Received January 26, 2021; accepted March 1, 2021; date of publication March 3, 2021;
date of current version April 2, 2021.

Digital Object Identifier 10.1109/TQE.2021.3063635

Quantum Algorithms for Mixed Binary Optimization Applied to Transaction Settlement

LEE BRAINE[1], DANIEL J. EGGER[2], JENNIFER GLICK[3], AND STEFAN WOERNER[2] (Member, IEEE)

[1]Chief Technology Office, Barclays, London E14 5HP, U.K.
[2]IBM Quantum, IBM Research Zurich, 8803 Rüschlikon, Switzerland
[3]IBM Quantum, IBM Thomas J. Watson Research Center, Ossining, NY 10562 USA

Corresponding author: Stefan Woerner (wor@zurich.ibm.com).

ABSTRACT In this article, we extend variational quantum optimization algorithms for quadratic unconstrained binary optimization problems to the class of mixed binary optimization problems. This allows us to combine binary decision variables with continuous decision variables, which, for instance, enables the modeling of inequality constraints via slack variables. We propose two heuristics and introduce the transaction settlement problem to demonstrate them. Transaction settlement is defined as the exchange of securities and cash between parties and is crucial to financial market infrastructure. We test our algorithms using classical simulation as well as real quantum devices provided by IBM quantum.

INDEX TERMS Mixed binary optimization (MBO), quadratic unconstrained binary optimization (QUBO), quantum finance, quantum optimization, transaction settlement.

I. INTRODUCTION

Quantum computers process information using the laws of quantum mechanics. They are well suited for a number of tasks, such as simulating quantum mechanical systems [1], [2] and factoring [3]. Additionally, quantum computers can provide a quadratic speed-up over classical Monte Carlo simulations, which may be used to evaluate risk [4], [5] and price financial derivatives [6], [7]. Another possible application area for quantum computers is optimization, particularly combinatorial optimization. It is not believed that quantum computers will be able to solve NP-hard problems in polynomial time [8]. There is, however, a significant effort in designing quantum heuristics that could be practically useful by finding near optimal solutions [9]–[12]. Adiabatic quantum computing [13], [14] and variational algorithms, such as the *variational quantum eigensolver* (VQE) [9] and the *quantum approximate optimization algorithm* (QAOA) [10], are designed to tackle *quadratic unconstrained binary optimization* (QUBO) problems. Importantly, quantum optimization algorithms may benefit from a warm-start obtained from the solution of an efficiently solvable relaxation of the problem. In doing so, the quantum algorithm inherits the performance guarantee of the classical algorithm, as shown for QUBO problems [15], [16]. However, many relevant problems in business and science are *mixed binary optimization* (MBO) problems, with discrete and continuous variables, or with constraints that cannot be modeled as part of a QUBO problem, e.g., inequality constraints. In this article, we introduce an approach to extend the existing quantum methods to more general MBO problem classes.

While quantum algorithms are not yet mature enough to deploy at scale in the market place, it is crucial to investigate their potential to business-relevant problems to one day gain a quantum advantage. We therefore test our algorithm on the *transaction settlement* problem by focusing on securities settlement in capital markets. Here, transaction settlement is the process in which securities (tradeable financial assets, such as shares, bonds, and derivatives) are delivered usually against a payment. This exchange between parties can be facilitated by a clearing house, which also mitigates the counterparty risk [17]. Financial institutions submit the details of the trades (e.g., buy x shares of some company for an amount y of some currency) to the clearing house, which runs a complex optimization algorithm on the resulting batch of transactions while taking into account credit and collateral facilities. The objective is typically to settle as many transactions as possible or to maximize the total value of the settled transactions. Transaction settlement is a difficult optimization problem due to a combination of both the legal constraints that must be satisfied when settling delivery-versus-payment (DvP) transactions and the additional optionality introduced by collateralizing assets and utilizing credit facilities. A variety of approaches are currently employed, ranging in complexity from basic gross settlement systems (which settle on a simple transaction-by-transaction basis) to sophisticated probabilistic techniques, such as simulated annealing (whereby an optimization process is performed to identify a sufficient subset of transactions that can settle and then that subset is actually settled). This is an industry process of systemic importance because of the volume and value of transactions settled, e.g., over \$1.85 quadrillion of securities transactions were processed in 2018 by subsidiaries of the posttrade market infrastructure DTCC [18].

This article is structured as follows. In Section II, we define QUBO problems and discuss their relation to MBO problems. In Section III, we introduce hybrid quantum-classical optimization algorithms for MBO by extending the known algorithms. In Section IV, we map the transaction settlement problem to this framework, and in Section V, we demonstrate the performance of our hybrid algorithm on concrete instances using cloud-based quantum hardware. We conclude in Section VI and discuss open questions and directions of further research.

II. MIXED BINARY OPTIMIZATION

QUBO problems, defined as

$$\min_{x\in\{0,1\}^n} \quad x^T A x + b^T x + c \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$, and $c \in \mathbb{R}$, have a wide range of applications, such as portfolio optimization or the traveling salesman problem, but are hard to solve [12], [19]. Variational quantum algorithms for combinatorial optimization can find good solutions to QUBO problems with n variables once mapped to an Ising Hamiltonian H for n qubits by setting $x_i = (1 - z_i)/2$ for $z_i \in \{-1, +1\}$ and replacing z_i by Pauli Z-matrices σ_i^Z, for $i = 1, \ldots, n$. This allows us, for instance, to use VQE or QAOA to approximate the ground state of H, which corresponds to the optimal solution of the QUBO problem. For further details on how to map QUBO problems to Ising Hamiltonians, we refer the interested reader to Lucas [19].

