Mixed-integer linear programming solver using Benders decomposition assisted by a neutral-atom quantum processor

M. Yassine Naghmouchi* and Wesley da Silva Coelho†
Pasqal SAS, 7 Rue Léonard de Vinci, 91300 Massy, France

(Received 9 February 2024; accepted 14 June 2024; published 12 July 2024)

This paper presents a hybrid classical-quantum approach to solve mixed-integer linear programming (MILP) using neutral-atom quantum computations. We apply Benders decomposition (BD) to segment MILPs into a master problem (MP) and a subproblem, where the MP is addressed using a neutral-atom device, after being transformed into a quadratic unconstrained binary optimization (QUBO) model, with an automatized procedure. Our MILP to QUBO conversion tightens the upper bounds of the continuous variables involved, positively impacting the required qubit count, and the convergence of the algorithm. To solve the QUBO, we develop a heuristic for atom register embedding and apply a variational algorithm for pulse shaping. In addition, we implement a proof of concept that outperforms existing solutions. We also conduct preliminary numerical results: In a series of small MILP instances our algorithm identifies over 95% of feasible solutions of high quality, outperforming classical BD approaches where the MP is solved using simulated annealing.

DOI: 10.1103/PhysRevA.110.012434

I. INTRODUCTION

Combinatorial optimization problems are crucial in industries such as logistics, planning, telecommunications, and resource management [1]. They offer significant economic and strategic benefits. However, as these problems increase in size, involving more variables and constraints, they become computationally challenging. Consequently, finding high-quality solutions quickly becomes a difficult task. To address these challenges, there is ongoing development in advanced classical optimization techniques. In particular, mixed-integer linear programming (MILP) [2] plays a crucial role in solving a wide range of optimization problems. It integrates integer and continuous variables, which adds computational complexity compared to pure integer linear programming (ILP) [3]. For instance, tasks such as solution space tightening using cutting planes, i.e., linear inequalities added to eliminate infeasible solutions, become more difficult with MILPs [4]. Benders decomposition (BD) [5] is an efficient method for solving MILPs. The approach stands out for its applicability to a wide range of MILPs, unlike other structure-dependent methods such as Dantzig-Wolfe decomposition [6]. It separates integer variables in a process called restriction [7]. This process splits the MILP into a master problem (MP) and a linear program (LP) subproblem (SP). While the SP is manageable on classical computers, the MP, containing discrete variables, constitutes a computational bottleneck. This work addresses this specific bottleneck using a neutral-atom quantum processor in a hybrid classical-quantum framework.

In recent years, hybrid classical-quantum approaches have started to gain traction in addressing NP-hard problems [8–12]. The approach assigns the computationally hard part, such as an ILP, to a quantum processing unit (QPU). Conversely, classical central processing units (CPUs) handle less computationally demanding parts like LPs. In MILP solving with BD, hybrid methods show promising results over classical methods [13–16]. Generally, these methods use a QPU to solve the MP after transforming it to a quadratic unconstrained binary optimization (QUBO) model. In neutral-atom quantum computing, atoms, controlled by lasers, serve as qubits and are versatile enough to encode any QUBO [17]. Known for its scalability and precision, enabled by optical tweezers, this method distinguishes itself among quantum technologies such as Josephson junctions [18], trapped ions [19], and photons [20]. In this context, the Hamiltonian governing qubit dynamics can be tailored to a QUBO model in such a way that the Hamiltonian's ground state encodes the optimal solution to the QUBO [21]. This elegant alignment between physics and algorithms enables us to tackle a vast variety of optimization problems. The algorithm design for neutral-atom systems includes register embedding and pulse shaping. Register embedding spatially arranges qubits to mirror the QUBO matrix, which serves as the problem's encoding method. Pulse shaping, in contrast, adjusts laser pulses to manipulate qubit states and cover strategies like the variational algorithms [9] and quantum approximate optimization (inspired) algorithms (QAOAs) [8]. While the problem encoding is based on register embedding, pulse shaping directs the algorithm toward finding a solution, with each component contributing to the algorithm's overall structure.

