at the risk of posing an infeasible problem).

The decisions of the VRPTW are routes. A route is a sequence of nodes $(i_0, i_1, \ldots, i_K, i_{K+1})$ satisfying the following constraints. A route begins and ends at the depot: $i_0 = i_{K+1} = d$. Each segment is a valid arc: $(i_k, i_{k+1}) \in A$, for all $0 \leq k \leq K$. The sum of the demands at the visited nodes must be less than vehicle capacity: $\sum_{j=1}^{k} q_{i_j} \leq Q$, for all $k \leq K$. The arrival time at node i_k must be before the time window ends. If we let $T_{i_0} = 0$, then the effective arrival time at node i_{k+1} is given by $T_{i_{k+1}} = \max\left\{a_{i_{k+1}}, T_{i_k} + t_{i_k,i_{k+1}}\right\}$ for all $0 \leq k \leq K$. Then we require $T_{i_k} \leq b_{i_k}$ for all k.

We index the set of routes by the set R. If route $r \in R$ has node sequence $(i_0, i_1, \ldots, i_K, i_{K+1})$, then it has cost $c_r = \sum_{k=0}^{K} c_{i_k,i_{k+1}}$. Finally, define $\delta_{i,r}$ to be a constant with value 1 if route r visits customer $i \in N$ (i.e., one of the nodes in the sequence defining the route is i), and zero otherwise.

The objective of the VRPTW is to minimize the total cost of the planned routes while servicing all customers. Associate a variable x_r which has value 1 if route r is chosen, and zero otherwise, and denote the entire vector of x_r values as x. Then we can write the problem as

$$\begin{aligned} \min_x \ & \sum_{r \in R} c_r x_r \\ \text{s.t.} \ & \sum_{r \in R} \delta_{i,r} x_r = 1, \forall i \in N, \\ & x_r \in \{0,1\}, \forall r \in R. \end{aligned} \tag{1}$$

The equality constraint enforces the requirement that all customer nodes are visited by exactly one route. This is precisely the set partitioning/exact covering problem, where we are trying to "cover" the customer nodes with routes, and math program (1) is precisely how one would formulate the problem as a integer linear program (with the added objective of minimizing weight/cost).

Challenges and classical computing methods

Let $n = |R|$, the cardinality of R. Even with all the requirements we make on a route (e.g. sum of demands must be less than vehicle capacity, etc.), n may be huge (the number of paths of length m on a fully connected graph with m nodes is $m!$). We will not consider this source of difficulty in much detail; we will consider R to be given and index all routes of interest, and view (1) as an "exact" problem for which an optimal or near-optimal solution is our ultimate goal. This provides sufficient challenge as described next.

We have n variables in problem (1) ($x \in \{0,1\}^n$), and thus 2^n possible solutions or values that x may take. This "exponential scaling" is a simple way to see the challenges faced by classical algorithms like branch and bound, implemented by commercial software like CPLEX. To get a sense of what this means in practice, consider the study in [6]. The numerical experiments include the performance of CPLEX (version 12.0, on a single 2.8 GHz dual core processor) on a collection of set partitioning problems coming from an airline crew scheduling problem. For their "small" instances (approximately 800 constraints or nodes and 8,000 variables or routes), CPLEX required less than 30 seconds to solve any instance (presumably, to some tolerance; CPLEX's default relative

gap tolerance is 0.01% [1]). However, the "medium" sized instances in [6] have approximately 1,200 constraints or nodes and 130,000 variables or routes. The authors report that CPLEX could not find any feasible solution in 10 hours of run time. While these results are problem specific and already a little out of date, they provide a useful reference point.

