IEEE Transactions on Quantum Engineering

Received 26 June 2023; revised 1 September 2023; accepted 10 September 2023; date of publication 13 September 2023; date of current version 19 October 2023.

Digital Object Identifier 10.1109/TQE.2023.3314839

Approaching Collateral Optimization for NISQ and Quantum-Inspired Computing (May 2023)

MEGAN C. GIRON[1], GEORGIOS KORPAS[1,2], WAQAS PARVAIZ[1,2], PRASHANT MALIK[3], AND JOHANNES ASPMAN[2]

[1]HSBC Lab, Innovation and Ventures, HSBC, E14 5HQ London, U.K.
[2]Department of Computer Science, Czech Technical University in Prague, Prague 2, Czechia
[3]Markets and Securities Services, HSBC, E14 5HQ London, U.K.

Corresponding authors: Megan C. Giron; Georgios Korpas; Waqas Parvaiz (e-mail: megan.giron@hsbc.com; georgios.korpas@hsbc.com; waqasparvaiz@live.co.uk).

ABSTRACT Collateral optimization refers to the systematic allocation of financial assets to satisfy obligations or secure transactions while simultaneously minimizing costs and optimizing the usage of available resources. This involves assessing the number of characteristics, such as the cost of funding and quality of the underlying assets to ascertain the optimal collateral quantity to be posted to cover exposure arising from a given transaction or a set of transactions. One of the common objectives is to minimize the cost of collateral required to mitigate the risk associated with a particular transaction or a portfolio of transactions while ensuring sufficient protection for the involved parties. Often, this results in a large-scale combinatorial optimization problem. In this study, we initially present a mixed-integer linear programming formulation for the collateral optimization problem, followed by a quadratic unconstrained binary optimization (QUBO) formulation in order to pave the way toward approaching the problem in a hybrid-quantum and noisy intermediate-scale quantum-ready way. We conduct local computational small-scale tests using various software development kits and discuss the behavior of our formulations as well as the potential for performance enhancements. We find that while the QUBO-based approaches fail to find the global optima in the small-scale experiments, they are reasonably close suggesting their potential for large instances. We further survey the recent literature that proposes alternative ways to attack combinatorial optimization problems suitable for collateral optimization.

INDEX TERMS Financial management, mathematical programming, optimization, quantum annealing (QA), quantum computing, simulated annealing (SA).

I. INTRODUCTION

In the context of a financial transaction, wherein one party lends assets to another, the lender assumes a credit risk arising from the possibility that the counterparty may default on its obligations. This risk also arises in derivatives' transactions where the party "in-the-money" is exposed to the party "out-of-the-money." To mitigate this risk, the borrower is required to provide low-risk securities (such as cash, bonds, or equities) to the lender for the duration of the transaction. This practice, known as *collateralization* [1], serves as a form of security against loan defaults as the lender can seize these assets to offset any losses resulting from default. The value of the collateral received is expected to be commensurate with the outstanding exposure, in order to effectively counterbalance the associated risk.

Often, a bilateral contract (or schedule) is formed to agree on the terms under which securities can be considered collateral, the process of evaluating the value of these assets, and other regulations. The relevant party may then accordingly select the assets they post to the counterparty. For large financial institutions, such as banks, this can involve a pool of numerous assets to choose from which need to be distributed amongst a portfolio of various counterparties (other banks, hedge funds, central banks, etc.). Each asset has an associated opportunity cost, which is a measure of how valuable the asset would be if it were used for another purpose, as well as a cost related to the risk of posting to a particular counterparty amongst other administrative costs. The bank must, therefore, carefully consider their choice of transactions to reduce these costs. However, the magnitude of

the possible combinations of allocations for large institutions makes this a time-consuming process. In addition, poor collateral management can have significant consequences. The 2008 financial crisis was partly due to the collateralization of high-risk securities and led to the bankruptcy of some of the largest financial institutions, among other consequences [2].

This crisis has led to the reformation of many financial processes through the implementation of regulations, such as Basel III [3], Dodd–Frank Act [4], EMIR [5], as well as motivating academic research with regard to how collateral can be better managed [6], [7]. These studies relating to collateral management are generally centered on financial theories, such as risk aversion and its global financial impact. A crucial aspect of collateral management is the development of an automated process that selects the optimal allocations. Despite the importance of collateral optimization (ColOpt), the literature surrounding this topic is sparse due to the competitive advantage these strategies offer to financial institutions.

Naturally, linear programming algorithms can provide a framework for tackling such problems [8]. Specifically, ColOpt is suitable to be implemented using mixed-integer linear program (MILP) solvers, such as the ones available with IBM CPLEX [9], Gurobi [10], or Mosek [11]. The success of a given ColOpt instance, or the quality of its solution, is dependent on a clear mapping between the business problem and the mathematical formulation as well as the choice of the precise implementing algorithm. The benefit of using numerical optimization is that we attain the allocation selections in a single process, which is in contrast to the other proposed models, such as "ranking-based," "economic-cost," and "waterfall" models, which are sequential in nature, rather than automated [12]. However, there are several limitations of MILP solvers when applied to problems, such as ColOpt, which potentially involve complex nonlinearities and large-scale datasets. For example, MILP solvers have exponential worst-case complexity and can take a significant amount of time to solve large-scale complex optimization problems. While having convergence certificates is very desirable, the exponential complexity is often a problem for many ColOpt instances that involve a large number of decision variables and constraints and it is not uncommon to either have long solution times or even infeasibility. That said, MILP solvers are the standard in both industrial and academic applications but the community is very keen on exploring alternative approaches.

Different avenues to (approximately) solve such computationally challenging problems could be provided through alternative computational models. For example, in [13], IBM's True Spike neuromorphic computer [14] was utilized to find approximate solutions for the graph partitioning problem. More interest is directed toward quantum computing [15]. These computers rely on quantum-mechanical effects for storing and processing information. However, because of the fragile environment required for these effects to occur, the realization of fault-tolerant quantum computers is still a difficult task.

Despite this, it is believed that noisy intermediate-scale quantum (NISQ) era devices can provide an advantage in the finance industry since these business use cases can be well formulated for near-term quantum devices [16], [17]. The field of "quantum finance" can be divided into three sections: stochastic modeling (for example, quantum alternatives to Monte Carlo simulations) [17], [18], [19], [20], machine learning [17], [18], [21], [22], [23], [24], and optimization [22], [25], [26], [27], [28], [29], [30], [31], all of which have had a recent gain in interest.

Very often, in this context, the prototypical optimization use case is that of (Markowitz) portfolio optimization. While this use case shares a few similarities with ColOpt, there are a few fundamental differences as well. The constraints of ColOpt, seem to be more involved even within the simplifications we present in Section III. An important difference is also the fact that our objective function is inherently linear, at first glance. However, several "tricks" can be performed in order to end up with a well-behaved formulation suitable for a variety of Ising NISQ or hybrid solvers.

Quantumly, there are many approaches to follow in order to approximate better solutions for a variety of NP-hard problem instances encountered in finance. The main three approaches (listed below) have a common theme: conversion of the original mathematical formulation of the problem from a linear program (LP) formulation to a quadratic unconstrained binary optimization (QUBO) problem. The reason lies in the inherent ability of quantum or hybrid approaches by modeling the Ising model type of systems (see Appendix B). There an optimization problem is mapped to the classical Hamiltonian of the Ising model, where its ground state encodes the optimum. As a matter of fact, many NP-hard problems, including Karp's list of 21 NP-hard problems, are known to admit at least one formulation of the Ising model [32]. The QUBO or Ising approach can be used and problems can be mainly tackled as follows.

1) Using variational quantum algorithms (VQAs) [33], such as the quantum approximate optimization algorithm (QAOA) [34], on gate-based quantum computers (e.g., IBM's superconducting quantum computer). A variety of tests have been performed in this context with significant (qubit) resource improvements recently [35].
2) Using quantum annealing (QA) [36], [37], [38], [39], [40] on adiabatic quantum computers (quantum annealers), such as the D-wave hardware [41] (see [42] for an application to portfolio optimization).
3) Using quantum-inspired methods that can be understood as using the QUBO formulation of the problem of interest with any approach that ranges from simulated annealing (SA) [43] on high-performance clusters or digital annealing [44], such as Fujitsu's field-programmable gate array (FPGA)-based "quantum-inspired" classical hardware time to solution digital annealing unit (DAU) [45].

These approaches are very promising and it is widely expected that near-term quantum computers, as well as the dedicated quantum-inspired hardware, may have a good chance to provide computational or business advantage[1] [46], [47] in the near term. Note that the formulation we will present below in Section III is hardware agnostic and, in that sense, any of the above approaches would be suitable in principle.

Additionally, there exists evidence that the class of problems that NISQ computers can solve is not a subset[2] of BPP [48]. However, we realize that their heuristic nature can conceal their applicability, especially when considering problem instances with sizes suitable for high-quality MILP solvers. Such a result was reported in [49], in which Pusey-Nazzaro et al. discuss how D-wave's 2000Q machine (as well as classical SA) failed to even come close to the branch-and-bound approaches [50], [51] for solving certain instances of the Knapsack Problem (KnapsackProb) (see Section II). One can potentially try to use the VQA approach instead [18], [33], [52], [[53], for example]. In this context, Nannicini [54] reported that there seems to be a lack of transparent computational advantage in introducing entanglement when VQAs are used. This lack of computational advantage is expected to some extent due to the well-known local minima problem that VQAs exhibit [55] (see [56] for the proposed way to avoid this problem). In addition, bias in the noise of circuits that implement VQAs can unfavorably affect the convergence ratios [57]. Other limitations are discussed in [58].

However, performance advantages have been showcased in a variety of applications within the context of digital and QA solutions, tensor networks, and analog and digital (gate-based) quantum computing. For example, Ebadi et al. [59] studied the maximal independent set (MaxIndSet) problem and found a superlinear quantum speedup, as opposed to classical solutions, when considering very hard graphs. From the point of view of computational complexity, the MaxIndSet problem is not particularly different from other NP-hard problems, such as the KnapsackProb (which actually is weakly NP-complete) or other NP-hard problems that admit an MILP formulation, and as such, the results of [59] are encouraging for other problems as well. An interesting benchmark test [60] using the D-wave machine showed very promising results. It is worth mentioning that D-wave recently announced [61] the largest quantum simulation done in different contexts (spin glasses) to what is of interest here.

In this article, we study the ColOpt problem in detail and provide an MILP formulation that we use as a testbed for formulating a QUBO version of ColOpt, which makes it suitable for feeding into quantum and quantum-inspired solvers and performing small-scale simulations of such solvers and comparing to MILP. Specifically, in our MILP formulation, we choose our objective to be minimization of the cost of posting collateral (different approaches to the objective of ColOpt have been proposed, for example, see [12]). We survey and try numerous QUBO encodings and find that, modulo emulator limitations, in such small instances, the quantum-inspired formulations perform well enough to be promising for implementation on real quantum or quantum-inspired hardware for very large instances. The rest of this article is organized as follows. In Section II, we provide an overview of some of the different QUBO formulations for the KnapsackProb problem. This section serves both as an introduction to the concepts used throughout the article and to inform the ColOpt problem that follows. In Section III, we provide the MILP formulation of the ColOpt problem as well as a few QUBO proposals. In Section IV, we provide a few numerical results using the formulation of Section III. Finally, Section V concludes this article.

We want to clarify that while our article investigates various QUBO formulations for the KnapsackProb, our ultimate goal is to apply this information to the ColOpt problem. Specifically, we plan to use the best-performing QUBO formulations from our KnapsackProb study to formulate and solve the collateral optimization problem using QUBO. However, we would like to note that our article does not aim to provide an empirical comparison between quantum and classical approaches for solving MILPs, given the limited computational resources available to us (see [62] for work on the comparison of classical and quantum (adiabatic) optimization, wherein the authors discovered "surprising" results favoring the D-wave machine). Additionally, we focus on small problem instances only, and we acknowledge that all the results presented in our study are heuristic in nature. Nevertheless, based on the literature results on the potential of QUBO formulations (quantumly and not only), we believe that the formulations we propose may have value in tackling larger instances of the collateral optimization problem and, therefore, may warrant further investigation.

In summary, the main objective of our article is to present a case study on the formulation and approach of the ColOpt problem using quantum computing techniques, with the overarching aim of advancing the ongoing effort toward achieving "quantum advantage" in practical applications.

II. INTERLUDE WITH THE KNAPSACK PROBLEM

To inform and ensure our formulations and the subsequent computations, we perform a simple test using a small KnapsackProb instance. In essence, KnapsackProb involves determining the optimal approach to filling a knapsack of capacity W with the highest possible value from a set of n items that have specific sizes and corresponding values (see Table 1). This problem is of interest due to its simplicity to formulate and its simple constraints. For us, it is further interesting since we view the ColOpt problem as a (somewhat complicated) generalization.