Since many industry relevant problems, such as portfolio optimization with budget constraints [20] or the knapsack problem and its variants [21], are not QUBO problems, we also consider MBO problems, here defined as

$$\min_{\substack{x\in\{0,1\}^n\\ y\in Y}} \quad x^T A(y) x + b(y)^T x + c(y) \tag{2}$$

where $Y \subset \mathbb{R}^m$ is the feasible set for the continuous variables y and $A : \mathbb{R}^m \to \mathbb{R}^{n\times n}$, $b : \mathbb{R}^m \to \mathbb{R}^n$, and $c : \mathbb{R}^m \to \mathbb{R}$ are given functions of y. MBO problems therefore combine discrete and continuous variables x and y, respectively. Furthermore, for a fixed $y \in Y$, the MBO problem (2) corresponds to the QUBO problem (1). While other definitions of MBO exist, we have chosen this one to be compatible with the variational heuristics we want to extend.

In QUBO problems, we model linear equality constraints of the form $u^T x + v = 0$ by adding quadratic penalty terms $(u^T x + v)^2$ scaled by a large weight λ to the objective, where $u \in \mathbb{R}^n$ and $v \in \mathbb{R}$ are given. This forces optimal solutions to satisfy the constraint to prevent the large penalty. Other types of constraints are usually not possible in QUBO, since they cannot be represented as quadratic penalty terms. In the *quantum alternating operator ansatz*, the mixing operator in QAOA is adjusted to keep the process in the feasible set assuming a feasible initial state [22]. This is a promising, but nongeneric, approach to incorporate constraints.

Since MBO problems allow continuous variables, we may introduce a slack variable $s \geq 0$ to enforce the equality constraint $u^T x + v + s = 0$. This is equivalent to the inequality constraint $u^T x + v \leq 0$, without s. Thus, by allowing binary and continuous variables, we extend the type of constraints we can model. The lower bound on s needs to be included as a continuous constraint defining the feasible set Y.

III. QUANTUM OPTIMIZATION ALGORITHMS FOR MBO

Variational quantum algorithms for QUBO first translate the problem into an n-qubit Ising Hamiltonian H, as discussed in Section II. Classical optimization can then find the values of the parameters θ of a n-qubit trial wavefunction $|\psi(\theta)\rangle$ to, for instance, minimize the expected value

$$\min_{\theta} \quad \langle\psi(\theta)| H |\psi(\theta)\rangle \tag{3}$$

i.e., by applying VQE [9]. The expected value is easily estimated by repeatedly measuring $|\psi(\theta)\rangle$ along the Z-axis as H only consists of σ^Z-terms. Since we consider classical optimization, we can translate every measurement of $|\psi(\theta)\rangle$ into an n-bit string and directly evaluate (1). If we measure $|\psi(\theta)\rangle$ N-times and denote the resulting bit strings by $x^j(\theta)$ where $j = 1, \ldots, N$, the resulting optimization problem is

$$\min_{\theta} \quad \sum_{j=1}^{N} x^j(\theta)^T A x^j(\theta) + b^T x^j(\theta) + c. \tag{4}$$

We may improve the performance of the optimization algorithm by replacing the sample average in problem (4) by another aggregation function, such as the Conditional Value at Risk (CVaR). That is, for a given $\alpha \in (0, 1]$, we only average over the best αN samples, which can help the classical optimization process find better results [12].

In this article, we consider two variational algorithms. First, VQE with a generic trial wavefunction $|\psi(\theta)\rangle$ and second, QAOA, where we construct $|\psi(\theta)\rangle$ based on H. In the following, we discuss how to extend these algorithms to be applicable to MBO problems.

If a generic trial-solution $|\psi(\theta)\rangle$ is given, as in VQE, we can map an MBO problem to a continuous optimization problem

$$\min_{\substack{\theta \\ y \in Y}} \quad \sum_{j=1}^{N} x^j(\theta)^T A(y) x^j(\theta) + b(y)^T x^j(\theta) + c(y) \qquad (5)$$

for which solutions can be approximated using classical optimization schemes.

Let us now consider a rule to derive a trial wavefunction $|\psi_H(\theta)\rangle$ from a given Hamiltonian H, such as given by QAOA. For MBO, every fixed $y \in Y$ defines a QUBO and can, thus, be translated to an Ising Hamiltonian $H(y)$. This allows us to define trial wavefunctions $|\psi_{H(y)}(\theta)\rangle$ that now depend on y. As in (5), we can translate the MBO to a continuous optimization problem, where the measurements $x^j(\theta, y)$ depend on θ as well as y. In summary, $A(y)$, $b(y)$, and $c(y)$ define the Hamiltonian $H(y)$, which in turn defines the θ- and y-dependent trial wavefunction $|\psi_{H(y)}(\theta)\rangle$. These two approaches allow us to extend the existing quantum heuristics from QUBO to MBO enabling us to test variational quantum algorithms on a larger problem class.

Next, we introduce a heuristic that is designed to handle slack variables resulting from modeling inequality constraints explicitly. This can help the classical optimizer used to solve (5) to move out of local minima. Suppose, for simplicity, that we have a problem with only binary variables and a single slack variable, coming from an inequality constraint, formally given by

$$\max_{\substack{x \\ s \geq 0}} x^T A x + b^T x + c + \lambda \left(u^T x + v + s\right)^2 . \qquad (6)$$

The large scalar $\lambda > 0$ enforces $u^T x + v + s = 0$. The new heuristic consists of repeating the following three steps a fixed number of times, or until some predefined termination criterion is met.

1) Perform a preset fixed number of iterations of a classical optimizer to jointly optimize θ and s, and store the resulting samples $x^j(\theta)$ from the last iteration.
2) Analyze every individual sample $x^j(\theta)$ and define
$$s^j = \max\left\{0, -(u^T x^j(\theta) + v)\right\} . \qquad (7)$$
In other words, for each sampled binary vector $x^j(\theta)$, we derive the slack variable s^j such that the constraint $u^T x + v + s = 0$ is satisfied, if possible, or such that we minimize the violation. Evaluate the overall objective value for $x^j(\theta)$ and s^j and identify the candidate solution that performs best, i.e., that achieves the smallest objective value.
3) Fix s^j determined in the previous step and rerun the classical optimizer over θ only for a preset fixed number of iterations.