In this study we propose a hybrid classical-quantum BD framework. As presented in Fig. 1, our approach begins by splitting the MILP into a MP and a SP. The MP is reformulated into a QUBO for quantum processing. Based on an iterative BD algorithm, the MP is solved using a QPU and the SP on a CPU. Register embedding and pulse shaping algorithms are

*Contact author: yassine.naghmouchi@pasqal.com
†Contact author: wesley.coelho@pasqal.com

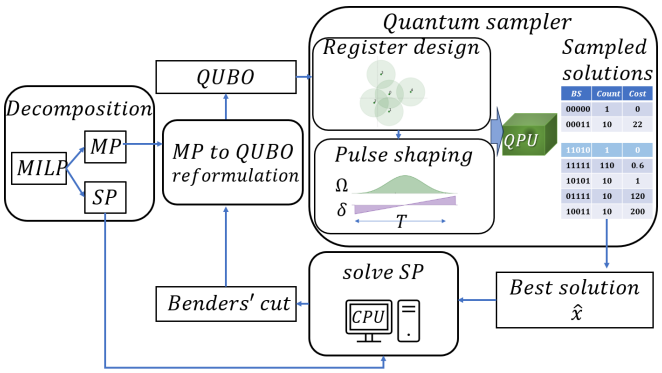

FIG. 1. Approach overview. Shown is a representation of our hybrid framework that merges classical computing with neutral-atom quantum processing for solving MILP problems. The MILP is divided into a MP and a SP. The MP is reformulated into a QUBO, which is then processed by a quantum sampler. Register embedding is applied to configure qubits for QUBO encoding, and pulse shaping tunes the laser pulses, preparing the QPU for solution exploration. The best quantum-derived solution, denoted by $\hat{x}$, is assessed using the SP on a CPU which determines whether a Benders cuts has to be added to the MP. Through iterative Benders cuts, the MP is refined, discarding "bad" solutions and guiding the process toward the most effective MILP solution.

executed prior to solving the MP, yielding the creation of the Hamiltonian which enables quantum sampling of various MP binary solutions. Once the best MP solution $\hat{x}$ is fixed, the SP which uses $\hat{x}$ as a parameter is solved by a classical solver like CPLEX [22] or Gurobi [23]. The SP optimal objective value is then used to assess the need to improve the current solution. If necessary, a Benders cut, which is a linear inequality derived from the SP, is integrated into the MP in the subsequent iteration to eliminate infeasible solutions and guide the algorithm toward optimality. Our iterative method is designed to achieve an optimal or near-optimal solution for the original MILP and is automatically adapted to various MILP scenarios.

The contributions of this study are manifold. We introduce an automated process for converting the MP into a QUBO model. First, we present a heuristic for register embedding. Then we implement a variational algorithm for pulse shaping, incorporating gradient descent techniques. Additionally, we conduct a proof of concept (POC) that demonstrates superior performance over a D-Wave annealer application of BD [13]. Moreover, we perform numerical results on a large set of MILP instances. This shows our method's efficiency in comparison to a classical BD approach where the MP is solved using simulated annealing [24]. The adaptability of BD to various MILP structures, along with the scalability and precision of neutral atoms in solving QUBOs, forms the foundation of our research motivation. We have been able to utilize neutral-atom computation to develop an automated problem-agnostic framework for solving MILPs through BD.

This paper is organized as follows. Section II provides an overview of the developments in (hybrid) decomposition methods. Section III presents both the physics and mathematical material used in this work. The conversion of the MP into a QUBO model is detailed in Sec. IV. Section V presents the design of the quantum algorithm. The overall hybrid algorithm is described in Sec. VI. Our POC and numerical results are discussed in Sec. VII. Section VIII summarizes the paper and discusses directions for future work.