3 VRPTW as QUBO

We discuss how to put (1) into a standard form of a QUBO. There are various conventions and forms that may be followed; an alternative form is the Ising model form that we will discuss later. For the moment, a QUBO is a mathematical optimization problem of the form

$$\begin{aligned} &\min_x \ x^{\mathrm{T}} M x \\ &\text{s.t. } x \in \{0,1\}^n \,, \end{aligned} \tag{2}$$

where M is a $n \times n$ real matrix. A common assumption is that M is upper triangular. Further, when $x \in \{0,1\}$, we have $x^2 = x$, and so a common formulation of (2) becomes

$$\begin{aligned} &\min_x \ \sum_i M_{i,i} x_i + \sum_{i<j} M_{i,j} x_i x_j \\ &\text{s.t. } x \in \{0,1\}^n \,, \end{aligned} \tag{3}$$

where the $(i,j)^{th}$ element of M is $M_{i,j}$.

The challenge is to construct a matrix M so that (3) is equivalent to (1). This is conceptually easy; we would like our matrix to encode the quadratic penalty (or energy) function

$$H : x \mapsto B \sum_r c_r x_r + A \sum_{i \in N} (1 - \textstyle\sum_r \delta_{i,r} x_r)^2$$

for (to be determined) real constants A, B. This transformation is consistent with the general suggestion for binary integer linear programs from [4, §3], as well as the transformation specific to the set partitioning problem from [4, §4.1]. It is easily explained as an exact penalty reformulation of (1) as well.

The main challenge is finding the right values of A and B so that minimization of H is equivalent to (1). This is established in the following result.

Proposition 1. *Assume $A > 0$, $B > 0$ satisfy*

$$\frac{A}{B} > \sum_r |c_r| \,.$$

Then x^* is a solution of $\min\{H(x) : x \in \{0,1\}^n\}$ and problem (1) is feasible if and only if x^* solves problem (1) (where $n = |R|$).

Proof. If x^* solves (1), then it is feasible, so the penalty term is zero: $\sum_i (1 - \sum_r \delta_{i,r} x_r^*)^2 = 0$. Assume for a contradiction that there is an $x^\dagger \in \{0,1\}^n$ with $H(x^\dagger) < H(x^*)$, or

$$B \sum_r c_r x_r^\dagger + A \sum_i (1 - \textstyle\sum_r \delta_{i,r} x_r^\dagger)^2 < B \sum_r c_r x_r^*. \tag{4}$$

If $x^\dagger$ is feasible in (1), then the penalty term is zero, and so $\sum_r c_r x_r^\dagger < \sum_r c_r x_r^*$ which contradicts the optimality of x^*; thus, we must have that $x^\dagger$ is infeasible in (1). Since $x_r^\dagger$ and $\delta_{i,r}$ are $\{0,1\}$-valued for all r and i, the smallest value that $A\sum_i(1-\sum_r \delta_{i,r}x_r^\dagger)^2$ can take is A (since it is infeasible, it cannot be zero). Meanwhile, by the (generalization of) the Cauchy-Schwarz inequality, $-B\sum_r c_r(x_r^\dagger - x_r^*) \leq B\left\|c\right\|_* \left\|x^\dagger - x^*\right\|$ for any norm $\|\cdot\|$ and its dual norm $\|\cdot\|_*$. In particular, using the infinity-norm, we have $-B\sum_r c_r(x_r^\dagger - x_r^*) \leq B\left\|c\right\|_1 \cdot 1$. Using $A \leq A\sum_i(1-\sum_r \delta_{i,r}x_r^\dagger)^2$ and $-B\left\|c\right\|_1 \leq B\sum_r c_r(x_r^\dagger - x_r^*)$ and plugging into (4), we have

$$-B\left\|c\right\|_1 + A < 0,$$

but upon rearranging and using the definition of the one-norm, we see this contradicts the assumption that $\frac{A}{B} > \sum_r |c_r|$. Thus $x^* \in \arg\min_x H(x)$.