This simple heuristic can help significantly to move out of local optima of the considered continuous optimization problem in θ and s. The algorithmic parameters, such as number of cycles and number of iterations per cycle, need to be determined according to the problem of interest. As we show in the following section, applying a classical optimizer directly to solve (5) often gets stuck in local optima, whereas our new heuristic reliably found the globally optimal solution of the considered test case. The introduced algorithm can be easily extended to more complex problems with additional continuous variables and multiple inequality constraints/slack variables.

The quantum algorithms our methods rely upon are heuristics and, therefore, cannot be analyzed using $\mathcal{O}$-notation. Instead, the potential quantum advantage must be studied empirically on a case-to-case basis and current research suggests that if an advantage can be gained, it will be for problem sizes with more than 100 variables [23], [24]. The energy landscapes of the QAOA and the VQE are typically nonconvex and contain many local minima. Current research has focused on strategies, such as reinforcement learning [25] and multistart methods [26], to navigate this landscape. The quality of the VQE and QAOA solutions increases with the depth of the quantum algorithm [23], which can lead to deep quantum circuits, which exhaust the coherence time of noisy quantum hardware. Furthermore, the optimization problems must be mapped to the native connectivity of the qubits using SWAP gates, which further increases the circuit depth [27]. It is therefore paramount to increase the quality of the hardware so that such algorithms may find practical relevance [28]. In addition, implementing small problems on noisy hardware shows how to leverage novel hardware platforms, such as tunable couplers [2] and pulse-level optimizations [29]–[31], to accelerate the development of quantum algorithms.

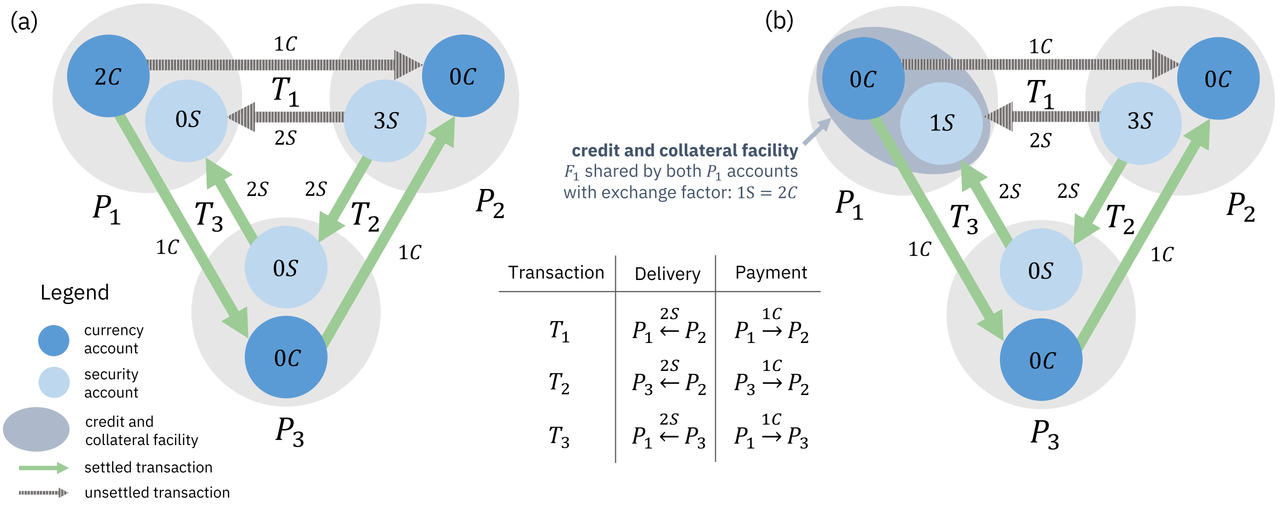
IV. TRANSACTION SETTLEMENT

In this section, we introduce the transaction settlement problem in which parties continuously submit the details of the trades to a clearing house. All of the corresponding DvP transactions recorded by the clearing house during a predefined time interval, plus all the unsettled transactions remaining from the previous batch, form a new batch of I transactions submitted by K parties. Γ_k is the set of transactions of party k. The clearing house must then settle as many transactions as possible within the given batch. The ith transaction involving a party k is described by a sparse vector $\vec{v}_{ik}$ of length J, the total number of security and currency accounts, with only two nonzero entries, one for the security (delivery) and one for the currency (payment). The transaction vectors of two parties k and k' that participate in a transaction i sum to zero: $\vec{v}_{ik} + \vec{v}_{ik'} = \vec{0}$. To each transaction i, we associate a binary variable x_i to indicate settlement.

We, thus, seek to maximize the weighted sum of settled transactions

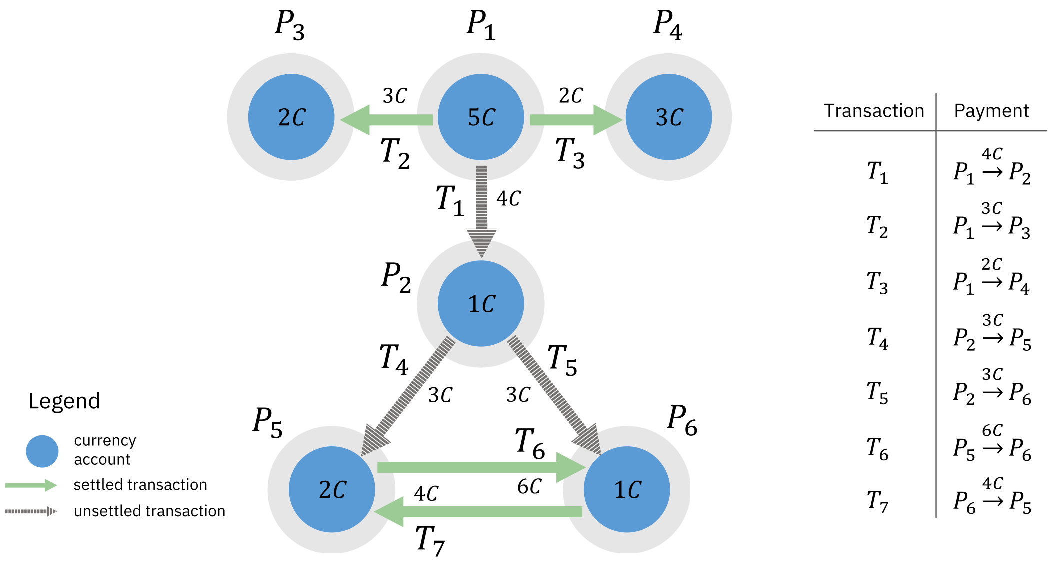
$$\max_{\vec{x}} \quad \sum_{i=1}^{I} w_i x_i$$

$$\text{subject to:} \quad \vec{b}_k + \sum_{i \in \Gamma_k} x_i \vec{v}_{ik} \geq \vec{l}_k \quad \forall_k. \qquad (8)$$