[1]By the term "business advantage," we mean enough computational resource savings or improvement in performance, enough to justify the adoption of the underlying technology.

[2]Chen et al. [48] define NISQ as the class that contains all problems that can be solved by a polynomial-time probabilistic classical algorithm with access to a noisy quantum device and where BPP⊆NISQ⊆BQP.

TABLE 1. Input Data Used of the Specific KnapsackProb instance Considered in This Article

Object label	A	B	C	D	E	F	G	H	I	J
Weight	23	31	29	44	53	38	63	85	89	82
Value	92	57	49	68	60	43	67	84	87	72

Total capacity of the knapsack is 165.

Given the large number of "hard" constraints of the ColOpt problem, see Section III, we aim to compare formulation for small instances of the Knapsack problem with the hope to inform our approach for collateral optimization. The "standard route" to encode constraints to a QUBO model is by using (balanced) slack variables for penalization [63] (see Appendix B). A different approach is using "unbalanced" penalization [64] that turns out to be particularly useful for QAOA solutions as it reduces the resources required by a gate-based machine. The MILP formulation of the KnapsackProb is a well-known and straightforward approach. In the problem instance we consider, we are given a set of weights $w \in \mathbb{Z}^n_{\geq 0}$ and their corresponding values $v \in \mathbb{Z}^n_{\geq 0}$, and the objective is to maximize the total value of the items that can be packed into a knapsack subject to a given weight limit. The problem can be mathematically defined as follows:

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} v_i x_i \\ \text{s.t.} \quad & \sum_{i=1}^{n} w_i x_i \leq W \end{aligned} \tag{1}$$

where W is the maximum weight limit (threshold) of the knapsack and x_i is the binary variable representing whether the ith item is to be placed in the knapsack. The best running-time algorithm for solving the KnapsackProb is based on dynamic programming with pseudopolynomial complexity $\mathcal{O}(d_n W)$ [65], where d_n is the number of distinct weights available while near-linear running-times' algorithms (in d_n, W) were documented in [66].

The specific problem instance we consider in our study, as outlined in [67], comprises ten items and possesses a known optimal solution. We leverage this knowledge to heuristically evaluate the effectiveness of our approach and guide our efforts toward tackling the larger ColOpt problem, as discussed in Section III. The relevant (toy) data pertaining to the items in this problem instance can be found in Table 1.

Although the KnapsackProb is known to be (weakly) NP-complete, simple instances, such as the one we consider in this study, can be efficiently solved by a range of classical solvers. For our experimental analysis, we used the HiGHS [68] and GLPK [69] solvers, both of which yielded solutions that were in agreement with the known optimal solution of the problem instance, as expected, while for the QUBO-formulated problem, we tested the open-source Julia libraries ToQUBO.jl [70], Qiskit's optimization module [71], the open-source Python library PyQubo [72] (in both cases operating under SA), and the emulation of the proprietary digital annealer of Fujitsu [45].

A. QUADRATIC UNCONSTRAINED BINARY OPTIMIZATION

The QUBO model can be applied to a wide range of combinatorial optimization problems that are known to be NP-hard, such as the maximum cut, minimum vertex cover, multiple knapsack, and graph coloring problems. Its applications span a diverse set of domains, including the automotive industry [73], portfolio optimization [74], [75], traffic flow optimization [75], job scheduling [76], [77], [78], railway conflict management [79], bioinformatics [80], and others [81]. An extensive list of QUBO formulations for interesting problems can be found in [82].

Due to its one-to-one mapping to the Ising Hamiltonian, QUBOs have become a fundamental element of quantum-inspired computing. Both the digital annealer developed by Fujitsu and the adiabatic quantum computers manufactured by D-wave systems (as well as other vendors, such as Qilimanjaro) employ the QUBO model to address complex optimization problems. Additionally, several approaches based on tensor networks seem to be suitable for a variety of QUBO-formulated optimization problems [83], [84], [85], [86]. Although QUBO is particularly well suited to these technologies, it can also be employed in NISQ devices using algorithms, such as the QAOA. As such, the QUBO model is an important tool for quantum optimization with potential applications across a range of quantum computing platforms and formulating constrained problems as such highly affects the quality of the solutions obtained.

Let us summarize the basics of QUBO via a graph problem. Given an undirected graph $G = (V, E)$ with a vertex set $V = \{1, 2, \ldots, N\}$ connected by the edge set $E = \{(i, j), \}$, $i, j \in V$, the cost function is defined as follows:

$$\min \quad \sum_{i=1}^{N} A_{ii} x_i + \sum_{i=1}^{N-1} \sum_{j>i}^{N} A_{ij} x_i x_j \tag{2}$$

where $x \in \{0, 1\}$ are the binary variables and the elements $A_{ij} \in \mathbb{R}^{N \times N}$ are the problem instance parameters.

At its most fundamental level, a QUBO can be expressed as follows:

$$\min \quad x^T Q x + b \tag{3}$$

where the decision matrix $Q \in \mathbb{R}^{N \times N}$ contains the problem instance and $b \in \mathbb{R}$ is a constant offset term.

By using a suitable change of variables $x_i = \frac{1-\sigma_i}{2}$, (2) can be mapped onto the Ising model Hamiltonian as follows:

$$H = -\sum_{j} h_j \sigma_j - \sum_{j<k} J_{jk} \sigma_j \sigma_k \tag{4}$$

where $\sigma \in \{-1, 1\}^N$ are the (classical) spins, $h \in \mathbb{R}^N$ is the magnetic field, and $J \in \mathbb{R}^{N \times N}$, $\text{diag}(J) = 0$, the spin–spin interaction symmetric matrix between adjacent spins j and k. See Appendix B for more details. The problem to be solved

then is

$$\min_{\sigma_i \in \{-1,1\}} \quad H. \tag{5}$$

For the QUBO formulation of KnapsackProb, we can take several slack-based approaches, including off-the-shelf LP-to-QUBO converters, such as Qiskit's QuadraticProgramToQubo class and methods as well as the Julia package ToQUBO.jl. Further to that, we can perform a "custom" slack formulation and also use the unbalanced penalization method we mentioned previously.

B. SLACK VARIABLE FORMULATION

In the process of converting MILPs to QUBOs, it is common practice to introduce a slack variable, $S \in \mathbb{R}_{\geq 0}$ (whose purpose will be discussed shortly), for each linear inequality and transform it into an equivalent linear equality. Subsequently, a penalty term is constructed based on the slack variable, and the term is squared as per the standard approach, as outlined in [63] (see also [87]).

A variety of different slack-based QUBO formulations exist for the KnapsackProb [88]. Here, the corresponding penalization term with weight $\lambda_0 \in \mathbb{R}_+$ is given by the equality

$$\lambda_0 \left(\sum_{i=1}^{n} w_i x_i - W + S \right)^2 = 0. \tag{6}$$

The purpose of the auxiliary slack variable S is to reduce this term to 0 once the constraint has been satisfied, $0 \leq S \leq \max_x \sum_i^n w_i x_i - W$. In practice, S is decomposed into binary representation using variables $s_k \in \{0, 1\}$ as follows:

$$S = \sum_{k=1}^{N_s} 2^{k-1} s_k. \tag{7}$$

The parameter N_s corresponds to the number of binary variables required to represent the maximum value that can be assigned to the slack variable, and in the case of KnapsackProb, $N_s = \lceil \log_2(W) \rceil$, where $\lceil x \rceil$ is the ceiling function. Formulating the slack variable as in (7) is commonly referred to as the "log encoding."

The full QUBO formulation for the KnapsackProb takes the form of maximizing the objective function

$$\sum_{i}^{n} v_i x_i - \lambda_0 \left(\sum_{i}^{n} w_i x_i - W + \sum_{k=1}^{N} 2^{k-1} s_k \right)^2 \tag{8}$$

which can be understood as an augmented Lagrangian [89]. Alternatively, there are other QUBO formulations of the KnapsackProb that we could decide to use that follow a similar Lagrangian paradigm. For example, we can instead consider maximizing the following objective function:

$$\sum_{i}^{n} v_i x_i - \lambda_0 \left(\sum_{i}^{n} w_i x_i - \sum_{k=1}^{W} k s_k \right)^2 - \lambda_1 \left(1 - \sum_{k=1}^{W} s_k \right)^2. \tag{9}$$

In this formulation, known as one-hot encoding, the number of slack bits is equal to the capacity of the knapsack W. Here, an additional penalty term is required to enforce only one of these slack bits to be assigned a value of 1. A drawback of this formulation is that the binary input length for the slack variables scales linearly with the values of the constraints; hence, it can lead to an unreasonably large number of bits for problem instances with large W that can exhaust available resources. This issue becomes quite relevant in Section IV.

C. BALANCED SLACK-BASED APPROACHES

In this section, we give an overview of the approaches used to determine solutions for different balanced formulations of the KnapsackProb QUBO. The known optimal solution for our small instance that we consider corresponds to an objective value of 309 and uses the full capacity of the knapsack.

Off-the-Shelf Converters: The first off-the-shelf approach we tried is ToQUBO.jl, an open-source Julia package that automatically reformulates a variety of optimization problems, including MI(L)Ps, to a QUBO. The user can use the JuMP [90] package to build the MILP form of KnapsackProb. ToQUBO.jl provides six ways for encoding variables into binary representations. This includes the logarithmic and one-hot approaches mentioned above as well as other less well-known techniques. For continuous decision variables, this can be very useful since the user can provide a tolerance factor to manage the upper bound on the representation error caused by the binarization. Additionally, ToQUBO.jl works in conjunction with QUBODrives.jl, a companion package that provides common API to use QUBO sampling and annealing machines, such as D-wave's simulated annealer and, with license, the quantum annealer via DWaveNeal.jl. We make use of ToQUBO.jl to employ both of the aforementioned binary encodings, both successfully finding the optimal solution. However, the unbalanced QUBO formulation is a recent development, which has not been widely adopted; hence, its encoding is not available as a part of the ToQUBO.jl package.

Another off-the-shelf converter is provided by Qiskit's optimization module that includes functionality for automatically transforming quadratic programs into QUBOs (the binary property allows us to use such). This transformation can be accomplished by first initializing a QuadraticProgram and, subsequently, utilizing the QuadraticProgramToQubo class to convert it into a QUBO via the log-encoding method for slack variables. The module allows the formulated QUBO to feed into several algorithms used by Qiskit to solve optimization problems, such as SamplingVQE or QAOA; however, the user can also extract the coefficient matrix to use with other solvers. We input this coefficient matrix into neal and Fujitsu's digital annealer, where we found that both solvers are able to reach optimum, with the caveat that a large number of runs are needed compared with other methods. For larger problem instances, this can become computationally expensive and, thus, may be an inadequate choice for ColOpt.

There are several variations of the balanced slack-based approach to QUBO formulations, which we summarize now.

Log Encoding: This approach refers to implementing (8). The number of slack bits required for this approach, for the instance of interest, is $N_s = 8$. Two different regimes for the weight of the penalty term, λ_0, are checked:

1) where the penalty term and the cost function have equal weighting ($\lambda_0 = 1$);
2) where the penalty term is more important than the cost function ($\lambda_0 = 1 \times 10^4$).

We utilized the neal package to implement D-wave's simulated annealer as our heuristic optimizer in both scenarios. The simulated annealer successfully returned the optimal solution in both cases, which was consistent with the results obtained from the classical solvers. The (emulation of the) digital annealer from Fujitsu was employed for both scenarios as well. Similar to the case with neal, it successfully produced optimal solutions in both instances.

One-Hot Encoding: We repeated the previously described process but with a focus on analyzing the solutions of (9). As previously mentioned, this type of formulation requires a large number of bits to encode the slack variables for large knapsack capacities, in this case, 165 bits. In this approach, we once more explored various weight regimes for λ_0 and λ_1, in relation to the weight of the cost function term. By selecting $\lambda_0 = 10^{-1}$ and $\lambda_1 = 10^3$, we consistently identified the optimal solution for both neal and Fujitsu cases.