II. RELATED WORK

Classical decomposition strategies, including methods such as Dantzig-Wolfe decomposition [6] and column generation [25], efficiently address NP-hard optimization problems by subdividing them into smaller manageable parts. In recent advancements, diverse approaches are explored to address complex problems through classical-quantum decomposition methods. One instance, as presented in [26], focuses on applying these methods to an internet of things use case, specifically for edge server placement and workload allocation optimization. This study employs a hybrid quantum-classical framework, partitioning the problem into a QUBO MP and an LP SP. Using the IBM quantum computing software development kit QISKIT [27], the authors proposes a QAOA, within the framework of the alternating direction method of multipliers (ADMM) [28], an optimization technique that divides complex problems into smaller manageable SPs for iterative solving.

Benders decomposition stands out among existing decomposition strategies for its applicability across a variety of MILP problems. It addresses both integer and continuous variables while being adapted to any constraint structure. Nonetheless, classical BD encounters several challenges. Its main limitations include slow convergence to an optimal solution and time-consuming iterations. These issues often arise due to suboptimal initial solutions, weak Benders cuts, and the presence of multiple equivalent solutions. Addressing these issues, significant enhancements have been made to the classical BD algorithm [29–33]. These include the development of advanced stabilization methods, which enhance the algorithm's resilience against quality fluctuations of the solution, consequently accelerating convergence. Additionally, the implementation of techniques to generate multiple solutions and cuts in each iteration has been investigated and has demonstrated effectiveness in reducing the BD algorithm's time to solution [34].

Recently, hybrid classical-quantum BD approaches have started to show potential in tackling these MILPs complexities. Addressing the computational difficulty of the MP, Zhao *et al.* [13] developed a hybrid quantum BD approach. This method involves transforming the MILP model of the MP into a QUBO model, which is then solved using quantum annealing, on D-Wave machines. Similarly, Gao *et al.* [14] tackle a unit commitment problem using a hybrid quantum-classical generalized BD algorithm. This approach also involves converting the MP into a QUBO and solving it via quantum annealing on the D-Wave machine, showing improved performance over the method of ADMM.

Further research by Chang *et al.* [15] explores the hybrid BD on noisy intermediate-scale quantum processors, demonstrating efficiency in small-scale scenarios, particularly in power-system-specific MILP. Their algorithm is tested on a D-Wave 2000Q QPU. Furthermore, Fan and Han [16] investigated the application of BD in network resource optimization areas like network function virtualization and multiaccess

edge computing. Franco *et al.* [35] demonstrate that while BD applies to a wider range of MILP problems, potentially requiring more qubits in extreme cases, Dantzig-Wolfe decomposition is more qubit efficient but limited to structurally constrained problems. On the quantum side, the authors use the D-Wave system.

While significant progress has been made in hybrid BD using annealing and gate-based quantum computing methods, we believe the exploration of neutral-atom quantum computing within this framework remains largely unexplored, which is the purpose of our present research.

III. BACKGROUND

This section presents the technical tools used in our research, including a brief introduction to quantum computing with neutral atoms and the foundations of classical BD.

A. Neutral-atom QPUs and optimization problems

Neutral-atom QPUs utilize state transitions in valence electrons of atoms like rubidium to establish qubit states. These states include the stable ground state $|g\rangle$ or $|0\rangle$ and an energized Rydberg state $|r\rangle$ or $|1\rangle$. In these QPUs, the atoms arranged in a specific spatial configuration, the so-called register, contribute in defining an effective Hamiltonian $H(t)$, which governs the dynamics of the quantum system. The Hamiltonian is given by

$$H(t) = \Omega(t)\sum_{u=1}^{|V|}\hat{\sigma}_u^x - \Delta(t)\sum_{u=1}^{|V|}\hat{n}_u + \sum_{u<v=1}^{|V|}U_{uv}\hat{n}_u\hat{n}_v. \quad (1)$$

Here $\Omega(t)$ denotes the Rabi frequency amplitude that controls the rate of state transitions of qubits, $\Delta(t)$ represents the laser detuning which affects the probability of atom excitation, and U_{uv} is the interaction strength between atoms u and v. The terms $\hat{\sigma}_u^x$ and $\hat{n}_u$ correspond to the Pauli X operator and $\hat{n}_u = (\mathbb{I} + \hat{\sigma}_u^z)/2$, respectively. Finally, V represents the set of atoms in the register.