Transaction	Delivery	Payment
T_1	$P_1 \xleftarrow{2S} P_2$	$P_1 \xrightarrow{1C} P_2$
T_2	$P_3 \xleftarrow{2S} P_2$	$P_3 \xrightarrow{1C} P_2$
T_3	$P_1 \xleftarrow{2S} P_3$	$P_1 \xrightarrow{1C} P_3$

FIGURE 1. Diagrams illustrating a batch of three DvP transactions T_1, T_2, T_3 between three parties P_1, P_2, P_3. Each party has a currency account C and security account S, with amounts shown in the interior circles. Each transaction is shown with a pair of arrows and the corresponding amount of security and currency involved. In (b), we consider a simplified credit and collateral facility (CCF) (with a net zero amount of credit) shared by both accounts of P_1, with an exchange factor between currency and security of $1S = 2\,C$. In both (a) and (b), the optimal solution settles transactions T_2 and T_3, as indicated by the pairs of solid green arrows. With the modification of P_1's account balances in (b), none of the transactions could settle without the presence of the CCF F_1, which P_1 can use to convert security into currency to facilitate the transaction with P_3.



Transaction	Payment
T_1	$P_1 \xrightarrow{4C} P_2$
T_2	$P_1 \xrightarrow{3C} P_3$
T_3	$P_1 \xrightarrow{2C} P_4$
T_4	$P_2 \xrightarrow{3C} P_5$
T_5	$P_2 \xrightarrow{3C} P_6$
T_6	$P_5 \xrightarrow{6C} P_6$
T_7	$P_6 \xrightarrow{4C} P_5$

FIGURE 2. Diagram illustrating a batch of seven payments $T_1, \ldots, T_7$ between six parties $P_1, \ldots, P_6$. Each party has a single currency account C, with amounts shown in the interior circles. Each transaction is shown with a single arrow and the corresponding amount of currency involved. Several optimal solutions exist that settle four total transactions, e.g., payments T_2, T_3, T_6, T_7, shown here with solid green arrows to differentiate the unsettled transactions indicated by grey hatched arrows.

The weight ω_i may, for instance, be the total monetary value of transaction i, or $\omega_i = 1\ \forall i$ if we seek to optimize the number of settled transactions. The K vector constraints in (8) feature the balance vector $\vec{b}_k$, which encodes the securities and currencies that party k owns before any transactions settle. Additionally, party k may be granted credit for some or all accounts, which is encoded by $\vec{l}_k$. There are $K \times J$ constraints in (8), but, in practice, the sparsity of $\vec{b}_k$, $\vec{v}_{ik}$, and $\vec{l}_k$ reduces the number of constraints. Thus, the optimization described by (8) must ensure that the credit limits are not exceeded. The transaction settlement optimization problem is complex and may feature dependence chains and even cyclical dependencies between transactions, as seen in Figs. 1 and 2.

Each account (associated with a security or currency) owned by a party can optionally participate in a simplified CCF, which permits the account to contribute to a collateral pool and also use the resulting credit pool. Several accounts owned by a party can participate in the same CCF, permitting shared pools (in principle, several credit and collateral facilities can be defined for each party). The accounts owned by

TABLE 1 Notation Used Throughout This Article to Define the Transaction Settlement Optimization Problem

Inputs	
transactions	$i \in \{1, \ldots, I\}$
parties	$k \in \{1, \ldots, K\}$
transactions of party k	Γ_k
currencies/securities	$j \in \{1, \ldots, J\}$
transaction values	$\vec{v}_{ik} \in \mathbb{R}^J$
transaction weights	$\vec{w} \in \mathbb{R}^I_{\geq 0}$
credit limits	$\vec{l}_k \in \mathbb{R}^J$
balance	$\vec{b}_k \in \mathbb{R}^J$
exchange factor	$\vec{r}_k \in \mathbb{R}^J_{\geq 0}$
credit pool limit	$p_k \in \mathbb{R}_{\leq 0}$
Decision Variables	
settle transaction or not	$\vec{x} \in \{0, 1\}^I$
slack variables	$\vec{s}_k \in \mathbb{R}^J_{\geq 0}$

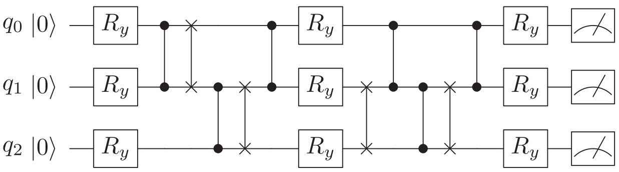


FIGURE 3. Variational circuit used for the first two problem instances of three DvP transactions between three parties without and with a simplified CCF, (12) and (13), respectively. The depth $d = 2$ variational form is composed of $d + 1$ layers of parameterized R_y gates (parameters omitted in the figure) interleaved with d fully connected entangling blocks of controlled-phase gates, for a total of $n(d + 1)$ independent angles, for n qubits. The linear connectivity of the *Valencia* quantum processor introduces swap gates for two-qubit gates between nonadjacent qubits. In practice, this circuit can be further optimized, e.g., cancelling a pair of cnots from an adjacent swap and cnot. A similar circuit is used for simulation of the third problem instance of seven payments between six parties, but with depth $d = 3$ and $n = 7$ qubits.

party k may add to or draw from the credit pool such that a total credit amount p_k is not exceeded. This can be modeled as an additional continuous constraint

$$\vec{r}_k \cdot \vec{l}_k \geq p_k \tag{9}$$

for each party k that participates in a CCF. The exchange factors $\vec{r}_k$ represent the conversion of a currency/security in one account into a currency/security of another account when exchanged via the facility and may be party dependent. In this scenario, the credit limits $\vec{l}_k$ can now be variable, instead of fixed for each account, to reflect the amount contributed to or drawn from the shared pool. The notation we use to formulate this problem is summarized in Table 1.