Conversely, assume that $x^\dagger$ solves $\min_x H(x)$, and that problem (1) is feasible. First, note that we can scale the objective of (1) by $B > 0$ and not change the optimal solution set. We have $\min_x H(x)$ must be less than or equal to the minimum objective value of (1); $H(x)$ equals the objective of (1) on the feasible set of (1), and the minimization of H is over a superset of the feasible set of (1), so the minimum must be less. Thus, we just need to establish that $x^\dagger$ is feasible for (1). So, assume for a contradiction that $x^\dagger$ is not feasible. By assumption, there exists x^* feasible in (1). Since $x^\dagger$ minimizes H, we have

$$B\sum_r c_r x_r^\dagger + A\sum_i(1-\textstyle\sum_r \delta_{i,r}x_r^\dagger)^2 \leq B\sum_r c_r x_r^*.$$

We can proceed exactly as before to obtain $-B\left\|c\right\|_1 + A \leq 0$, which still contradicts the assumption that $\frac{A}{B} > \sum_r |c_r|$. Therefore $x^\dagger$ is feasible in (1), and thus optimal. □

Constructing M

Let $B = 1$ and $A = \sum_r |c_r| + 1$. The penalty function H is

$$\begin{aligned}
B\sum_r c_r x_r + A\sum_{i\in N}(1-\textstyle\sum_r \delta_{i,r}x_r)^2 &= B\sum_r c_r x_r + A\sum_i \left(1 - \textstyle\sum_r 2\delta_{i,r}x_r + (\sum_r \delta_{i,r}x_r)^2\right) \\
&= B\sum_r c_r x_r + A\sum_i \left(1 - \textstyle\sum_r 2\delta_{i,r}x_r + \sum_{r,r'}\delta_{i,r}\delta_{i,r'}x_r x_{r'}\right) \\
&= B\sum_r c_r x_r + A\,|N| - A\textstyle\sum_r x_r \sum_i 2\delta_{i,r} + A\sum_{r,r'} x_r x_{r'} \sum_i \delta_{i,r}\delta_{i,r'}.
\end{aligned}$$

Assume, for simplicity, that $R = \{1,\ldots,n\}$. Then the diagonal of M is

$$\begin{aligned}
M_{r,r} &= Bc_r - 2A\textstyle\sum_{i\in N}\delta_{i,r} + A\sum_{i\in N}\delta_{i,r}^2 \\
&= Bc_r - A\textstyle\sum_{i\in N}\delta_{i,r}
\end{aligned}$$

where the term $A\sum_i \delta_{i,r}^2$ accounts for quadratic terms, and the simplification is possible since $\delta_{i,r}$ is $\{0,1\}$-valued and so $\delta_{i,r}^2 = \delta_{i,r}$. The upper triangular part is

$$M_{r,r'} = 2A\textstyle\sum_{i\in N}\delta_{i,r}\delta_{i,r'}, \quad r < r',$$

where the factor of two accounts for symmetry (i.e., the terms that would have been in the lower triangle). By assumption, $M_{r,r'} = 0$, for $r > r'$. The constant $A\,|N|$ must be accounted for so that the optimal objective values of (3) and (1) agree.

Ising model form

To get the Ising model formulation of the problem, we introduce the $\{-1,+1\}$-valued variables $s_r = 2x_r - 1$. Consequently, $x_r = 1/2(s_r + 1)$. Using this change of variables for the general expression $x^{\mathrm{T}}Mx$, we get

$$\begin{aligned}\sum_{i,j} M_{i,j}x_i x_j &= \sum_{i,j}(1/4)M_{i,j}(s_i s_j + s_i + s_j + 1)\\ &= d + \sum_i 1/4\textstyle\sum_j (M_{i,j} + M_{j,i})s_i + \sum_{i\neq j}(1/4)M_{i,j}s_i s_j\end{aligned}$$

with the constant $d = 1/4\left(\sum_i M_{i,i} + \sum_{i,j} M_{i,j}\right)$, which also accounts for the quadratic terms since $s_i^2 = 1$ for $s_i \in \{-1,+1\}$.