D. UNBALANCED PENALIZATION APPROACH

Given that the number of qubits required scales proportionally with the number of variables, we sought to employ the methodology proposed by Montanez-Barrera et al. [64] in order to eliminate the need for slack variables. To accomplish this, we adopted an approximation technique that creates penalty terms that take on small values when the constraint is fulfilled and large values when it is violated. We rearrange the inequality to define an auxiliary function

$$h(x) = \sum_{i}^{n} w_i x_i - W \leq 0. \tag{10}$$

Using the exponential function $f(x) := e^{h(x)}$, the constraint on $h(x)$ is satisfied. However, since only linear and quadratic terms may be encoded into a QUBO, it is necessary to use a second-order Taylor approximation of $f(x)$. For weights λ_0 and λ_1, the resulting QUBO for the KnapsackProb problem reads

$$\sum_{i}^{n} v_i x_i + \lambda_0 \left(\sum_{i}^{n} w_i x_i - W \right) + \lambda_1 \left(\sum_{i}^{n} w_i x_i - W \right)^2 = 0. \tag{11}$$

It should be noted that Montanez-Barrera et al. [64] investigated this type of QUBO formulation for different instances of the traveling salesperson problem (TravSalProb), bin packing problem (BinPackProb), and KnapsackProb, and found that the minimum energy eigenvalue of the corresponding Hamiltonian did not necessarily coincide with the optimal solution. However, the optimal solution was always found to be amongst the lowest energy eigenvalues, which enhances the confidence for this approach in large-scale experiments.

To clarify, the advantage of reformulating the problem as an unbalanced QUBO rather than the traditional balanced approaches is that one significantly reduces the number of variables, and hence, the bits required to represent the problem, which in effect reduces the resource cost, as well as the search space of the optimal solution. However, drawbacks associated with the unbalanced approach are that, because of the use of a heuristic penalization function, our constraints are less strict and the groundstate of the corresponding Ising Hamiltonian is less likely to coincide with the optimal solution of the problem at hand.

By using the weights provided by Montanez-Barrera et al. [64] as a quick check, we were able to reach optimality for this KnapsackProb instance for both PyQUBO and Fujitsu unbalanced approaches, signifying that the unbalanced formulation exhibits (some) robustness. For larger KnapsackProb instances tested, we produced close to optimal results which, however, would periodically softly break the maximum weight limit.

TABLE 2. Summary of Methods Used to Solve the Provided Instance of the KnapsackProb

Problem Encoding	Reference	Solver
ILP	[70]	GLPK
	[69]	HiGHS
ToQUBO.jl	[71]	D-wave
QuadraticProgramToQUBO	[72]	D-wave (PyQubo)
		Fujitsu
Log	Sec. II-C	D-wave (PyQubo)
		Fujitsu
One-hot	Sec. II-C	D-wave (PyQubo)
		Fujitsu
Unbalanced	Ref. II-D, [65]	D-wave (PyQubo)
		Fujitsu

All methods found the optimal solution.

E. SUMMARY OF THE SIMULATIONS

Below, in Table 2, we provide a list of MILP and QUBO solvers, using the two aforementioned approaches, slack based and unbalanced.

F. SURVEY OF ALTERNATIVE APPROACHES

There exist various alternative approaches that exhibit varying degrees of divergence, in terms of algorithmic implementation, from the aforementioned methods. This section serves solely to provide a survey of some of these approaches without undertaking an experimental analysis.

Quantum Hybrid Frank–Wolfe Method: Recently, a hybrid-quantum generalization to the Frank–Wolfe method was proposed in [91]. This hybrid (Quantum) Frank–Wolfe (Q-FW) augmented Lagrangian method is suitable to tackle large QUBO instances due to a tight copositive relaxation of

the original QUBO formulation while dealing with the expensive hyperparameter tuning found in other QUBO heuristics. The Q-FW method first formulates constrained QUBOs as copositive programs, then employs the Frank–Wolfe method while satisfying linear (in)equality constraints. This is converted to a set of unconstrained QUBOs suitable to be run on, e.g., quantum annealers. It was found that Q-FW successfully satisfied linear equality and inequality constraints, in the context of QUBOs in computer vision applications, and Yurtsever et al. [91] solved intermediary QUBO problems on actual quantum devices demonstrating that Q-FW offers a promising alternative validity of Q-FW to "traditional" quantum QUBO solvers.

In a broader context, Q-FW seems to have the ability to address the costly hyperparameter tuning associated with other QUBO heuristics. By formulating constrained QUBOs as copositive programs, it adeptly handles linear equality and inequality constraints and transforms them into unconstrained QUBOs compatible with quantum annealers or other Ising machines. The general applicability and comparative efficiency of Q-FW against established quantum QUBO solvers is still unclear and subject to further research.

Grover Adapted Binary Optimization (GABO): Moving to fault-tolerant architectures, quadratic speed-up for combinatorial optimization problems is achievable with the GABO [92] when compared with brute force search. However, to achieve this, efficient oracles must be developed to represent problems and identify states that satisfy specific search criteria. Quantum arithmetic is commonly utilized to accomplish this task, but this approach can be expensive in terms of required Toffoli gates and ancilla qubits, which may pose a challenge in the near future. Interestingly, Gilliam et al. [92] provide such an oracle construction that makes GABO a promising approach for future quantum computers.

GABO might be able to offer a significant quantum advantage, a Grover-like quadratic speed-up for combinatorial optimization problems compared with the traditional brute force search. However, this speed-up can be only impactful in the realm of fault-tolerant quantum architectures, and thus not applicable to NISQ devices.

Graph Neural Networks (GNNs): Physics-inspired GNNs have been used in [93] (see [94] for a survey), where a physics-informed GNN-based scalable general-purpose QUBO solver is proposed. The approach therein is suitable for encoding any k-local Ising model, such as the $k = 2$ ColOpt problem discussed later. The GNN solver first drops the integrality constraints in order to obtain a differentiable relaxation f' of the original objective function f and, subsequently, proceeds to unsupervised learning on the node representations. The GNN is then trained to generate soft assignments, predicting the likelihood of each vertex in the graph belonging to one of the two distinct classes in conjunction with heuristics that aid in the consistency of the problem. Interestingly, the authors benchmark this approach in demanding problem instances of MAXCUT to find that it competes with the best in class SDP algorithms, such as the Goemans–Williamson algorithm [95].

The parallel processing capabilities of modern GPUs are well suited for GNN operations, making it feasible to handle large-scale graphs and complex optimization problems. Furthermore, the adaptability of GNNs, combined with unsupervised learning, allows for general-purpose solutions that can be applied across a variety of problem instances without the need for extensive retraining. This universality potentially saves computational resources in the long run. However, possible relaxations of integrality constraints for differentiability can sometimes lead to solutions that are not directly applicable or optimal for the original discrete problem. Another remark is that while GNNs can predict soft assignments efficiently, converting these into definitive solutions might lead to suboptimal results.

QUBO Continuous Relaxations With Light Sources: Finally, let us mention a recent heuristic quantum-inspired (relaxation) approach for solving QUBOs, as introduced in [96]. Concretely, the binary variables of the QUBO problem are represented by the relative phases of laser sources, transforming the discrete optimization problem into a continuous one. The lasers interact through a unique optical coupler, which uses programmable diffractive elements and additional optical components to control the interaction between all pairs of lasers, with a dynamic range of up to 8 bits. This design enables a fully connected network between all lasers, facilitating high-resolution pairwise interactions that are crucial in solving QUBO problems.

In [96], a benchmarking was performed for instances of the 3-regular 3-XORSAT problem and it was found that this method achieved significantly better time to solution (TTS) results as the instance size was increased. Despite this, seemingly incredible result, it is quite unclear whether this approach can perform on par for problems that do not have known polytime solutions (for generic instances of 3-XORSAT, there exists an algorithm with $\mathcal{O}(N^{2.736\ldots})$ complexity). In fact, it would be interesting to benchmark this approach against specialized SAT and MIP solvers.

Among its advantages, the method's transformation of discrete variables into continuous may offer new avenues for efficient problem solving. The fully connected network ensures precise pairwise interactions, a fundamental requirement for many combinatorial problems, also one that not all Ising machines can achieve. The aforementioned results of this method are obtained via an emulation architecture, rather than the physical machine which, in turn, might be able to perform even better. Another benefit of this tech is that it is readily available. However, there are a number of uncertainties. While the method showed promise on specific benchmarks, such as the 3-XORSAT problem, its general applicability and competitiveness against established solvers remain untested and benchmarking, e.g., within the MaxSAT, evaluation challenge would be highly enlightening. Furthermore, its performance on problems lacking known

polynomial-time solutions is yet to be determined, raising questions about its versatility.

Simulated Quantum Annealing (SQA): Obtaining quantum advantage for certain problem instances may be, in some cases, closer than one would anticipate by utilizing the technique of SQA [97]. Specifically, in [97], a method to rigorously demonstrate that the Markov chain underlying SQA effectively samples the target distribution and discovers the *global minimum* of the spike cost function in poly time supporting is developed. While the analysis is limited to a very specific model and cannot be considered conclusive, the authors use interesting techniques, such as initiating warm starts (a very popular technique applied to deep learning [98] as well as in QAOA [99]), from the adiabatic path and using the quantum ground state probability distribution to comprehend the stationary distribution of SQA.

Interestingly, SQA might be considered effective, as compared with a purely classical algorithm, for optimization problems with spike (deep and narrow) global optima. As such, exploiting hybrid solvers that combine SQA and classical algorithms could offer an alternative approach [100] due to SQA's effectiveness in sampling target distributions. However, to the best of our knowledge, its applicability is currently limited to specific models, and the broader effectiveness remains an open question.

III. COLLATERAL OPTIMIZATION

In Section II, our main objective was to identify the most suitable formulations of the KnapsackProb as a QUBO. However, for smaller scale problems, MILP solvers are generally expected to perform better than heuristic, hybrid, and near-term quantum solvers. Therefore, in this section, we first formulate the collateral optimization problem ColOpt as an MILP and, subsequently, reformulate it as a QUBO, similar to the approach taken in Section II.

The objective is to conduct several small-scale problems utilizing hybrid solvers with the purpose of algorithmic reformulation of the problem and its implementation. The ultimate goal is not to demonstrate the potential of hybrid or quantum alternatives for combinatorial optimization but rather to establish an automated approach to solving the problem once hardware capabilities become more advanced. In Table 4, we describe the most common financial terms used in the context of ColOpt, while Table 5 summarizes the mathematical notation that will be utilized in the following.

A. COLOPT MILP FORMULATION

In order to mitigate the risk of a borrower defaulting on a loan, it is necessary for them to furnish collateral in the form of stocks, bonds, cash, or other assets to offset any outstanding exposure. In the present scenario, we consider a financial institution that has a collection (or inventory) of assets, indicated by $\mathcal{I}$, which must be allocated among a set of accounts, indicated by $\mathcal{A}$. We shall use the indices i and j to refer to an asset and an account, respectively. The total number of assets and accounts is given by n and m, respectively. It should be noted that our primary focus in this section is on the algorithmic reformulation of the problem and its implementation, which could be automated once sufficiently powerful hardware becomes available. Consequently, we shall run a set of small-scale problems employing hybrid solver emulators, as shown in Section II. Our objective is not to demonstrate the superior performance of hybrid or quantum alternatives for combinatorial optimization but rather to heuristically identify the most appropriate formulation of the problem at hand.

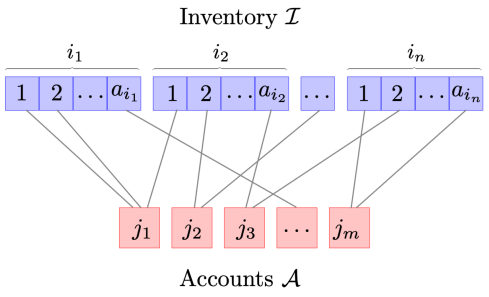

FIGURE 1. Schematic representation of the collateral optimization problem. The figure above can be seen as a bipartite graph where one has to make optimal allocation of nonunique and weighted pairings.

Interestingly, ColOpt can be formulated as a bipartite matching problem, as shown in Fig. 1. This bipartite graph is created with two sets of nodes: one set representing the inventory of assets $\mathcal{I}$, and the other set representing accounts $\mathcal{A}$. The edges between these nodes represent potential allocations of assets to accounts, with weights on these edges representing the suitability, cost, or value of the allocation, and the edges are multidirected and need to respect certain constraints. To model the constraints of ColOpt modifies the graph accordingly, as we will explain in the following text.

Limits: Delving deeper into the problem, consider each asset $i_k \in \mathcal{I}$, where $\mathcal{I} = i_1, i_2, \ldots, i_n$. Each asset i (momentarily simplifying the indices) is subdivided into a maximum quantity denoted by a_i. In more formal terms, there is a constraint on the maximum quantity of asset i that can be assigned. This is useful because, in the context of utilizing stocks as collateral, a financial institution may need or require the enforcement of an upper limit regarding the number of shares that can be allocated. The quantity of asset i_k can be converted into a corresponding dollar value by multiplying by the dimensionful term v_i, which is the market value (USD) per unit quantity.