We now transform problem (8), following Section II, into the form given in (2), which may be solved using the variational quantum solvers discussed in Section III. Nonnegative slack variables $\vec{s}_k$ are used to cast the inequality constraints into equality constraints, i.e., (8) becomes

$$\begin{aligned} &\max_{\substack{\vec{x} \\ \vec{s}_k \geq 0}} \quad \sum_{i=1}^{I} w_i x_i \\ &\text{subject to:} \quad \vec{b}_k + \sum_{i \in \Gamma_k} x_i \vec{v}_{ik} = \vec{l}_k + \vec{s}_k \quad \forall_k. \end{aligned} \tag{10}$$

Next, we map this constrained optimization problem to an unconstrained optimization problem by transforming the equality constraint into a quadratic penalty term scaled by a large number $\lambda \geq 0$

$$\max_{\substack{\vec{x} \\ \vec{s}_k \geq 0}} \left[\sum_{i=1}^{I} w_i x_i - \lambda \sum_{k=1}^{K} \left(\vec{b}_k + \sum_{i \in \Gamma_k} x_i \vec{v}_i - \vec{l}_k - \vec{s}_k \right)^2 \right]. \tag{11}$$

In this form, we can solve the optimization problem using the methods of Section III, where the number of qubits needed to represent the trial wavefunction is equal to the number of transactions under consideration. The nonnegativity constraint on the slack variables as well as constraint (9) is handled by the classical optimizer as they do not involve the decision variables x_i. Our formulation of the transaction settlement problem assumes that the order in which transactions are settled within a batch can be ignored. Therefore, only the settled transactions affect the balances.

We now describe the transaction settlement problem for three concrete test cases, with numerical results discussed later in Section V. The first two test cases involve three transactions between three parties of a single currency and security, whereas the third test case is a batch of seven payments of a single currency between six parties. In all three problems, we maximize the total *number* of transactions.

The first problem instance, shown in Fig. 1(a), does not feature a CCF, whereas the second instance, shown in Fig. 1(b), utilizes one for both accounts held by the first party. Starting with the first instance, we solve the following:

$$\begin{aligned} &\max \quad (x_1 + x_2 + x_3) \\ &\text{subject to} \\ &P_1 : \quad (2, 0) + (-1, 2)x_1 + (-1, 2)x_3 \geq (0, 0) \\ &P_2 : \quad (0, 3) + (1, -2)x_1 + (1, -2)x_2 \geq (0, 0) \\ &P_3 : \quad (0, 0) + (-1, 2)x_2 + (1, -2)x_3 \geq (0, 0) \end{aligned} \tag{12}$$

where, for each party P_i, the account balances, transaction values, and account credit limits are indicated by vectors (C, S) specifying the amounts of currency C and security S. The optimal solution settles transactions T_2 and T_3, that is, $(x_1, x_2, x_3) = 011$, whereas a greedy algorithm may be able to settle only a single transaction T_1.

Next, we consider the batch of three transactions in the presence of a simplified CCF. In this model, illustrated in Fig. 1(b), the two accounts held by party P_1 can contribute to and draw from a shared credit pool F_1 containing a net zero balance. We adjust the balance of P_1 such that no transactions can settle without the CCF, whereas the optimal solution settles T_2 and T_3 with the facility (note that settling only T_1 is a valid solution, though not optimal). The constraints for P_1