Neutral-atom QPUs naturally solve the maximum independent set (MIS) problem [36] in unit disk graphs [37]. The MIS problem, which involves finding the largest set of nonadjacent vertices (atoms), is optimally addressed by these QPUs, due to the Rydberg blockade mechanism. This mechanism uses the principle that two atoms within a certain distance cannot both be in an excited state simultaneously. In the context of encoding unit disk graphs, an edge between two vertices exists if they are close enough for the Rydberg blockade to take effect; essentially, if the atoms are within the blockade radius, they are considered adjacent in the graph. This spatial encoding reflects the edges and vertices of the unit disk graph directly into the quantum system, facilitating a native solution of the MIS.

More generally, neutral-atom QPUs are capable of solving any optimization problems that can be encoded into a QUBO format, as demonstrated by studies such as [17,38]. Here the Rydberg blockade mechanism offers a method for the encoding of optimization problems into a QUBO format. Each pair of adjacent atoms, within the blockade radius, can be seen as a product of two binary variables multiplied by the interaction of the atoms. Solving a QUBO problem with neutral-atom devices entails the following two major phases.

1. Problem encoding via register embedding

In this initial phase, called register embedding, the problem is embedded into the QPU register by spatially arranging the atoms. The spatial configuration of the atoms is designed in such a way that their interactions U_{uv} in Eq. (1) represent the off-diagonal terms of the QUBO instance. This setup is crucial as the quality of the solution for the input QUBO instance is highly dependent on how well the resulting interactions U_{uv} mimic the off-diagonal QUBO coefficients.

2. Problem solving via pulse shaping

The second phase includes the design and execution of specific pulse sequences to guide the system toward a good solution to the original QUBO. Pulse design, a process known as pulse shaping, is essential for finalizing the Hamiltonian construction. It involves setting parameters such as pulse duration T, laser detuning $\Delta(t)$, and Rabi frequency $\Omega(t)$. The choice and the effectiveness of pulse shaping in solving the problem using a neutral-atom QPU depends on the encoding phase.

(a) Pulse shaping under exact encoding. If the problem is encoded exactly during the encoding phase, adiabatic pulses can be designed. These pulses leverage the adiabatic theorem [39] to ensure the system smoothly transitions toward its ground state, which is ideal for problems like the MIS in unit disk graphs [36]. In such cases, careful adjustments of $\Omega(t)$ and $\Delta(t)$ over time guide the system adiabatically from its initial state to the desired final state, maintaining the system in its ground state throughout and leading to high-quality solutions.

(b) Pulse shaping under approximate encoding. When exact encoding is not feasible, variational algorithms are employed to design pulses that still aim to steer the system toward effective solutions. Although these pulses may not be adiabatic, they are crafted to achieve satisfactory results by approximating the desired Hamiltonian dynamics, thus navigating the system toward good solutions for the input QUBO instance under constrained or imperfect encoding conditions.

Note that neutral-atom QPUs are resilient to noise, which helps in the efficient solving of combinatorial optimization problems. As highlighted in [40], errors in digital quantum computing can propagate from one gate to another, potentially compounding as the computation progresses. In contrast, the analog evolution in neutral-atom quantum computing inherently reduces error propagation. This robustness is due to the direct correspondence between the state of the computational basis in which the qubits are measured and the solution to the optimization problem. In some cases, noise can even enhance performance by helping the system explore a broader solution space, as suggested in [41].