Tiers: Every asset is linked to a tier, represented as $\omega_i \in [0, 1]$. This tier acts as a measure of the asset's quality, where distinct tiers correspond to various degrees of quality or attractiveness in the context of the ColOpt problem. The higher the value of ω_i, the higher the quality of asset i.

Exposure: When a financial institution borrows from one of its lenders, collateral must be posted to adequately cover capital that could be lost in the event of a default. This capital requirement is known as "exposure." For each account j, there is a required exposure (in USD) that must be met

IEEE Transactions on Quantum Engineering

indicated by c_j. Additionally, the duration of a transaction to a particular account can either be short term or long term. A binary variable $d_j \in \{0, 1\}$ is used to indicate the duration of account j. A value 1 is assigned to short-term and 0 for long-term transfers. To reduce the risk of losing posted collateral, it is required to minimize the use of high-quality assets for long-term transactions while maximizing their use for short-term transactions. We chose our decision variable to be a matrix $Q \in \mathbb{Q}_{\leq 1}^{n \times m}$, where the element Q_{ij} is the fractional amount of asset i that is allocated to account j. That is, the rows of Q correspond to the number of assets and the columns of the partition of each

$$Q = \begin{pmatrix} a_{11} & \dots & a_{1\,m} \\ \vdots & \ddots & \vdots \\ a_{1n} & \dots & a_{nm} \end{pmatrix}. \tag{12}$$

To illustrate an objective function that represents our goals considers the simple case of allocating a single asset i across two accounts, j and l, which have long- and short-term requirements, respectively. In this case, the objective function can be formulated as follows:

$$\min_{Q} \quad \omega_i Q_{ij} + (1 - \omega_i) Q_{il}. \tag{13}$$

In the expression above, the coefficient preceding short-term allocations is set to $1 - \omega_i$. This is so that we favor allocations of higher quality assets for trades with a short duration. To generalize this for all assets and accounts, we need a mechanism that updates these tiers according to the type of account they are posting collateral toward. This can be done by constructing a coefficients matrix Ω, where each element can be determined using

$$\Omega_{ij} = |\omega_i - d_j|. \tag{14}$$

Just as with the tiers, each element of Ω has a range of [0,1]. The objective function is then just the sum of elementwise multiplications between Ω and Q

$$\min_{Q} \quad \sum_{i=1}^{n} \sum_{j=1}^{m} \Omega_{ij} Q_{ij}. \tag{15}$$

To post collateral such that the financial institution meets the exposure for each account, we include a requirement constraint

$$\sum_{i=1}^{n} Q_{ij} a_i v_i H_{ij} \geq c_j \quad \forall j \in \mathcal{A}. \tag{16}$$

Here, v_i denotes the dollar market value for a single unit of quantity for asset i. Hence, the term on the left-hand side (LHS) represents the dollar value for the quantity of collateral that is chosen to be allocated, adjusted by a fractional factor H_{ij}, which is referred to as the haircut. Since markets are dynamic, the value of a posted collateral can diverge from its market value over time. In the case that the value drops below the required collateral value, the receiver is at risk. To avoid this, each account owner can evaluate the risk and place a haircut factor to reduce the value of an asset. The haircut is defined as the percentage difference between the market value and its value while used as collateral. For example, a haircut of 10% corresponds to $H_{ij} = 0.9$, meaning that the collateral value is 90% of the original market value.

We need to ensure that we do not allocate more collateral than we have available in inventory (i.e., we do not allocate more than 100% of the maximum available quantity). In financial terms, this prevents us from short selling the asset. This is done by including a consistency constraint

$$\sum_{j=1}^{m} Q_{ij} \leq 1 \quad \forall i \in \mathcal{I}. \tag{17}$$

There is also the trivial constraint to make sure that Q_{ij} does not take negative values

$$Q_{ij} \geq 0 \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{A}. \tag{18}$$

Further Constraints That Can Be Imposed: For example, there may be limits on the amount of a particular asset i to account j, given by B_{ij}. If $B_{ij} = 0$, the allocation is not eligible. This is a one-to-one constraint and has the following mathematical form:

$$Q_{ij} a_i \leq B_{ij} \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{A}. \tag{19}$$

Moreover, there are constraints that restrict the allocation of specific groups of assets to a single account, which exhibits a many-to-one relationship. For instance, certain types of assets, say $\{i_{X_1}, i_{X_2}, i_{X_3}\}$, may be subject to restrictions due to their interrelationships (for example, there exists a parent company X that posts these assets). To formalize this constraint, we introduce $\mathcal{G}$, which represents the set of all groups of assets. The binary variable T_{ig} is used to indicate whether asset i belongs to the group g, while K_{gj} represents the upper bound on the total amount of assets from group g that can be allocated to account j

$$\sum_{i=1}^{n} T_{ig} Q_{ij} a_i \leq K_{gj} \quad \forall g \in \mathcal{G}, \forall j \in \mathcal{A}. \tag{20}$$

The aim of imposing limits on the allocation of assets is to promote diversification and thereby reduce the risk borne by the receiver. In this article, we concentrate on allocating cash rather than equity and bonds, which allows us to avoid the constraint that $Q_{ij} a_i$ must take an integer value. Consequently, we can formulate the problem of collateral optimization as a continuous optimization problem without the need for additional constraints.

The Complete ColOpt Problem: Taking into account the information presented in the previous paragraphs, we can express the ColOpt problem using an MILP formulation

$$\min_{Q} \quad \sum_{i=1}^{n} \sum_{j=1}^{m} \Omega_{ij} Q_{ij} \qquad \forall i \in \mathcal{I}, \forall j \in \mathcal{A} \tag{21a}$$

$$\text{s.t.} \quad \sum_{j=1}^{m} Q_{ij} \leq 1 \qquad \forall i \in \mathcal{I} \tag{21b}$$

$$Q_{ij}a_i \leq B_{ij} \qquad \forall i \in \mathcal{I}, \forall j \in \mathcal{A} \quad (21c)$$

$$\sum_{i=1}^{n} T_{ig}Q_{ij}a_i \leq K_{gj} \qquad \forall g \in \mathcal{G}, \forall j \in \mathcal{A} \quad (21d)$$

$$\sum_{i=1}^{n} Q_{ij}a_i v_i H_{ij} \geq c_j \qquad \forall j \in \mathcal{A} \quad (21e)$$

$$Q_{ij} \geq 0 \qquad \forall i \in \mathcal{I}, \forall j \in \mathcal{A}. \quad (21f)$$

For clarity, we will summarize the constraints presented above. Constraint (21b) ensures that no asset is distributed to the accounts beyond unity. Constraint (21c) amounts to the limit constraints for each asset–account pairing. Constraint (21d) limits the quantity of particular groups of assets to certain accounts but will be ignored in what follows. Constraint (21e) is the requirement constraint that enforces that we allocate a suitable value such that the lender's loan is secured.

B. COLOPT QUBO FORMULATION

Binarization: To formulate the QUBO, we need to make a change of variables so that our decision variable is represented by binaries. This change imposes certain limitations on the allocation of assets, which are discussed in detail in the following text. We make use of an n-bit binary variable, similar to the methods used by the authors in [101] and [102]. To enable binary encoding of the decision variable Q, we can represent it as a matrix q containing binary elements. This transformation enables us to only allocate assets in a limited number of ways, as detailed in the following equation:

$$q = \begin{pmatrix} q_{11} & \cdots & q_{01} \\ \vdots & \ddots & \vdots \\ q_{n1} & \cdots & q_{nm} \end{pmatrix}. \quad (22)$$

Here, q_{ij} is an n-bit binary variable expressing the fractional allocation of asset i to account j

$$q_{ij} = x_{ij}^{b=1}, \ldots, x_{ij}^{b=B}, \quad x_{ij}^{b} \in \{0, 1\} \quad (23)$$

where B is the number of bits chosen.

Using $B = 4$ as an example, the largest number that can be represented by four bits is $1111_{\text{bin}} = 15_{\text{dec}}$. Thus, we split our allocation into 15 fractions, where if $q_{ij} = 0100_{\text{bin}} = 4_{\text{dec}}$, then 4/15 of asset i is allocated to account j. By increasing B, we increase the precision of our allocations.

By implementing a similar method to Braun et al. [103], we can discretize our fractional allocation by discretizing the interval $[Q_{ij}^{\min}, Q_{ij}^{\max}]$

$$Q_{ij}^{\max} - Q_{ij}^{\min} = \sum_{b=1}^{B} p_{ijb} = \sum_{b=1}^{B} \frac{2^{(b-1)}\left(Q_{ij}^{\max} - Q_{ij}^{\min}\right)}{M}. \quad (24)$$

M is the maximum value that can be represented by a binary string of length B ($M = 2^B - 1$). *Remark:* $Q_{ij}^{\max}$, $Q_{ij}^{\min}$ have been included here for generality, but in our problem, $Q_{ij}^{\max} = 1$ and $Q_{ij}^{\min} = 0$ by definition. Thus, $Q_{ij}^{\max} - Q_{ij}^{\min}$ is always 1.

The discretized amount of item i that is allocated to account j is then described by the dot product of the binary vector $x_{ijb} = (x_{ij1}, \ldots, x_{ijB})^T$ and $p_{ijb} = (p_{ij1}, \ldots, p_{ijB})^T$

$$q_{ij} = \sum_{b=1}^{B} p_{ijb}x_{ijb} \equiv p_{ijb}x_{ij}^{b} \quad (25)$$

where we use the Einstein summation convention (upper–lower repeated indices contract) for brevity. By pairing each bit in the bit string of q_{ij} with a coefficient p_{ijb} allows us to represent more values in the allowed range, which improves accuracy.

The cost function is then written as follows:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \Omega_{ij}p_{ijb}x_{ij}^{b}. \quad (26)$$

We then make the same replacement of $Q_{ij} \to p_{ijb}x_{ij}^{b}$ in the remaining constraints of the problem to construct the binarized collateral optimization problem. Again, a wide number of classical solvers (open source and commercial) are available that can find the global minimum of the problem in this form.

The total number of variables used to construct this binarized version of the problem is $\mathcal{O}(nmB)$. Since we have replaced the continuous variables with discrete binaries, the accuracy of the solution is expected to be reduced. This can be mitigated by increasing B; however, a compromise is needed between accuracy and resource usage. Regardless, the granularity for the fractional allocation is $1/M$.

Slack-Based Formulation: We hereby introduce a QUBO formulation for the collateral optimization problem, incorporating slack variables. The constraints outlined in (21b)–(21e) significantly influence the number of bits necessary to encode these slack variables, potentially leading to an extensive bit requirement. To address this issue, we employ the log-encoding method earlier in Section II-C.

For constraints expressed as "less-than-or-equal-to" inequalities, the number of bits required to encode the slack variable can be readily computed using $\lceil \log_2 u \rceil$, where u represents the upper bound of the constraint. Nonetheless, addressing the requirement constraint (21e) necessitates a more nuanced approach. As the objective is to minimize the excess value of the collateral posted, employing slack variables might prove inadequate since their purpose is to diminish the corresponding penalty term to 0 for any values satisfying this constraint. In contrast, within MILP frameworks, the solution to a minimization problem typically aligns closely with the lower bound of such a constraint. In light of this observation, we choose to alter the exposure requirement by transforming it into an equality constraint, thereby relaxing the original formulation

$$\sum_{i=1}^{n} Q_{ij}a_i v_i H_{ij} = c_j, \quad \forall j \in \mathcal{A} \quad (27)$$

and as a result, the associated penalty term requires no slack variables.

Hence, with the objective of minimizing the associated penalty terms originating from the aforementioned expression, the QUBO should yield a solution conforming to the boundaries of the exposure constraints. A limitation of this strategy is the intrinsic stochastic characteristic of annealing techniques and their propensity to become ensnared in local minima, potentially resulting in marginally exceeding or not quite meeting the mandatory exposure. Furthermore, because the upper bound for each consistency constraint is equal to one, we need to adjust their form so that we can calculate the number of slack bits required. We proceed by rearranging the binarized version of this constraint, which then attains a fractional form

$$\sum_{j=1}^{m} p_{ijb} x_{ij}^{b} = \sum_{j=1}^{m} \sum_{b=1}^{B} \frac{2^{b-1} x_{ij}^{b}}{M} \leq 1 \tag{28}$$

$\forall i \in \mathcal{I}$. By multiplying both sides by M, it is straightforward to realize that the highest value that can be represented by the bitstring is the one we use as our new upper bound, and this further allows us to easily determine the number of slack bits needed.