Using the data for the vehicle routing problem, define $J_{r,r'}$ for $r < r'$ by

$$J_{r,r'} = \frac{M_{r,r'}}{4} = \frac{A}{2}\textstyle\sum_{i\in N}\delta_{i,r}\delta_{i,r'},$$

and note that $J_{r,r'} = M_{r,r'} = 0$ for $r > r'$. Define h_r for $r \in R$ by

$$\begin{aligned}h_r &= \frac{1}{4}(M_{r,r} + M_{r,r}) + \frac{1}{4}\sum_{r':r'\neq r} M_{r,r'} + M_{r',r}\\ &= \frac{B}{2}c_r - \frac{A}{2}\sum_{i\in N}\delta_{i,r} + \frac{1}{4}\sum_{r':r'\neq r} 2A\textstyle\sum_{i\in N}\delta_{i,r}\delta_{i,r'}\\ &= \frac{B}{2}c_r - \frac{A}{2}\sum_{i\in N}\left(\delta_{i,r} - \delta_{i,r}\left(\textstyle\sum_{r'}\delta_{i,r'} - \delta_{i,r}\right)\right)\\ &= \frac{B}{2}c_r - \frac{A}{2}\sum_{i\in N}\delta_{i,r}\left(2 - \textstyle\sum_{r'}\delta_{i,r'}\right).\end{aligned}$$

The vehicle routing problem can then be written as

$$\begin{aligned}&\min_s \textstyle\sum_r h_r s_r + \sum_{r<r'} J_{r,r'}s_r s_{r'}\\ &\text{s.t. } s \in \{-1,+1\}^n,\end{aligned} \tag{5}$$

with a constant

$$d = \frac{1}{4}\left(2\textstyle\sum_r M_{r,r} + \sum_{r<r'} M_{r,r'}\right) = \frac{B}{2}\textstyle\sum_r c_r - \frac{A}{2}\sum_r\sum_{i\in N}\delta_{i,r} + \frac{A}{2}\sum_{r<r'}\sum_{i\in N}\delta_{i,r}\delta_{i,r'}$$

so that the optimal objective value agrees with QUBO (3).

4 Converting MIRPLib instances to VRPTW

The core model for MIRPs given in [5] is an arc-flow-based mixed-integer linear program (MILP). It is fundamentally a ship (vehicle) routing problem on a time-expanded network of supply and demand ports, which also allows for decisions about the inventory levels on ships, and at supply and demand ports. Furthermore, production and consumption rates can also be decided subject to constraints. Data required from the MIRP instance/model are summarized in Table 1. We proceed to describe some of the data simplifications we make and constructions required to obtain a problem in the VRPTW form.

Table 1: Simplified MIRP data used in constructing VRPTW

Data	Meaning/Notes
T	Set of time periods, $T = \{0, 1, \ldots, n_T\}$
J	Set of supply/demand ports
$\widehat{A}$	Arcs, subset of $J \times J$
$\widehat{c}_a$	cost to traverse arc a
$\widehat{t}_a$	time to traverse arc a
$\widehat{Q}$	(Homogeneous) vessel capacity
R_j	Per-unit price for product discharged at port j
F_j	Amount of product that can be loaded/discharged at port j in a time period
$D_{j,t}$	Consumption/production in port j in time period t
S_j	Capacity/maximum inventory in port j (assume $S_j \geq \widehat{Q}$)
I_j^0	Initial inventory at port j

4.1 MIRP data simplifications

Homogeneous ships

A critical assumption is that of fleet homogeneity; specifically, that the speed each ship, the capacity of each ship, the arcs that each ship may traverse, and the cost of traversing each arc are the same. This is an assumption of the VRPTW formulation (1) that is difficult to relax. Consequently, we will focus on and adapt those MIRPLib instances that have homogeneous fleets.