For an in-depth understanding of neutral-atom-based quantum computing, one may refer to [42].

B. Benders decomposition for MILPs

Benders decomposition works by separating a MILP known as the original problem (OP) into a MP, which deals with discrete variables, and a SP, which focuses on the remaining variables, often continuous. To perform BD, the OP is reformulated into an equivalent MILP. This reformulation relies on the principles of polyhedral and duality theory in linear optimization [43]. Consider real matrices A of dimensions $m_1 \times n$, G of dimensions $m_1 \times p$, and B of dimensions $m_2 \times n$. Let c, h, b, and b' be vectors with dimensions n, p, m_1, and m_2, respectively. The OP is initially expressed as

$$\max_{x,y} c^T x + h^T y \tag{2}$$

$$\text{s.t. } Ax + Gy \leqslant b, \tag{3}$$

$$Bx \leqslant b', \tag{4}$$

$$x \in \{0,1\}^n, \tag{5}$$

$$y \in \mathbb{R}_+^p. \tag{6}$$

The constraints (5) are integrality constraints; they introduce binary decision variables x. The constraints (6) introduce non-negative continuous decision variables y. The objective function (2) consists in maximizing a linear function of x and y. The constraints (3) associate the binary variables with the continuous ones, while the constraints (4) exclusively involve the binary variables.

Following the principles of standard BD, the constraints (3) are included in the SP, with the objective function $h^T y$. On the other hand, the constraints (4) are incorporated into the MP.

For a fixed solution $\hat{x}$ from the MP, the LP version of the SP is

$$\max_{y} h^T y \tag{7}$$

$$\text{s.t.} \quad A\hat{x} + Gy \leqslant b, \tag{8}$$

$$y \in \mathbb{R}_+^p. \tag{9}$$

Its dual, denoted by SPD, is

$$\min_{\mu} \quad f(\hat{x}) = (b - A\hat{x})^T \mu \tag{10}$$

$$\text{s.t.} \quad G^T \mu \geqslant h, \tag{11}$$

$$\mu \in \mathbb{R}_+^m. \tag{12}$$

Let x^* be the optimal solution of the SP and y^* be the optimal solution of the SPD. By strong duality theory [44], the optimal objective value of the SP is equal to the optimal objective value of the SPD. We have that

$$f(x^*) = h^T y^*.$$

In linear programming, the set of all feasible solutions defined by linear constraints forms a polyhedron. For a bounded polyhedron, the vertices, known as extreme points, are key to finding potential optimal solutions. According to the corner-point theorem, also known as the fundamental theorem of linear programming [45], if an optimal solution exists in a bounded linear program, it will be located at one of these vertices. In contrast, unbounded linear programs may still have extreme points, but they are not guaranteed to provide bounded optimal solutions. Instead, these unbounded problems can exhibit directions along which the objective function can increase indefinitely without violating the constraints. These directions are characterized by vectors known as extreme rays, emerging from the polyhedron and indicating where the objective function can grow indefinitely while still satisfying all the constraints.

The feasible solution space of an LP problem is well known to be characterized by a combination of its extreme points and extreme rays, as established by Minkowski's theorem [46]. In the context of BD, applying this theorem to the SPD enables the reformulation of the OP into an equivalent MILP model as follows:

$$\max_{x,\Phi} \quad c^T x + \phi \tag{13}$$

$$\text{s.t.} \quad (b - Ax)^T \mu_o \geqslant \phi \, \forall \, o \in \mathcal{O}, \tag{14}$$