The penalty term for each of the n consistency constraints is written as follows:

$$\sum_{i=1}^{n} \left(\sum_{j=1}^{m} M(p_{ijb} x_{ij}^{b} - 1) + S_{\text{con}} \right)^2 \tag{29}$$

where S_{con} is the slack variable for each constraint, which is encoded by binary variables s_k via

$$S_{\text{con}} = \sum_{k=1}^{\lceil \log_2(M) \rceil} 2^{k-1} s_k. \tag{30}$$

Instead of introducing a penalty term for one-to-one constraints (21c), we can ensure that these are satisfied by reducing the number of bits representing each allocation so that these limits cannot be violated. The number of bits representing an allocation n_{ij} can be determined by

$$n_{ij} = \left\lfloor \log_2 \left(\frac{B_{ij}}{a_{ij}} M \right) \right\rfloor. \tag{31}$$

Note that, now in (25), the upper limit of the summation is no longer B but now n_{ij}, as this is the number of bits representing the allocation q_{ij}.

We choose to floor the result from the logarithmic function, as the alternative would still allow violations. However, a consequence of this is that the one-to-one constraints in the QUBO are more restrictive than their MILP counterpart.

Aside from these nuances above and the additional step of binarization, constructing the balanced formulation for this QUBO follows the same process described above in the discussion of the KnapsackProb instance. For the many-to-one constraints (21d), we introduce log-encoded slack variables $S_{K_{ij}}$. It is straightforward to derive the full QUBO objective as follows:

$$\begin{aligned} &\lambda_0 \sum_{i=1}^{n} \sum_{j=1}^{m} \Omega_{ij} p_{ijb} x_{ij}^{b} \\ &+ \lambda_1 \sum_{i=1}^{n} \left(\sum_{j=1}^{m} M(p_{ijb} x_{ij}^{b} - 1) + S_{\text{con}} \right)^2 \\ &+ \lambda_2 \sum_{j=1}^{m} \left(\sum_{i=1}^{n} p_{ijb} x_{ij}^{b} a_i v_i H_{ij} - c_j \right)^2 \\ &+ \lambda_3 \sum_{j=1}^{m} \sum_{g=1}^{G} \left(\sum_{i=1}^{n} p_{ijb} x_{ij}^{b} T_{ig} a_i - K_{gj} + S_{K_{gj}} \right)^2. \end{aligned} \tag{32}$$

Unbalanced Formulation: The previous constraints, set in (21b)–(21f), can be converted into penalty terms for the QUBO through unbalanced penalization, as we displayed with the KnapsackProb instance in Section II-D. For this approach, auxiliary functions are defined from the constraints, and Taylor approximations of appropriate exponentiations of these functions are used to derive the penalty terms.

For instance, consider the consistency constraint (21b). We move the upper bound to the LHS of the inequality and set this as our auxiliary function $h(x)$

$$h(x) = \sum_{j=1}^{m} p_{ijb} x_{ij}^{b} - 1 \leq 0. \tag{33}$$

Since this is a "less equal to zero" inequality, we use $e^{h(x)}$ to derive a penalty term that takes small values when this constraint is satisfied and large values when violated. As mentioned previously, QUBOs may only contain linear and quadratic terms so, therefore, we need to take a second-order Taylor approximation to obtain

$$\lambda_1 \left(\sum_{j=1}^{m} p_{ijb} x_{ij}^{b} - 1 \right) + \lambda_2 \left(\sum_{j=1}^{m} p_{ijb} x_{ij}^{b} - 1 \right)^2 \tag{34}$$

for all $i \in \mathcal{I}$. Essentially, the first of these terms favor solutions that satisfy the constraint while being as far away from the upper bound as possible. The second term, instead, favors solutions that are as close to this upper bound. Effective tuning of the parameters is, therefore, necessary to balance the effects of each term. Note, however, that in this case, we do not need to relax the exposure constraint to an equality as we can better manage how far beyond the solution is from these lower bounds. This equation is valid only for one asset; therefore, we can modify (34) to consider all assets

$$\lambda_1 \sum_{i=1}^{n} \left(\sum_{j=1}^{m} p_{ijb} x_{ij}^{b} - 1 \right) + \lambda_2 \sum_{i=1}^{n} \left(\sum_{j=1}^{m} p_{ijb} x_{ij}^{b} - 1 \right)^2. \tag{35}$$

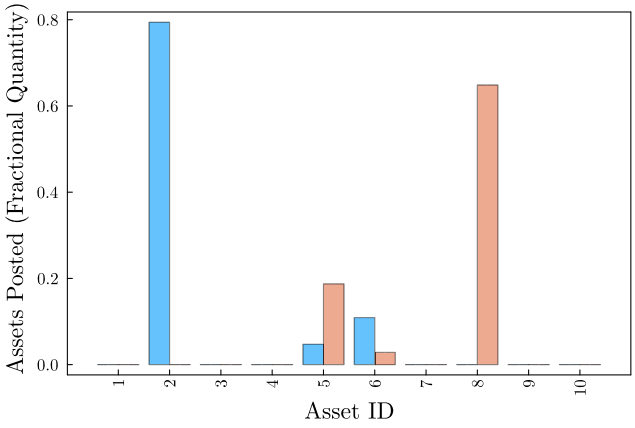

FIGURE 2. Optimal allocations of assets among accounts with short-term (red) and long-term (blue) requirements, determined through solving the ColOpt instance as a continuous LP with HiGHS. Assets IDs of 1–4 are low-tier, 5–6 are mid-tier, and the final 7–10 are the high-tier assets.

Following the same methods and the discussion on the one-to-one constraints used in the previous formulation, we can promote the remaining constraints to penalty terms in the QUBO.

The final QUBO can be written as follows:

$$\begin{aligned}
&\lambda_0 \sum_{i=1}^{n}\sum_{j=1}^{m} \Omega_{ij} p_{ijb} x_{ij}^b + \lambda_1 \sum_{i=1}^{n} \left(\sum_{j=1}^{m} p_{ijb} x_{ij}^b - 1 \right) \\
&+ \lambda_2 \sum_{i=1}^{n} \left(\sum_{j=1}^{m} p_{ijb} x_{ij}^b - 1 \right)^2 \\
&- \lambda_3 \sum_{j=1}^{m} \left(\sum_{i=1}^{n} p_{ijb} x_{ij}^b a_i v_i H_{ij} - c_j \right) \\
&+ \lambda_4 \sum_{j=1}^{m} \left(\sum_{i=1}^{n} p_{ijb} x_{ij}^b a_i v_i H_{ij} - c_j \right)^2 \\
&+ \lambda_5 \sum_{j=1}^{m}\sum_{g=1}^{G} \left(\sum_{i=1}^{n} p_{ijb} x_{ij}^b T_{ig} a_i - K_{gj} \right) \\
&+ \lambda_6 \sum_{j=1}^{m}\sum_{g=1}^{G} \left(\sum_{i=1}^{n} p_{ijb} x_{ij}^b T_{ig} a_i - K_{gj} \right)^2 \qquad (36)
\end{aligned}$$

IV. NUMERICAL ILLUSTRATIONS

Utilizing the formulations, as presented in Section III, we now define a small instance of the ColOpt problem based on a synthetic (however, realistic) small dataset. We perform our tests on an Apple MacBook Pro with M2 Max processor and 16 GB of memory. We remark that unlike the KnapsackProb that even with a small instance of a few items has a relatively simple structure with a single constraint, the small ColOpt problem instance we describe below has more complex constraints, and the interdependencies between the accounts and assets lead to a larger QUBO matrix and a much more challenging optimization landscape. In what follows, we utilize SA, which as a metaheuristic algorithm [104], is quite sensitive to the problem structure and its performance can vary significantly depending on the problem instance. The increased complexity of the collateral optimization problem may make it harder for the SA to explore the solution space effectively, leading to suboptimal results or longer convergence times. In such cases, it might be beneficial to fine-tune the parameters of SA to improve its performance on more complicated problem instances. An interesting approach toward that direction was followed in [31] which, for our context, we leave for future work.

Concretely, we have a portfolio of ten assets that have an approximate combined value of \$8.86M. These assets can be categorized by their tier rating, $\omega = \{0.2, 0.5, 0.8\}$, into low-, mid-, and high-tiered assets, respectively. Furthermore, the number of assets belonging to each category is chosen to be 4, 2, and 4, respectively. These assets are to be distributed in order to meet the requirements of five accounts. These requirements are distinguished by their duration, two of which are long term and have a combined exposure of $\sim$ \$1.49M. The remaining are short-term requirements with a total exposure of $\sim$ \$1.09M.

Due to restrictions, we slightly relax the problem by removing many-to-one constraints (21d). It is clear that, in the absence of these constraints, this instance of the ColOpt has a global optimum that can be easily obtained using classical strategies. As mentioned earlier, a compromise was needed between the precision of our results and the runtime performance, along with the limitation of the total number of bits that can be implemented in the solvers. To do this, we set the length of the bitstring representing each allocation, (23), to be 7. Hence, the granularity of our allocations is $1/127 \approx 0.0079\%$ (since $M = 127$). In other words, the lowest percentage of available asset quantity we may post to any one account is approximately 0.0079%. This becomes important if an asset quantity is significantly large, as the corresponding solution will allocate more than what is necessary to meet the requirements. Also, if very strict limits were considered, many violations could occur if the bitstring length chosen was inadequate. To this effect, we ensure that our sample contains sensible values for the quantity of each asset so that they can be distributed with enough precision to satisfy the exposure requirements efficiently. If no inequality constraints are included in the instance, then a total of 350 qubits are required for both formulations (10 assets, 5 accounts, 7). However, due to the way we introduce the constraints, as discussed in Section III, we can reduce this number to 228 qubits in the case of the unbalanced formulation and 298 qubits for the balanced form, as the consistency constraint in this case still requires slack variables.

Solving QUBO equations to obtain results that accurately reflect the goal of the objective function while simultaneously satisfying all constraints relies on the fine-tuning of the Lagrange multipliers. A potential solution is to make use of "hybrid solvers," such as D-wave's constrained quadratic

TABLE 3. Values Used for Tuning the Lagrangian Multipliers for Each Term in the QUBO for Differing Implementations

Formulation	Solver	Cost function	Consistency		Exposure		Objective value
		λ_0	λ_1	λ_2	λ_3	λ_4	
Balanced	D-wave's SA sampler	10^3	1	-	1	-	0.5898
Balanced	Fujitsu's digital annealer	10^5	1	-	300	-	0.7559
Unbalanced	D-wave's SA sampler	1.5×10^4	1	1	1	50	0.5244
Unbalanced	Fujitsu's digital annealer	2×10^4	1	1	1	50	0.5803
Continuous LP	HiGHS (Simplex)	-	-	-	-	-	0.4746

Objective value obtained for running this solution is also documented and, for comparison, the value obtained by a continuous solver is displayed.

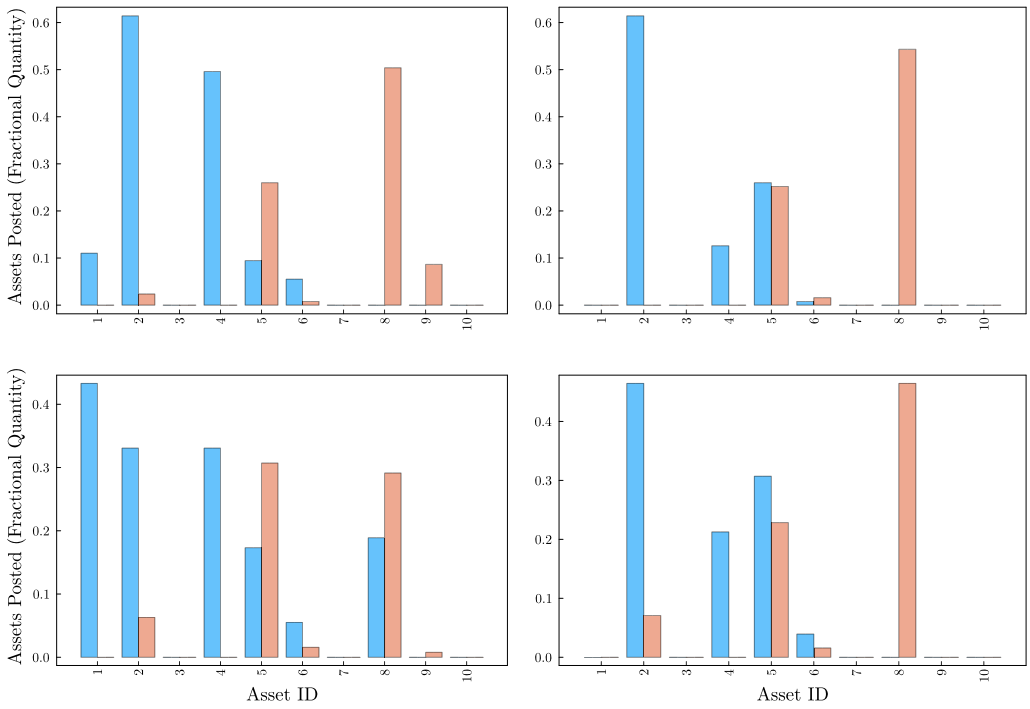

FIGURE 3. Allocation of different assets amongst accounts with (red) short-term and (blue) long-term requirements. Results are determined by (top) D-wave's simulated annealer and (bottom) Fujitsu's digital annealer with (left) balanced and (right) unbalanced formulations. The asset IDs of 1–4 are low-tier, 5–6 are mid-tier, and the final 7–10 are the high-tier assets.

model [105] that automatically calculates these, we instead use an intuitive approach. The consistency constraint (21b) is a hard constraint, as a solution that violates this constraint does not translate into a sensible business solution. Conversely, we treat the exposure requirement (21e) as a soft constraint and allow for small violations to some margin ϵ. Additionally, for both balanced and unbalanced formulations, it is important to note that there are significant differences between the magnitudes of the coefficients of each term in the QUBO. This can make fine-tuning of the penalty weights a difficult task. To manage this, we normalize each term in the QUBOs by dividing by their largest coefficient and then scale such that the lowest coefficient in each term has an order of magnitude of 1. We chose the weight for the cost function to be a magnitude larger than those of the constraints so that we achieve high-quality solutions. We then retrospectively increase the weight of the exposure and consistency terms to ensure that the system's constraints are satisfied.