Network simplifications

The MIRP model is on a time-expanded network; a node is a (port,time) pair. One side effect is that this allows for, e.g. seasonal dependence on travel times. We will simplify this, and assume that there exists a reduction of the network to a "static" one, $(J, \widehat{A})$, where J is the set of ports, and $\widehat{A}$ are the allowed arcs (at any time) between them. Similarly, we assume that we can then obtain costs $\widehat{c}_a$ and travel time $\widehat{t}_a$ for each arc a. Combined with the assumption of fleet homogeneity, the costs and travel times can be used for all ships in the fleet.

Fixed and known production/consumption/loading/discharging

The core MIRP model allows for the decision of the consumption/production rate in each time period. For the purposes of the VRPTW, we will assume that the consumption/production rate $D_{j,t}$ is known and fixed for each time period t and for each port j. The core MIRP model also allows for decisions about the loading/discharging rate of product onto/off the ships. We will similarly assume that the loading and discharging rates F_j are known and fixed for each port j.

Full load and discharge

Another assumption we will make is that ships will load and discharge fully. This is not a particularly unrealistic assumption in many maritime vehicle routing problems. Furthermore, when combined with fleet homogeneity and fixed consumption rates, it allows us to determine meaningful time windows for each port.

4.2 VRPTW demand levels and vehicle capacity

The network of the VRPTW (detailed in the following section) will have a collection of nodes associated with each port $j \in J$. If port j is a demand port, the corresponding nodes have demand level $\widehat{Q}$ (a full discharge from a ship). If port j is a *supply* port, the corresponding nodes have demand level $-\widehat{Q}$ (a full load onto a ship).

Whereas the VRPTW implicitly treats the depot node d as a supply/source node, we would like the possibility that a valid route allows a ship to visit an alternating sequence of supply and demand ports. By treating a supply port as a node with negative demand we can achieve that. Since a valid route in the VRPTW must satisfy the condition that the cumulative demand at any node in the route is less than the vehicle capacity, we can enforce the logic that a ship must load product before discharging it by setting the vehicle capacity Q in the VRPTW to zero.

4.3 Determining customer time windows

Perhaps the main challenge is how to define meaningful time windows for the nodes. It is in this procedure that domain knowledge could play a role, and affect how closely the obtained VRPTW approximates the MIRP. In what follows, we propose a fairly simple approach.

Consider a demand port j, which we decided will have a demand level $\widehat{Q}$. Its initial inventory is I_j^0. Its capacity is S_j. The earliest a ship can arrive is the smallest value t such that $I_j^0 - \sum_{t' \leq t} \left|D_{j,t'}\right| + \widehat{Q} \leq S_j$. At this time period, enough inventory will have been depleted that a full shipload can be discharged into inventory.

However, in the VRPTW, there is no need to use a discretized view of time, and we can obtain a sharper lower bound for the time window. Define $D_j^C : t \mapsto D_{j,t'}$ if $t \in [t', t'+1)$, as a (piecewise) continuous version of the consumption at demand port j. Then we can define the lower bound of the time window as the smallest t such that $I_j^0 - \int_{[0,t]} \left|D_j^C(t')\right| dt' + \widehat{Q} \leq S_j$. The end of the time window is then the largest t such that $I_j^0 - \int_{[0,t]} \left|D_j^C(t')\right| dt' \geq 0$. This is the latest a ship could arrive before the port completely runs out of inventory.

In the maritime inventory routing setting, we would like a ship to visit an alternating sequence of supply and demand ports. To achieve this, we construct extra nodes corresponding to our current set of ports, but with different time windows; conceptually they are the same port but at different times (similar to the time-expanded network of MIRP). To define these nodes for demand ports, we can index them by the corresponding port, and by how many dischargings have already occurred. Again, they all have the same demand level ($\widehat{Q}$). The time windows are simple to determine as well; if we have a node $n = (j, q)$, then it corresponds to the demand port j at which q dischargings have occurred. The time window is $[a_n, b_n]$, where a_n is the smallest t such that $I_j^0 - \int_{[0,t]} \left|D_j^C(t')\right| dt' + (q+1)\widehat{Q} \leq S_j$, whereas b_n is the largest t such that $I_j^0 - \int_{[0,t]} \left|D_j^C(t')\right| dt' + q\widehat{Q} \geq 0$.