$$(b - Ax)^T r_f \geqslant 0 \, \forall \, f \in \mathcal{F}, \tag{15}$$

$$Bx \leqslant b', \tag{16}$$

$$x \in \{0,1\}^n, \tag{17}$$

$$\phi \in \mathbb{R}. \tag{18}$$

In this reformulated OP, only one continuous variable ϕ is used. The remaining variables are the binary variables x. Constraints (14) and (15) represent the Benders optimality cuts and feasibility cuts, respectively. Here the sets $\mathcal{O}$ and $\mathcal{F}$ are the extreme points and extreme rays of the SPD, respectively, and the vectors μ_o for extreme points and r_f for extreme rays are used to construct Benders cuts. They can be obtained through conventional solvers, such as ILOG CPLEX [22]. Note that sets $\mathcal{O}$ and $\mathcal{F}$ can be exponential in number. However, BD often efficiently finds an optimal solution using a selected subset of these sets.

C. Principle of the Benders decomposition algorithm

The approach begins with a restricted version of the MP (13)–(18), initially setting both optimality and feasibility cut sets $\mathcal{O}$ and $\mathcal{F}$ to empty ($\mathcal{O} = \mathcal{F} = \emptyset$). As the algorithm progresses, it iteratively adds optimality and feasibility cuts to the MP. At each iteration, the MP is solved to derive an optimal solution, which is then used as a parameter to solve the SP.

If the SP is feasible, an optimality cut is generated, guiding future MP solutions toward the OP's optimal solution. This process uses the real variable ϕ in the formulation (13)–(18). The optimal objective value of the SP $(b - Ax)^T \mu_o$ is compared to ϕ, and if $(b - Ax)^T \mu_o \leqslant \phi$, the optimality cut (14), constraining ϕ to not exceed the objective value of the actual extreme point of the SPD, is added. On the other hand, as the OP maximizes ϕ, at optimality, ϕ will necessarily be equal the objective value of the SPD and thereby equal the optimal objective value of the SP (according to the strong duality theorem). Conversely, if the SP is infeasible, a feasibility cut is produced to eliminate infeasible solutions from the MP's

ALGORITHM 1. Classical Benders decomposition.

Input: Problem parameters c, h, A, G, b, B, b'
Input: $\phi_{\max}$ ▷ Initial upper bound for ϕ
Initialize $\mathcal{O} \leftarrow \emptyset, \mathcal{F} \leftarrow \emptyset$
$\phi \leftarrow \phi_{\max}$
convergence $\leftarrow$ False
while not convergence **do**
  Solve the MP with current cuts
  $x, \phi \leftarrow$ solution of MP
  Solve the SP using x to find y
  if the SP is feasible **then**
    $\mu_o \leftarrow$ dual variables from the SP
    $cut\_value \leftarrow (b - Ax)^T \mu_o$
    if $cut\_value < \phi$ **then**
      Add optimality cut to $\mathcal{O}$: $\phi \leqslant cut\_value$
    else
      convergence $\leftarrow$ True
    end if
  else
    Generate a feasibility cut based on SP infeasibility
    Add this cut to $\mathcal{F}$
  end if
end while
return Optimal solution x, ϕ, y

solution space. These cuts act as a filter, maintaining consistency within the solutions of the OP.

The algorithm operates as follows. The MP tries to maximize the value of ϕ, yielding an upper bound of the latter, while the SP, upon finding a feasible solution, generates an optimality cut that decreases this upper bound. This iterative process of maximizing and constraining ϕ continues until the optimality cut from the SP can no longer decrease the value of ϕ. At this point, we have that $(b - Ax)^T \mu_o \geqslant \phi$ and the algorithm terminates, indicating that the MP and SP have converged to an optimal solution for the OP. Algorithm 1 outlines the implementation of the classical Benders decomposition method.

It is important to note that the resolution of the MP presents considerable complexity, mainly attributed to the fact that the latter integrates binary variables. This complexity becomes more critical as the solution process progresses, particularly with the dynamic integration of Benders cuts. Consequently, the MP frequently becomes a computational bottleneck. To overcome this, we study the application of neutral-atom-based quantum computing in a complete hybrid classical-quantum framework. In the next section we detail the reformulation of the MP into a QUBO model, which is suited for this type of QPU.