Overall, our results were mixed in the sense that none of our runs managed to reach the global optimum nor to produce the globally optimal allocation with each run converging in a different local minimum. We estimate that one reason for this behavior is the limited number of runs performed that do not allow the annealing process to explore sufficient search space. This can easily be redeemed by increasing the number of runs (potentially decreasing the step size) and utilizing more compute powers. However, due to limited resources and compromising between computational runtime and the accuracy of the solution, we opted to choose a modest number of runs.

Table 3 displays the values chosen for each of the penalty weights and the resultant objective value that was outputted. Fig. 2 shows the global optimal solution solved using HiGHS. The asset allocations using neal and Fujitu's digital annealer are shown in Fig. 3. Additionally, Fig. 4 displays the percentage differences, for each of the solvers, between

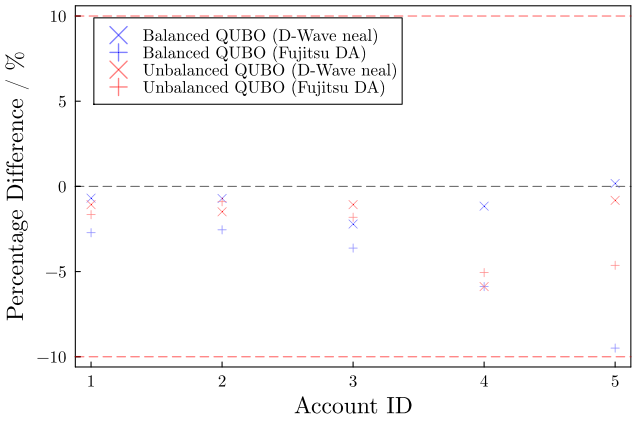

FIGURE 4. Percentage of the exposure requirements that have been met for each account. The dashed line represents the solution given by HiGHS solver that perfectly meets each requirement. We see greater deviations for the requirement of account 4, which is due to its outstanding exposure being an order of magnitude less than those of the other accounts; hence, it has a lower weighting in the QUBO.

the total values posted and the required exposures for all the accounts.

V. SUMMARY AND CONCLUSION

We surveyed the problem of collateral optimization from a business perspective and provided a business-realistic MILP formulation suitable to be mapped to a QUBO. In turn, we provided two QUBO formulations, one based on slack variables and one based on unbalanced penalization, and we implemented a small ColOpt problem instance on small emulations of hybrid solvers. We observed that the unbalanced penalization approach yields objective function values much closer to the global optimum, obtained using the simplex method, while the slack-based (balanced) approaches were further off. To this end, both approaches fail to find the global optimum even for such a relatively small, but nontrivial, problem instance. While we did not run our computations on specialized hardware (quantum annealer or digital annealer), to some extent, we indeed rediscover the expectation that heuristic approaches fail to find globally optimal solutions very often, including when implemented in quantum hardware [106]. The "take-home message" from the numerical illustrations clearly showcases the relevance of the formulation chosen for a given problem when one is interested in making it quantum ready.

Note, however, that the aim of this article is to show the relevance as well as the technical formulation of the collateral optimization problem for quantum or hybrid solutions rather than to perform a detailed benchmarking and, as a matter of fact, several improvements can be performed in order to obtain higher quality solutions (warm starting techniques, optimizing the annealing schedule, QUBO parameter optimization, utilization of GPU and tensor cores, and explore improved methods for SA [107]).

TABLE 4. Definition of Financial Terms Used

Term	Definition
Collateral	Any valuable asset used within lending agreements that can be seized by the lender from the borrower if they fail to repay the loan. This can consist of any asset deemed valuable by the lender. Acceptable types of assets are: equity (stocks), bonds and cash.
Market Value	The price that an asset could be sold for when auctioned in the marketplace.
Haircut	A reduction applied to the market value of an asset. Used to ensure that the posted asset will cover the cost of the outstanding exposure, given the risk that an assets value can fluctuate.
Exposure	The amount of capital that one stands to lose (risk) if an investment fails.

The question of whether MILPs, such as the collateral optimization problem, should be cast as QUBOs and approached by heuristic solvers (quantum or not) is tricky. To some extent, classical solvers, such as Gurobi, CPLEX, or even HiGHS, perform exceptionally well in many large instances. Improving thereof could be less or more beneficial depending on the situation or problem instance and its complexity. If a "quantum" solution is preferred, finding the ultimate strategy for this approach is a crucial problem (formulation of the problem, hyperparameter choice, solver choice, etc.).

The ColOpt problem, as presented above, can be extended in a variety of ways. The formulations provided above are in no way unique and further investigation might yield better results even for the local solvers. For example, recent research [108] has shown that an alternative formulation based on implicit penalization by restricting the Hamiltonian dynamics, in the context of parallelized QAOA, can be powerful and such an approach can be suitable for the collateral optimization problem as well. A different approach based on a stochastic quantum Monte Carlo algorithm, that mimics QA, was proposed [109] for such large-scale optimization problems, making it another suitable candidate for ColOpt. The benefit of this approach is that it can handle fully connected graphs that can be the case in certain ColOpt instances.

APPENDIX A
COLLATERAL OPTIMIZATION TERMINOLOGY

In Table 4, we collect a few of the common terms used in the context of ColOpt and, more generally, collateral management.

Table 5 collects all relevant ColOpt-related quantities/variables used in the main body of this article.

APPENDIX B
CONSTRAINT PENALIZATION IN QUBOS AND THE ISING MODEL

In the main body of this article, we have seen that in order to encode penalty terms into QUBOs, we need to use, for example, slack variables. Here, we summarize how we proceed to do so. In the KnapsackProb as well as the collateral

TABLE 5. List of All Symbols Used in the Formulation of the Collateral Optimization Problem

Symbol	Explanation
$\mathcal{I}$	Set of n inventory assets.
i	Index representing a particular asset.
a_i	Maximum available quantity of asset i.
v_i	Dollar value of a single unit of asset i.
ω_i	Tier value representing the quality of asset i.
$\mathcal{A}$	Set of m accounts.
j	Index representing a particular account.
c_j	Dollar value exposure of account j required.
d_j	Binary value indicating whether account obligation j is short or long term.
Ω_{ij}	Tier value of asset i depending on the duration of account j.
Q_{ij}	Decision variable indicating the percentage of asset i allocated to account j.
B_{ij}	Limit on individual allocation of asset i to account j.
H_{ij}	Haircut factor used to reduce the value of an asset.
K_{gj}	Limit on the allocation of assets in group g to account j.
T_{ig}	Binary value indicating whether asset i belongs to group g.

optimization problem, we have constraints of the form

$$Ax \leq c \tag{37}$$

$$Bx \geq d. \tag{38}$$

Let us consider the former constraint since the latter is essentially the same up to a factor of -1. It is common to view this constraint as $Ax - c \leq 0$ and then introduce a slack variable $s \in \mathbb{R}_{\geq 0}$ such that we define the penalty term $Ax - c + s = 0$. This is defined such that

$$\sup s = c - \inf Ax. \tag{39}$$

Then, as mentioned already in Section II, one needs to map a QUBO to the Ising Hamiltonian that performs QA, a restricted model of adiabatic quantum computation. Let us now briefly explain this connection.

The (classical) Ising model was first introduced as a mathematical model of ferromagnetism [110]. The variables take values in a discrete (binary) set $\sigma_i = \{\pm 1\}$ and are typically referred to as *spin* since, in the physical model, they describe the atomic spin of the particles. The model consists of a lattice Λ, with each lattice site, $j \in \Lambda$, having an assigned spin σ_j. The energy of a specific spin configuration is measured by the Hamiltonian of the system

$$H = -\sum_{j<k} J_{j,j+1}\sigma_j\sigma_{j+1} - \sum_j h_j\sigma_j + \varepsilon$$

where $J_{j,j+1}$ encodes the "nearest neighbor interaction" between adjacent sites, and h_j describes some external field that interacts with each site [32]. The overall minus sign is just a convention[3] and ε refers to a constant energy overhead.

In the quantum version of the Ising Hamiltonian, the sites are represented by a qubit. The spins are then simply given by the Pauli matrix σ_j^z acting on the jth site, with eigenvalues ± 1 when acting on the computational basis states $\{|0\rangle, |1\rangle\}$. In physics, the interesting problem is typically to find the ground state energy, or lowest energy eigenstate, of the Hamiltonian.

One thing we can immediately notice is that this model is quadratic in the spins and, thus, resembles the QUBO. By the simple change of variables $x_j = \frac{1-\sigma_j}{2}$, we directly see that the spin variables ± 1 are mapped to the binary variables $\{0, 1\}$. In the quantum version, we map the QUBO variables x_j to the operators $x_j = \mathbf{1} \otimes \cdots \otimes \frac{1}{2}(\mathbf{1} - \sigma_z) \otimes \mathbf{1} \otimes \cdots$, where the nontrivial operator acts on the jth site. This operator has eigenvalues 0 or 1 when acting on the states $|0\rangle$, respectively $|1\rangle$. Through this change of variables, the QUBO problem is, thus, equivalent to finding the ground state energy of the Ising Hamiltonian. When mapping between the QUBO and the Ising model, we might also need to account for the fact that we only minimize over the Ising model, while sometimes, for example in the KnapsackProb, we are seeking a maximum. This is of course easily accounted for by changing the relevant signs.

As an example, the KnapsackProb (8) has an Ising Hamiltonian with

$$\begin{aligned} J_{j,j+1} &= -\frac{\lambda_0}{4} w_j w_{j+1} \\ h_j &= -\left(\frac{v_j}{2} + \lambda_0 K w_j\right) \\ \varepsilon &= -\frac{\lambda_0}{4}\sum_i^n w_i^2 - \frac{1}{2}\sum_i^n v_i - \lambda_0 K^2 \end{aligned}$$

where we define $K = S - W + \frac{1}{2}\sum_{i=1}^n w_i$ for convenience.

ACKNOWLEDGMENT

The authors would like to thank J. Marecek, V. Kungurtsev, D. Snelling, R. Pikett, and P. Intallura for useful discussions and suggestions.

Disclaimer. This article was prepared for information purposes and is not a product of HSBC Bank Plc. or its affiliates. Neither HSBC Bank Plc. nor any of its affiliates make any explicit or implied representation or warranty and none of them accept any liability in connection with this article, including, but limited to, the completeness, accuracy, reliability of information contained herein, and the potential legal, compliance, tax, or accounting effects thereof.

REFERENCES

[1] M. Simmons, *Collateral Management: A Guide to Mitigating Counterparty Risk*. New York, NY, USA: Wiley, 2019. [Online]. Available: https://www.wiley.com/en-us/Collateral+Management%3A+A+Guide+to+Mitigating+Counterparty+Risk-p-9781119377122

[2] A. Nützenadel, "The financial crisis of 2008 experience, memory, history," *J. Mod. Eur. Hist.*, vol. 19, no. 1, pp. 3–7, Nov. 2020, doi: 10.1177/1611894420973590.