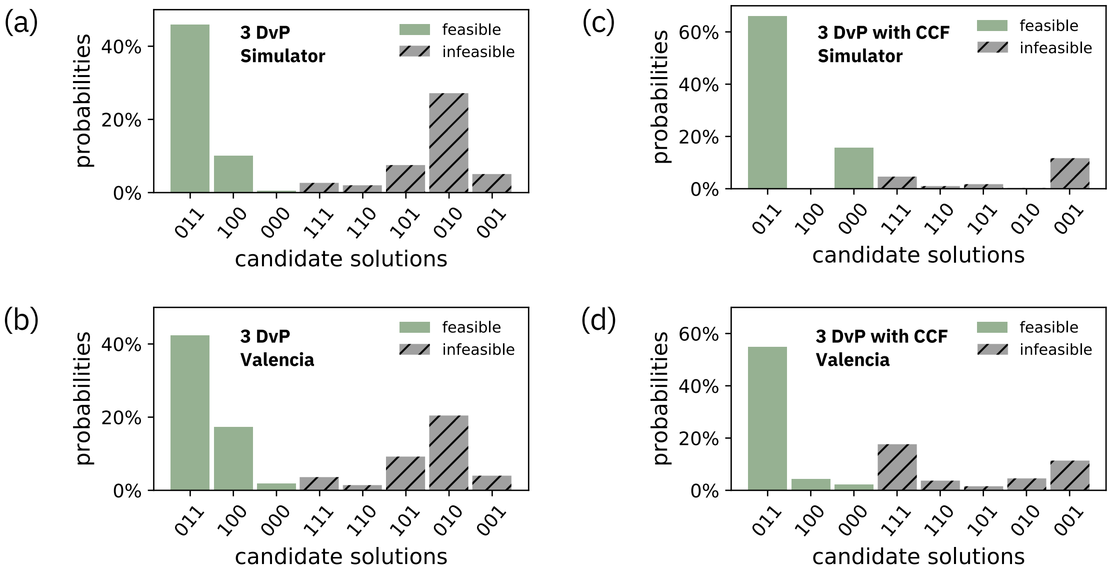


FIGURE 4. Results for (a), (b) the three DvP transactions of (12) and (c), (d) the three DvP transactions with a CCF of (13). (a) and (c) show the results from classical simulation, whereas (b) and (d) show the results from the *Valencia* 5-qubit quantum processor. Candidate solutions are grouped into feasible (green bars) and infeasible (grey hatched bars) sets and then sorted left to right by descending value of the objective function (number of trades settled). In all four cases, the optimal solution $(x_1, x_2, x_3) = 011$, indicating that transactions T_2 and T_3 settle, is sampled with the highest probability. Both problems used the R_y variational circuit of depth $d = 2$ shown in Fig. 3, a penalty factor $\lambda = 10^3$ for adding constraints to the objective function, the CVaR aggregation function with $\alpha = 25\%$, and 150 iterations of COBYLA (without additional cycles to explicitly handle slack variables, see Section III).

are then modified from those in (12) to

$$\begin{aligned} &\max \quad (x_1 + x_2 + x_3)\\ &\text{subject to}\\ &P_1: \quad (0,1) + (-1,2)\,x_1 + (-1,2)\,x_3 \geq (l_C, l_S)\\ &F_1: \quad l_C + 2l_S = 0,\ \ l_C \geq -2,\ \ l_S \geq 0\\ &P_2: \quad (0,3) + (1,-2)\,x_1 + (1,-2)\,x_2 \geq (0,0)\\ &P_3: \quad (0,0) + (-1,2)\,x_2 + (1,-2)\,x_3 \geq (0,0). \end{aligned} \tag{13}$$

Party P_1 now has an effective credit line for each account l_C and l_S. We assume that the CCF F_1 has a zero net balance so that l_C and l_S sum to zero with the appropriate exchange factors between accounts. In particular, the credit limit l_C on the currency account cannot extend past $-1S \times (2C/S) = -2C$, where $1S$ is the starting balance of the security account and $1S = 2C$ is the exchange factor. A similar analysis holds for the security account: $l_S \geq 0$.

The final problem instance, illustrated in Fig. 2, involves a batch of seven payments of a single currency between six parties. The objective and full list of constraints is the following set of equations:

$$\begin{aligned} &\max(x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7)\\ &\text{subject to}\\ &P_1: \quad 5 - 4x_1 - 3x_2 - 2x_3 \geq 0\\ &P_2: \quad 1 + 4x_1 - 3x_4 - 3x_5 \geq 0\\ &P_3: \quad 2 + 3x_2 \geq 0\\ &P_4: \quad 3 + 2x_3 \geq 0\\ &P_5: \quad 2 + 3x_4 - 6x_6 + 4x_7 \geq 0\\ &P_6: \quad 1 + 3x_5 + 6x_6 - 4x_7 \geq 0 \end{aligned} \tag{14}$$

where the constraints for P_3 and P_4 are always satisfied and can be removed from the model. Several optimal solutions exist, for example, with payments T_2, T_3, T_6, T_7 settling. In the next section, we discuss numerical results for each of these three problem instances.

V. RESULTS

We solve each problem instance described in Section IV using the hybrid/quantum classical algorithms outlined in Section III. Following Nannicini [11], we use COBYLA [32] as the classical optimizer, since it is well suited to handle continuous variable constraints. For each problem instance, we classically simulate the hybrid quantum-classical algorithm using Qiskit [33] and, for the first two problem instances, compare the simulations to runs on the *ibmq_valencia* 5-qubit backend of IBM quantum. We employ readout-error mitigation using *Qiskit Ignis* [33], [34] to correct measurement errors. The variational circuit we use, shown in Fig. 3, is composed of $d + 1$ layers of single-qubit Y-rotations interleaved with d blocks of controlled-Z gates [1]. We set a quadratic penalty factor of $\lambda = 10^3$ for the problem constraints [see (11)], and utilize the CVaR aggregation function mentioned in Section III with $\alpha = 25\%$ in place of the sample average for 8192 samples drawn from each trial wavefunction. The results of the quantum algorithm for the first two