We proceed similarly for supply ports. Node $n = (j, q)$ corresponds to the supply port j at which q loadings have already occurred. For the time windows, the earliest a ship can arrive at supply port j that has already been visited q times is the smallest t such that $I_j^0 + \int_{[0,t]} \left|D_j^C(t')\right| dt' - (q+1)\widehat{Q} \geq 0$ (after enough product is available for another full shipload). As well, the latest a ship can arrive is the largest t such that $I_j^0 + \int_{[0,t]} \left|D_j^C(t')\right| dt' - q\widehat{Q} \leq S_j$ (before inventory capacity is exceeded).

Question: What if these time windows overlap? Conjecture: We can easily post-process a solution to obtain one that obeys the assumptions under which the time windows were constructed; perhaps this is really a question of berth limits (if, e.g. three time windows overlap, then the port

must have at least three berths).

4.4 Modifying travel time and costs

The base arc costs and travel times $\widehat{c}_a$ and $\widehat{t}_a$ are modified to account for revenue received and loading/discharging times. Let $j(n)$ equal the port corresponding to node n. For all arcs incoming to a (demand) port, we subtract from the cost the revenue received for selling the full shipload of product: $c_{n,n'} = \widehat{c}_{j(n),j(n')} - \widehat{Q}R_{j(n')}$. Letting $R_j = 0$ for any supply port j, we obtain a general rule.

Similarly, we add the loading/discharging time to the travel time of any inbound arcs: $t_{n,n'} = \widehat{t}_{j(n),j(n')} + \widehat{Q}/\left|F_{j(n')}\right|$. This enforces a ship to fully load/discharge before the end of the time window at a particular node.

4.5 Designing the network to enforce ship initial conditions

While the VRPTW formalism does not explicitly consider a fixed number of vehicles, ships for long-haul maritime shipping can be a significant investment, and so in reality a fixed number are available, and may be anywhere in the world at the start of the time horizon. To enforce this, we add dummy nodes, one for each ship, which are the only nodes that have arcs incoming from the depot. From these dummy nodes, we have arcs going to the node corresponding to the physical location and time at which the ship is first available. This can capture details like a ship being in the middle of a long voyage at the start of the time horizon of interest. These dummy nodes, like any other node, must be visited exactly once by a route in the solution; thus a ship must be used exactly once, although we can add arcs from the dummy node immediately back to the depot, allowing for a ship to "exit" without ever being used.

4.6 Generating routes

There are many options here. In this work, a greedy search heuristic is used; see [3].

References

[1] IBM ILOG CPLEX optimization studio CPLEX parameters reference. `https://www.ibm.com/support/knowledgecenter/SSSA5P_12.6.3/ilog.odms.studio.help/pdf/paramcplex.pdf`. Accessed: 30 August 2018.

[2] Martin Desrochers, Jacques Desrosiers, and Marius Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations research*, 40(2):342–354, 1992.

[3] Stuart Harwood, Claudio Gambella, Dimitar Trenev, Andrea Simonetto, David Bernal, and Donny Greenberg. Formulating and solving routing problems on quantum computers. *IEEE Transactions on Quantum Engineering*, 2:1–17, 2021.

[4] Andrew Lucas. Ising formulations of many NP problems. *Frontiers in Physics*, 2:5, 2014.

[5] Dimitri J. Papageorgiou, George L. Nemhauser, Joel Sokol, Myun-Seok Cheon, and Ahmet B. Keha. MIRPLib–a library of maritime inventory routing problem instances: Survey, core model, and benchmark results. *European Journal of Operational Research*, 235(2):350–366, 2014.

[6] Abdelouahab Zaghrouti, François Soumis, and Issmail El Hallaoui. Integral simplex using decomposition for the set partitioning problem. *Operations Research*, 62(2):435–449, 2014.