[3] "Basel III: Finalising post-crisis reforms," *SIAM Rev.*, 2017. [Online]. Available: https://www.bis.org/bcbs/publ/d424.htm

[3] However, the sign of J_{jk} does have a more interesting interpretation. If $J_{jk} > 0$, the model describes ferromagnetism, while if $J_{jk} < 0$, it describes antiferromagnetism.

[4] P. M. McBride, "The Dodd–Frank act and OTC derivatives: The impact of mandatory central clearing on the global OTC derivatives market," *Int. Lawyer*, vol. 44, no. 4, pp. 1077–1122, 2010, doi: jstor.org/stable/41806613.

[5] A. Delivorias, "European market infrastructure regulation—Regulation of OTC derivatives in the European Union," *Eur. Parliamentary Res. Serv.*, 2017. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32012R0648

[6] D. Brigo, "Counterparty risk FAQ: Credit VAR, PFE, CVA, DVA, closeout, netting, collateral, re-hypothecation, WWR, basel, funding, CCDS and margin lending," 2011, *arXiv:1111.1331*, doi: 10.48550/arXiv.1111.1331.

[7] Deloitte, "Collateral management—A survey of the current practices and trends in the banking industry," London, U.K., 2014. [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/fr/Documents/Pages/Services%20financiers/Deloitte_Collateral-managament-Survey-current-practices-and-trends_juill%20%20%20.pdf

[8] D. Bertsimas, *Introduction to Linear Optimization*. Belmont, MA, USA: Athena Scientific, Jan. 1997. [Online]. Available: http://athenasc.com/linoptbook.html

[9] IBM ILOG CPLEX, "V12. 1: User's manual for CPLEX," 2009. [Online]. Available: https://tomopt.com/docs/TOMLAB_CPLEX.pdf

[10] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, Beaverton, OR, USA, 2023.

[11] MOSEK ApS, "The MOSEK optimization toolbox for MATLAB manual," Version 9.0, 2019.

[12] J. Bylund, "Collateral optimization," Master's thesis, Dept. Phys., Umea Univ., Umea, Sweden, 2017.

[13] S. M. Mniszewski, "Graph partitioning as quadratic unconstrained binary optimization (QUBO) on spiking neuromorphic hardware," in *Proc. Int. Conf. Neuromorphic Syst.*, 2019, pp. 1–5, doi: 10.1145/3354265.3354269.

[14] C. D. Schuman et al., "Opportunities for neuromorphic computing algorithms and applications," *Nature Comput. Sci.*, vol. 2, no. 1, pp. 10–19, Jan. 2022, doi: 10.1038/s43588-021-00184-y.

[15] T. D. Ladd et al., "Quantum computers," *Nature*, vol. 464, no. 7285, pp. 45–53, 2010, doi: 10.1038/nature08812.

[16] D. J. Egger et al., "Quantum computing for finance: State of the art and future prospects," *IEEE Trans. Quantum Eng.*, vol. 1, Oct. 2020, Art. no. 3101724, doi: 10.1109/TQE.2020.3030314.

[17] D. Herman et al., "A survey of quantum computing for finance," 2022, *arXiv:2201.02773*, doi: 10.48550/arXiv.2201.02773.

[18] A. Bouland, W. van Dam, H. Joorati, I. Kerenidis, and A. Prakash, "Prospects and challenges of quantum finance," 2020, *arXiv:2011.06492*, doi: 10.48550/arXiv.2011.06492.

[19] P. Rebentrost, A. Luongo, S. Bosch, and S. Lloyd, "Quantum computational finance: Martingale asset pricing for incomplete markets," 2022, *arXiv:2209.08867*, doi: 10.48550/arXiv.2209.08867.

[20] P. Intallura, G. Korpas, S. Chakraborty, V. Kungurtsev, and J. Marecek, "A survey of quantum alternatives to randomized algorithms: Monte Carlo integration and beyond," 2023, *arXiv:2303.04945*, doi: 10.48550/arXiv.2303.04945.

[21] M. Pistoia et al., "Quantum machine learning for finance ICCAD special session paper," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Des.*, 2021, pp. 1–9, doi: 10.1109/ICCAD51958.2021.9643469.

[22] A. Jacquier, O. Kondratyev, A. Lipton, and M. L. de Prado, *Quantum Machine Learning and Optimisation in Finance: On the Road to Quantum Advantage*. Birmingham, U.K.: Packt Publishing, 2022. [Online]. Available: https://www.packtpub.com/product/quantum-machine-learning-and-optimisation-in-finance/9781801813570

[23] D. Emmanoulopoulos and S. Dimoska, "Quantum machine learning in finance: Time series forecasting," 2022, *arXiv.2202.00599*, doi: 10.48550/arXiv.2202.00599.

[24] J. Kirk et al., "Emergent order in classical data representations on Ising spin models," 2023, *arXiv:2303.01461*, doi: 10.48550/arXiv.2303.01461.

[25] G. Rosenberg, P. Haghnegahdar, P. Goddard, P. Carr, K. Wu, and M. L. Prado, "Solving the optimal trading trajectory problem using a quantum annealer," in *Proc. 8th Workshop High Perform. Comput. Finance*, 2015, pp. 1–7, doi: 10.1145/2830556.2830563.

[26] P. Rebentrost and S. Lloyd, "Quantum computational finance: Quantum algorithm for portfolio optimization," 2018, *arXiv:1811.03975*, doi: 10.48550/arXiv.1811.03975.

[27] L. Leclerc et al., "Financial risk management on a neutral atom quantum processor," 2022, *arXiv:2212.03223*, doi: 10.48550/arXiv.2212.03223.

[28] D. Lim and P. Rebentrost, "A quantum online portfolio optimization algorithm," 2022, *arXiv:2208.14749*, doi: 10.48550/arXiv.2208.14749.

[29] S. Brandhofer et al., "Benchmarking the performance of portfolio optimization with QAOA," *Quantum Inf. Process.*, vol. 22, no. 1, pp. 1–27, 2023, doi: 10.1007/s11128-022-03766-5.

[30] M. Vesely, "Finding the optimal currency composition of foreign exchange reserves with a quantum computer," 2023, *arXiv:2303.01909*, doi: 10.48550/arXiv.2303.01909.

[31] W. Sakuler, J. M. Oberreuter, R. Aiolfi, L. Asproni, B. Roman, and J. Schiefer, "A real world test of portfolio optimization with quantum annealing," 2023, *arXiv:2303.12601*, doi: 10.48550/arXiv.2303.12601.

[32] A. Lucas, "Ising formulations of many NP problems," *Front. Phys.*, vol. 2, 2014, Art. no. 5, doi: 10.3389/fphy.2014.00005.

[33] M. Cerezo et al., "Variational quantum algorithms," *Nature Rev. Phys.*, vol. 3, no. 9, pp. 625–644, 2021, doi: 10.1038/s42254-021-00348-9.

[34] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," 2014, *arXiv:1411.4028*, doi: 10.48550/arXiv.1411.4028.

[35] Y. Chatterjee, E. Bourreau, and M. J. Rančić, "Solving various NP-hard problems using exponentially fewer qubits on a quantum computer," 2023, *arXiv:2301.06978*, doi: 10.48550/arXiv.2301.06978.

[36] B. Apolloni, C. Carvalho, and D. de Falco, "Quantum stochastic optimization," *Stochastic Processes Appl.*, vol. 33, no. 2, pp. 233–244, Dec. 1989, doi: 10.1016/0304-4149(89)90040-9.

[37] A. B. Finnila, M. A. Gomez, C. Sebenik, C. Stenson, and J. D. Doll, "Quantum annealing: A new method for minimizing multidimensional functions," *Chem. Phys. Lett.*, vol. 219, no. 5/6, pp. 343–348, Mar. 1994, doi: 10.1016/0009-2614(94)00117-0.

[38] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, "A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem," *Science*, vol. 292, no. 5516, pp. 472–475, Apr. 2001, doi: 10.1126/science.1057726.

[39] T. Albash and D. A. Lidar, "Adiabatic quantum computation," *Rev. Mod. Phys.*, vol. 90, Jan. 2018, Art. no. 015002, doi: 10.1103/RevModPhys.90.015002.

[40] P. Hauke, H. G. Katzgraber, W. Lechner, H. Nishimori, and W. D. Oliver, "Perspectives of quantum annealing: Methods and implementations," *Rep. Prog. Phys.*, vol. 83, no. 5, 2020, Art. no. 054401, doi: 10.1088/1361-6633/ab85b8.

[41] D. Willsch et al., "Benchmarking advantage and D-wave 2000q quantum annealers with exact cover problems," *Quantum Inf. Process.*, vol. 21, no. 4, 2022, Art. no. 141, doi: 10.1007/s11128-022-03476-y.

[42] F. Phillipson and H. S. Bhatia, "Portfolio optimisation using the D-wave quantum annealer," in *Proc. 21st Int. Conf. Comput. Sci.*, Krakow, Poland, 2021, pp. 45–59, doi: 10.48550/arXiv.2012.01121.

[43] P. M. J. van Laarhoven and E. H. L. Aarts, "Simulated annealing: Theory and applications," in *Mathematics and Its Applications*. Dordrecht, The Netherlands: Kluwer Academic, Jun. 1987, doi: 10.1007/978-94-015-7744-1_2.

[44] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. G. Katzgraber, "Physics-inspired optimization for quadratic unconstrained problems using a digital annealer," *Front. Phys.*, vol. 7, 2019, Art. no. 48, doi: 10.3389/fphy.2019.00048.

[45] S. Tsukamoto, M. Takatsu, S. Matsubara, and H. Tamura, "An accelerator architecture for combinatorial optimization problems," *FUJITSU Sci. Tech. J.*, vol. 53, no. 5, pp. 8–13, Sep. 2017. [Online]. Available: https://www.fujitsu.com/global/about/resources/publications/fstj/archives/vol53-5.html

[46] Z. Zhou, Y. Du, X. Tian, and D. Tao, "QAOA-in-QAOA: Solving large-scale maxcut problems on small quantum machines," *Phys. Rev. Appl.*, vol. 19, Feb. 2023, Art. no. 024027, doi: 10.1103/PhysRevApplied.19.024027.

[47] T. Lubinski, C. Coffrin, C. McGeoch, P. Sathe, J. Apanavicius, and D. E. B. Neira, "Optimization applications as quantum performance benchmarks," 2023, *arXiv:2302.02278*, doi: 10.48550/arXiv.2302.02278.

[48] S. Chen, J. Cotler, H.-Y. Huang, and J. Li, "The complexity of NISQ," 2022, *arXiv:2210.07234*, doi: 10.48550/arXiv.2210.07234.

[49] L. Pusey-Nazzaro et al., "Adiabatic quantum optimization fails to solve the knapsack problem," 2020, *arXiv:2008.07456*, doi: 10.48550/arXiv.2008.07456.

[50] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. New York, NY, USA: Dover, Jul. 1998. [Online]. Available: https://store.doverpublications.com/0486402584.html

[51] L. Huang et al., "Branch and bound in mixed integer linear programming problems: A survey of techniques and trends," 2021, *arXiv:2111.06257*, doi: 10.48550/arXiv.2111.06257.

[52] R. Herrman, L. Treffert, J. Ostrowski, P. C. Lotshaw, T. S. Humble, and G. Siopsis, "Globally optimizing QAOA circuit depth for constrained optimization problems," *Algorithms*, vol. 14, no. 10, 2021, Art. no. 294, doi: 10.3390/a14100294.

[53] D. Amaro, C. Modica, M. Rosenkranz, M. Fiorentini, M. Benedetti, and M. Lubasch, "Filtering variational quantum algorithms for combinatorial optimization," *Quantum Sci. Technol.*, vol. 7, no. 1, Jan. 2022, Art. no. 015021, doi: 10.1088/2058-9565/ac3e54.

[54] G. Nannicini, "Performance of hybrid quantum-classical variational heuristics for combinatorial optimization," *Phys. Rev. E*, vol. 99, Jan. 2019, Art. no. 013304, doi: 10.1103/PhysRevE.99.013304.

[55] L. Bittel and M. Kliesch, "Training variational quantum algorithms is NP-hard," *Phys. Rev. Lett.*, vol. 127, no. 12, 2021, Art. no. 120502, doi: 10.1103/PhysRevLett.127.120502.

[56] J. Rivera-Dean, P. Huembeli, A. Acín, and J. Bowles, "Avoiding local minima in variational quantum algorithms with neural networks," 2021, *arXiv:2104.02955*, doi: 10.48550/arXiv.2104.02955.