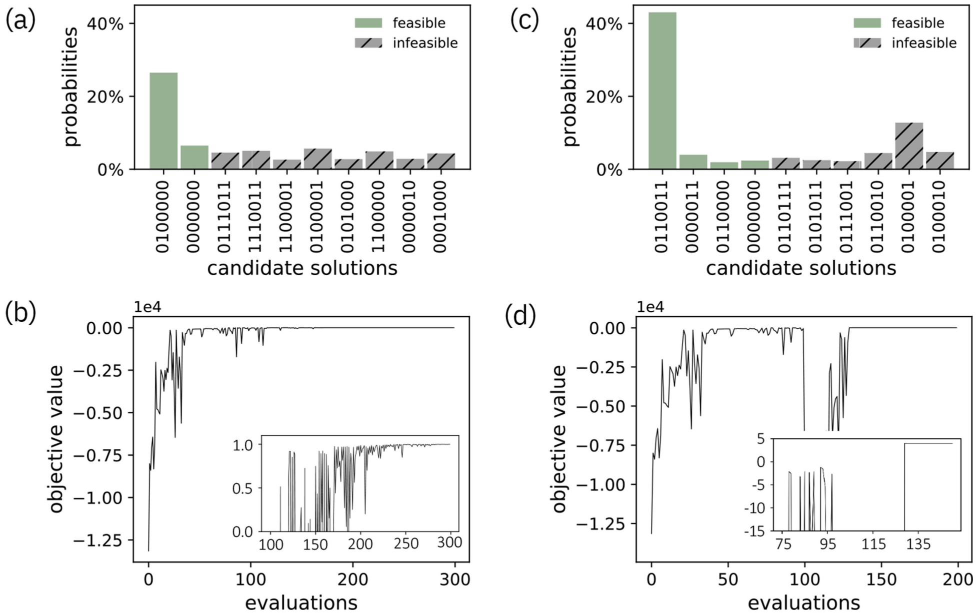


FIGURE 5. Results for six parties and seven payments of a single currency using classical simulation. The problem is first approached without (a), (b) and then with (c), (d) explicit handling of slack variables, i.e., the heuristic described in Section III. In the first approach, we apply COBYLA with 300 iterations, whereas in the second, we run 1 cycle with 100 + 100 iterations of COBYLA (the first 100 iterations on the full problem and the second 100 iterations on the parameters of the variational form only). We show here the probabilities of the ten most frequent states resulting from the optimization process (a), (c), where candidate solutions are grouped into feasible (green bars) and infeasible (grey hatched bars) sets and then sorted left to right by descending value of the objective function (number of trades settled). In (a), the feasible solutions sampled are suboptimal, the best one having a single payment settled. In (b), an optimal solution with four payments settled is found with a probability larger than 40%. In addition, we show the progress during the optimization, where it can clearly be seen how the explicit handling of slack variables (d) achieves the optimal solution while the direct application of COBYLA gets stuck in a local optimum (b). Results shown here used an R_y variational circuit of depth $d = 3$, a penalty factor $\lambda = 10^3$ for adding constraints to the objective function, and the CVaR aggregation function with $\alpha = 25\%$.

problem instances using a simulator and the *Valencia* quantum processor are shown in Fig. 4.

We compare the results for the first problem, given in (12), from classical simulation in Fig. 4(a) to those found from running on *Valencia* in Fig. 4(b), and similarly for the second problem, given in (13) in Fig. 4(c) and (d), respectively. For both problem instances, with and without the CCF, we find good agreement between simulation and experiment. The optimal solution that settles T_2 and T_3 (the bitstring 011) appears with the highest probability.

For the third problem, given in (14), we simulate classically and compare the algorithm with explicit handling of slack variables and without, as described in the heuristic of Section III. In the latter case, we find that the objective function never reaches the maximum value of four (i.e., four settled transactions), but saturates at unity, as seen in Fig. 5(b). As a result, only suboptimal feasible solutions are sampled [see Fig. 5(a)]. In contrast, with explicit handling of slack variables, Fig. 5(d) shows how the objective function achieves the maximum value within the second part (step 3, evaluations 100–200) of the first cycle, and that we now sample a globally optimal solution settling payments T_2, T_3, T_6, T_7 (the bitstring 0 110 011) in Fig. 5(c). Note that the heuristic finds only one out of the three possible globally optimal solutions. To influence the actual solution found, we could adjust the weights w_i in (10), for instance, to prioritize transactions with a larger volume.

All the algorithms described here have been implemented using Qiskit [33], which has been used to access the classical simulator as well as the quantum hardware, and the detailed settings can be found in the captions of Figs. 4 and 5.

VI. CONCLUSION

While quantum algorithms are not yet ready to be applied at scale in the market place, quantum technologies have greatly progressed in recent years. It is, therefore, important to understand how quantum algorithms can be applied to business-relevant problems even though large-scale demonstrations cannot be tackled with today's quantum hardware. In this article, we take a step in this direction by extending the existing work on quantum optimization to the class of MBO and propose hybrid quantum/classical heuristics to address such problems. This allows us, for instance, to model inequality constraints, which significantly extends the applicability of these algorithms. Inequality constraints are crucial for many applications, such as portfolio optimization, where to date, quantum algorithms were always assuming a cardinality equality constraint instead of a real budget constraint. We tested our algorithms on the transaction settlement problem for batches of payment and DvP securities transactions, which is a difficult and important optimization challenge in capital markets. Better algorithms could increase settlement efficiency (in terms of the number of transactions settled for a given batch), thereby minimizing the time intervals between trade and settlement. This could reduce replacement cost risk (the risk of loss of unrealized gains due to delay in settlement), reduce liquidity risk (substantial liquidity pressures can emerge if a participant fails to settle its net funds debit positions), and reduce credit risk (particularly if there is a decline in value of the securities).

The algorithms presented in this article are exploiting only very little of the structure of a problem. Although this allows for great flexibility, it also clearly indicates directions of future research. By restricting to problems with certain properties, such as, for instance, convexity in the continuous variables, it may be possible to derive algorithms that outperform the current state-of-the-art and even derive some guarantees on convergence under assumptions of the underlying quantum algorithms. The proposed structure also allows for the use of problem specific variational forms, such as those proposed by QAOA, which may further improve the performance.

ACKNOWLEDGMENT

Barclays and International Business Machines Corporation 2019. BARCLAYS is a registered trademark of Barclays PLC, all rights are reserved. IBM, the IBM logo, and ibm.com are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. The current list of IBM trademarks is available at https://www.ibm.com/legal/copytrade.

REFERENCES

[1] A. Kandala *et al.*, “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets,” *Nature*, vol. 549, no. 7671, pp. 242–246, 2017, doi: 10.1038/nature23879.

[2] M. Ganzhorn *et al.*, “Gate-efficient simulation of molecular eigenstates on a quantum computer,” *Phys. Rev. Appl.*, vol. 11, no. 4, 2019, Art. no. 0 44092, doi:10.1103/PhysRevApplied.11.044092.

[3] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” *SIAM J. Comput.*, vol. 26, no. 5, pp. 1484–1509, 1997, doi: 10.1137/S0097539795293172.

[4] S. Woerner and D. J. Egger, “Quantum risk analysis,” *npj Quantum Inf.*, vol. 5, pp. 1–8, 2019, doi:10.1038/s41534-019-0130-6.

[5] D. J. Egger, R. G. Gutierrez, J. C. Mestre, and S. Woerner, “Credit risk analysis using quantum computers,” *IEEE Trans. Comput.*, to be published, doi: 10.1109/TC.2020.3038063.

[6] N. Stamatopoulos *et al.*, “Option pricing using quantum computers,” Quantum, vol. 4, 2020, Art. no. 291, doi:10.22331/q-2020-07-06-291.

[7] P. Rebentrost, B. Gupt, and T. R. Bromley, “Quantum computational finance: Monte Carlo pricing of financial derivatives,” *Phys. Rev. A*, vol. 98, 2018, Art. no. 0 22321, doi: 10.1103/PhysRevA.98.022321.