[57] V. Kungurtsev, G. Korpas, J. Marecek, and E. Y. Zhu, "Iteration complexity of variational quantum algorithms," 2022, *arXiv:2209.10615*, doi: 10.48550/arXiv.2209.10615.

[58] G. De Palma, M. Marvian, C. Rouzé, and D. S. Françça, "Limitations of variational quantum algorithms: A quantum optimal transport approach," *PRX Quantum*, vol. 4, Jan. 2023, Art. no. 010309, doi: 10.1103/PRXQuantum.4.010309.

[59] S. Ebadi et al., "Quantum optimization of maximum independent set using Rydberg atom arrays," *Science*, vol. 376, no. 6598, pp. 1209–1215, 2022, doi: 10.1126/science.abo6587.

[60] B. Tasseff et al., "On the emerging potential of quantum annealing hardware for combinatorial optimization," 2022, *arXiv:2210.04291*, doi: 10.48550/arXiv.2210.04291.

[61] A. D. King et al., "Quantum critical dynamics in a 5,000-qubit programmable spin glass," *Nature*, vol. 617, pp. 61–66, Apr. 2023, doi: 10.1038/s41586-023-05867-2.

[62] M. Jünger et al., "Quantum annealing versus digital computing," *ACM J. Exp. Algorithmics*, vol. 26, pp. 1–30, Jul. 2021, doi: 10.1145/3459606.

[63] F. Glover, G. Kochenberger, and Y. Du, "A tutorial on formulating and using QUBO models," 2018, *arXiv:1811.11538*, doi: 10.48550/arXiv.1811.11538.

[64] A. Montanez-Barrera, A. Maldonado-Romo, D. Willsch, and K. Michielsen, "Unbalanced penalization: A new approach to encode inequality constraints of combinatorial problems for quantum optimization algorithms," 2022, *arXiv:2211.13914*, doi: 10.48550/arXiv.2211.13914.

[65] K. Axiotis and C. Tzamos, "Capacitated dynamic programming: Faster knapsack and graph algorithms," 2018, *arXiv:1802.06440*, doi: 10.48550/arXiv.1802.06440.

[66] M. H. Bateni, M. T. Hajiaghayi, S. Seddighin, and C. Stein, "Fast algorithms for knapsack via convolution and prediction," in *Proc. 50th Annu. ACM SIGACT Symp. Theory Comput.*, 2018, pp. 1269–1282, doi: 10.48550/arXiv.1811.12554.

[67] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. Hoboken, NJ, USA: Wiley, 1990.

[68] Q. Huangfu and J. A. J. Hall, "Parallelizing the dual revised simplex method," *Math. Program. Comput.*, vol. 10, no. 1, pp. 119–142, Dec. 2017, doi: 10.1007/s12532-017-0130-5.

[69] GNU linear programming kit. [Online]. Available: https://www.gnu.org/software/glpk/

[70] P. Xavier, T. Andrade, J. Garcia, and D. Bernal, ToQUBO.jl, Mar. 2022. [Online]. Available: https://docs.juliahub.com/ToQUBO/W6hpN/0.1.0/

[71] Qiskit Optimization Development Team, Quadraticprogramtoqubo. [Online]. Available: https://qiskit.org/ecosystem/optimization/stubs/qiskit_optimization.converters.QuadraticProgramToQubo.html

[72] M. Zaman, K. Tanahashi, and S. Tanaka, "PYQUBO: Python library for mapping combinatorial optimization problems to QUBO form," *IEEE Trans. Comput.*, vol. 71, no. 4, pp. 838–850, Apr. 2022, doi: 10.1109/TC.2021.3063618.

[73] A. Glos, A. Kundu, and Özlem Salehi, "Optimizing the production of test vehicles using hybrid constrained quantum annealing," *SN Comput. Sci.*, vol. 4, 2023, Art. no. 609, doi: 10.1007/s42979-023-02071-x.

[74] S. Palmer, S. Sahin, R. Hernandez, S. Mugel, and R. Orus, "Quantum portfolio optimization with investment bands and target volatility," 2021, *arXiv:2106.06735*, doi: 10.48550/arXiv.2106.06735.

[75] L. Braine, D. J. Egger, J. Glick, and S. Woerner, "Quantum algorithms for mixed binary optimization applied to transaction settlement," *IEEE Trans. Quantum Eng.*, vol. 2, 2021, Art. no. 3101208, doi: 10.1109/TQE.2021.3063635.

[76] D. Venturelli, D. J. J. Marchand, and G. Rojo, "Quantum annealing implementation of job-shop scheduling," 2015, *arXiv:1506.08479*, doi: 10.48550/arXiv.1506.08479.

[77] J. Zhang, G. L. Bianco, and J. C. Beck, "Solving job-shop scheduling problems with QUBO-based specialized hardware," in *Proc. Int. Conf. Automated Plan. Scheduling*, 2022, vol. 32, pp. 404–412, doi: 10.1609/icaps.v32i1.19826.

[78] D. Amaro, M. Rosenkranz, N. Fitzpatrick, K. Hirano, and M. Fiorentini, "A case study of variational quantum algorithms for a job shop scheduling problem," *EPJ Quantum Technol.*, vol. 9, no. 1, Feb. 2022, Art. no. 5, doi: 10.1140/epjqt/s40507-022-00123-4.

[79] K. Domino, M. Koniorczyk, K. Krawiec, K. Jałowiecki, S. Deffner, and B. Gardas, "Quantum annealing in the NISQ era: Railway conflict management," *Entropy*, vol. 25, no. 2, 2023, Art. no. 191, doi: 10.3390/e25020191.

[80] Y. Matsumoto and S. Nakamura, "Qualign: Solving sequence alignment based on quadratic unconstrained binary optimization," *SSRN Electron. J.*, to be published, doi: 10.2139/ssrn.4037935.

[81] A. Luckow, J. Klepsch, and J. Pichlmeier, "Quantum computing: Towards industry reference problems," *Digitale Welt*, vol. 5, pp. 38–45, 2021, doi: 10.1007/s42354-021-0335-7.

[82] D. Ratke, "List of QUBO formulations," blog.xa0.de, 2021. [Online]. Available: https://blog.xa0.de/post/List-of-QUBO-formulations/

[83] G. Evenbly and G. Vidal, "Tensor network states and geometry," *J. Stat. Phys.*, vol. 145, pp. 891–918, 2011, doi: 10.1007/s10955-011-0237-4.

[84] J. C. Bridgeman and C. T. Chubb, "Hand-waving and interpretive dance: An introductory course on tensor networks," *J. Phys. A, Math. Theor.*, vol. 50, no. 22, 2017, Art. no. 223001, doi: 10.1088/1751-8121/aa6dc3.

[85] D. A. Zheltkov and A. Osinsky, "Global optimization algorithms using tensor trains," in *Large-Scale Scientific Computing*. Berlin, Germany: Springer, 2020, pp. 197–202, doi: 10.1007/978-3-030-41032-2_22.

[86] A. Nikitin, A. Chertkov, R. Ballester-Ripoll, I. Oseledets, and E. Frolov, "Are quantum computers practical yet? A case for feature selection in recommender systems using tensor networks," 2022, *arXiv:2205.04490*, doi: 10.48550/arXiv.2205.04490.

[87] B. C. B. Symons, D. Galvin, E. Sahin, V. Alexandrov, and S. Mensa, "A practitioner's guide to quantum algorithms for optimisation problems," 2023, *arXiv: 2305.07323*, doi: 10.48550/arXiv.2305.07323.

[88] R. A. Quintero and L. F. Zuluaga, "Characterizing and benchmarking QUBO reformulations of the knapsack problem," Depart. of Ind. and Syst. Eng., Lehigh Univ., Bethlehem, PA, USA, ISE Tech. Rep. 21T-028, 2021.

[89] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Amsterdam, The Netherlands: Elsevier, 1982. [Online]. Available: https://www.mit.edu/~dimitrib/lagr_mult.html

[90] I. Dunning, J. Huchette, and M. Lubin, "Jump: A modeling language for mathematical optimization," *SIAM Rev.*, vol. 59, no. 2, pp. 295–320, 2017, doi: 10.1137/15M1020575.

[91] A. Yurtsever, T. Birdal, and V. Golyanik, "Q-FW: A hybrid classical-quantum Frank-Wolfe for quadratic binary optimization," in *Proc. 17th Eur. Conf. Comput. Vis.*, Tel Aviv, Israel, 2022, pp. 352–369, doi: 10.48550/arXiv.2203.12633.

[92] A. Gilliam, S. Woerner, and C. Gonciulea, "Grover adaptive search for constrained polynomial binary optimization," *Quantum*, vol. 5, 2021, Art. no. 428, doi: 10.48550/arXiv.1912.04088.

[93] M. J. A. Schuetz, J. K. Brubaker, and H. G. Katzgraber, "Combinatorial optimization with physics-inspired graph neural networks," *Nat. Mach. Intell.*, vol. 4, pp. 367–377, 2022, doi: 10.1038/s42256-022-00468-6.

[94] P. Veličković, "Everything is connected: Graph neural networks," *Curr. Opin. Struct. Biol.*, vol. 79, 2023, Art. no. 102538, doi: 10.1016/j.sbi.2023.102538.

[95] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," in *J. ACM*, vol. 42, no. 6, pp. 1115–1145, Nov. 1995, doi: 10.1145/227683.227684.

[96] I. Meirzada et al., "Lightsolver—A new quantum-inspired solver cracks the 3-Regular 3-XORSAT challenge," 2022, *arXiv:2207.09517*, doi: 10.48550/arXiv.2207.09517.

[97] E. Crosson and A. W. Harrow, "Simulated quantum annealing can be exponentially faster than classical simulated annealing," in *Proc. IEEE 57th Annu. Symp. Found. Comput. Sci.*, 2016, pp. 714–723, doi: 10.1109/FOCS.2016.81.

[98] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," 2016, *arXiv:1608.03983*, doi: 10.48550/arXiv.1608.03983.

[99] D. J. Egger, J. Mareček, and S. Woerner, "Warm-starting quantum optimization," in *Quantum*, vol. 5, 2021, Art. no. 479, doi: 10.22331/q-2021-06-17-479.

[100] D. Volpe, G. A. Cirillo, M. Zamboni, and G. Turvani, "Integration of simulated quantum annealing in parallel tempering and population annealing for heterogeneous-profile QUBO exploration," *IEEE Access*, vol. 11, pp. 30390–30441, 2023, doi: 10.1109/ACCESS.2023.3260765.

[101] J. Lang, S. Zielinski, and S. Feld, "Strategic portfolio optimization using simulated, digital, and quantum annealing," *Appl. Sci.*, vol. 12, no. 23, 2022, Art. no. 12288, doi: 10.3390/app122312288.

[102] D. Ottaviani and A. Amendola, "Low rank non-negative matrix factorization with D-wave 2000q," 2018, *arXiv:1808.08721*, doi: 10.48550/arXiv.1808.08721.

[103] M. C. Braun, T. Decker, N. Hegemann, S. F. Kerstan, and F. Lorenz, "Towards optimization under uncertainty for fundamental models in energy markets using quantum computers," 2023, *arXiv:2301.01108*, doi: 10.48550/arXiv.2301.01108.

[104] J. Marecek, "Handbook of approximation algorithms and metaheuristics," *Comput. J.*, vol. 53, no. 8, pp. 1338–1339, 2010, doi: 10.1093/comjnl/bxp121.

[105] D-Wave, Constrained quadratic models. [Online]. Available: https://docs.ocean.dwavesys.com/en/stable/concepts/cqm.html

[106] D. Vert, R. Sirdey, and S. Louise, "Benchmarking quantum annealing against 'hard' instances of the bipartite matching problem," *SN Comput. Sci.*, vol. 2, 2021, Art. no. 106, doi: 10.1007/s42979-021-00483-1.

[107] H. Nakano and K. Fujiyoshi, "Improved method of simulated annealing for unreachable solution space," *Space*, vol. 1, no. 2, 2016.

[108] K. Ender, A. Messinger, M. Fellner, C. Dlaska, and W. Lechner, "Modular parity quantum approximate optimization," *PRX Quantum*, vol. 3, Jul. 2022, Art. no. 030304, doi: 10.1103/PRXQuantum.3.030304.

[109] N. Onizawa, R. Sasaki, D. Shin, W. J. Gross, and T. Hanyu, "Stochastic quantum Monte Carlo algorithm for large-scale combinatorial optimization problems," 2023, *arXiv:2302.12454*, doi: 10.48550/arXiv.2302.12454.

[110] B. A. Cipra, "An introduction to the Ising model," *Amer. Math. Month.*, vol. 94, no. 10, pp. 937–959, 1987, doi: 10.1080/00029890.1987.12000742.