[8] E. Bernstein and U. Vazirani, “Quantum complexity theory,” *SIAM J. Comput.*, vol. 26, no. 5, pp. 1411–1473, 1997, doi: 10.1137/S0097539796300921.

[9] A. Peruzzo *et al.*, “A variational eigenvalue solver on a photonic quantum processor,” *Nature Commun.*, vol. 5, 2014, Art. no. 4213, doi: 10.1038/ncomms5213.

[10] E. Farhi, J. Goldstone, and S. Gutmann, “A quantum approximate optimization algorithm,” 2014, *arXiv:1411.4028*.

[11] G. Nannicini, “Performance of hybrid quantum-classical variational heuristics for combinatorial optimization,” *Phys. Rev. E*, vol. 99, 2019, Art. no. 0 13304, doi: 10.1103/PhysRevE.99.013304.

[12] P. K. Barkoutsos, G. Nannicini, A. Robert, I. Tavernelli, and S. Woerner, “Improving variational quantum optimization using CVaR,” *Quantum*, vol. 4, 2020, Art. no. 256, doi:10.22331/q-2020-04-20-256.

[13] M. W. Johnson *et al.*, “Quantum annealing with manufactured spins,” *Nature*, vol. 473, no. 7346, pp. 194–198, 2011, doi: 10.1038/nature10012.

[14] A. M. Childs, E. Farhi, and J. Preskill, “Robustness of adiabatic quantum computation,” *Phys. Rev. A*, vol. 65, no. 1, 2001, Art. no. 012322, doi: 10.1103/PhysRevA.65.012322.

[15] D. J. Egger, J. Marecek, and S. Woerner, “Warm-starting quantum optimization,” 2020, *arXiv:2009.10095*.

[16] R. Tate, M. Farhadi, C. Herold, G. Mohler, and S. Gupta, “Bridging classical and quantum with SDP initialized warm-starts for QAOA,” 2020, *arXiv:2010.14021*.

[17] D. Loader, *Clearing, Settlement and Custody*. Oxford, U.K.: Butterworth-Heinemann, 2002, doi: 10.1016/C2018-0-04499-0.

[18] DTCC, DTCC—Annual reports—Financial performance, 2018. Accessed: Sep. 30, 2019. [Online]. Available: http://www.dtcc.com/annuals/2018/#/financial-performance

[19] A. Lucas, “Ising formulations of many NP problems,” *Frontiers Phys.*, vol. 2, pp. 1–14, 2014, doi: 10.3389/fphy.2014.00005.

[20] D. Venturelli and A. Kondratyev, “Reverse quantum annealing approach to portfolio optimization problems,” *Quantum Mach. Intell.*, vol. 1, pp. 17–30, 2019, doi: 10.1007/s42484-019-00001-w.

[21] H. Kellerer, P. Ulrich, D. Pisinger, *Knapsack Problems*. Berlin, Germany: Springer-Verlag, 2004, doi:10.1007/978-3-540-24777-7.

[22] S. Hadfield, Z. Wang, B. O’Gorman, E. Rieffel, D. Venturelli, and R. Biswas, “From the quantum approximate optimization algorithm to a quantum alternating operator ansatz,” *Algorithms*, vol. 12, no. 2, 2019, Art. no. 34, doi: 10.3390/a12020034.

[23] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, “Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices,” *Phys. Rev. X*, vol. 10, 2020, Art. no. 0 21067, doi: 10.1103/PhysRevX.10.021067.

[24] G. G. Guerreschi and A. Y. Matsuura, “QAOA for max-cut requires hundreds of qubits for quantum speed-up,” *Sci. Rep.*, vol. 9, 2019, Art. no. 6903, doi: 10.1038/s41598-019-43176-9.

[25] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev, and P. Balaprakash, “Learning to optimize variational quantum circuits to solve combinatorial problems,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, pp. 2367–2375, 2020, doi: 10.1609/aaai.v34i03.5616.

[26] R. Shaydulin, I. Safro, and J. Larson, “Multistart methods for quantum approximate optimization,” in *Proc. IEEE High Perform. Extreme Comput. Conf.*, 2019, pp. 1–8, doi: 10.1109/HPEC.2019.8916288.

[27] M. P. Harrigan *et al.*, “Quantum approximate optimization of non-planar graph problems on a planar superconducting processor,” *Nat. Phys.*, vol. 17, pp. 332–336, 2021, doi: 10.1038/s41567-020-01105-y.

[28] P. Jurcevic *et al.* “Demonstration of quantum volume 64 on a superconducting quantum computing system,” 2020, *arXiv:2008.08571*.

[29] P. Gokhale, A. Javadi-Abhari, N. Earnest, Y. Shi, and F. T. Chong, “Optimized quantum compilation for near-term algorithms with openpulse,” 2020, *arXiv:2004.11205*.

[30] N. Lacroix *et al.* “Improving the performance of deep quantum optimization algorithms with continuous gate sets,” *PRX Quantum*, vol. 1, 2020, Art. no. 0 20304, doi: 10.1103/PRXQuantum.1.020304.

[31] J. P. T. Stenger, N. T. Bronn, D. J. Egger, and D. Pekker, “Simulating the dynamics of braiding of Majorana zero modes using an IBM quantum computer,” 2020, *arXiv:2012.11660*.

[32] M. Powell, *Advances in Optimization and Numerical Analysis*. Dordrecht, The Netherlands: Kluwer, 1994, pp. 51–57, doi: 10.1007/978-94-015-8330-5.

[33] H. Abraham *et al.*, “Qiskit: An open-source framework for quantum computing,” 2019, doi: 10.5281/zenodo.2562110.

[34] A. Dewes *et al.*, “Characterization of a two-transmon processor with individual single-shot qubit readout,” *Phys. Rev. Lett.*, vol. 108, 2012, Art. no. 0 57002, doi: 10.1103/PhysRevLett.108.057002.