IV. MASTER-PROBLEM REFORMULATION TO QUBO

A QUBO problem can be formulated as $\min_z \{z^T Q z \mid z \in \{0, 1\}^t\}$, where Q is a symmetric matrix and z is a binary vector. The objective of this optimization problem is to find the binary vector z^* that minimizes the quadratic objective function $z^T Q z$ over all binary vectors. Before using a QPU for optimizing the MP, we transform the latter, originally represented as a MILP, into a QUBO model. In what follows we detail the methodology of this reformulation, associated with a single BD iteration. For foundational insights into the MILP to QUBO conversion, refer to [21]. Note that similar methodologies are discussed in [13]. Here we give a detailed approach that further refines the upper bounds of the continuous variables resulting from the conversion, which positively impacts the required number of qubits as well as the convergence of the algorithm.

A. Master-problem objective-function reformulation

The objective function in the MP, as defined in (13), contains a linear term and a continuous variable, given by $c^T x + \phi$. The linear component $c^T x$, which exclusively involves binary variables, is directly adaptable to a QUBO. This is achieved by employing the diagonal matrix $\mathrm{diag}(c)$, with the vector c populating its diagonal, thereby transforming $c^T x$ into

$$H_c = x^T \mathrm{diag}(c) x. \tag{19}$$

The continuous variable ϕ can be binary encoded using a binary vector w of length L, formulated as

$$H_\phi = \sum_{i=0}^{P-1} 2^i w_i + \sum_{j=1}^{D} 2^{-j} w_{P+j} - \sum_{k=1}^{N} 2^{k-1} w_{P+D+k}. \tag{20}$$

Here P represents the number of bits for the positive-integer part of ϕ. We have that $P = \lfloor \log_2(\phi_{\max}) \rfloor + 1$, where $\phi_{\max}$ is an upper bound of ϕ. For the fractional part, the number of bits D can be obtained based on a desired precision ϵ, calculated as $D = \lfloor \log_2(\epsilon) \rfloor + 1$. The N denotes the number of bits for the negative-integer part of ϕ, leading to a total length $L = P + D + N$ for the vector w. Note that careful determination of P, D, and N is crucial, as it affects both the numerical precision of ϕ and the quantum resource requirements in terms of the number of qubits needed.

To determine $\phi_{\max}$, we address the linear relaxation of the formulation of the OP as given by (2)–(6), without considering $c^T x$ in the objective function. The linear relaxation is the formulation obtained by relaxing integer constraints (5), allowing continuous values, and is given by

$$\max_{x,y} \quad \phi_{\max} = h^T y \tag{21}$$

$$\text{s.t.} \quad Ax + Gy \leqslant b, \tag{22}$$

$$Bx \leqslant b', \tag{23}$$

$$x \in [0, 1]^n, \tag{24}$$

$$y \in \mathbb{R}_+^p. \tag{25}$$

B. Constraint reformulation

1. Master constraints

The reformulation of the MP constraints (4) involves integrating slack variables, a common technique in classical optimization for transforming inequalities into equalities. The inequalities $Bx \leqslant b'$ are first converted into the equalities $Bx + s_m - b' = 0$, where s_m is a vector of continuous positive variables. Let $1 \leqslant k \leqslant m_2$. The kth component s_m^k of the slack

vector s_m undergoes a binary encoding, yielding

$$s_m^k = \sum_{i=0}^{Q_1^k-1} 2^i v_i^{m,k} + \sum_{j=1}^{R_1^k} 2^{-j} v_{Q_1+j}^{m,k}.$$

Here $v^{m,k}$ denotes a binary vector with a length of $Q_1^k + R_1^k$, where Q_1^k represents the number of qubits required for the integer part of s_m^k and R_1^k corresponds to the number of qubits for its fractional part. The value of Q_1^k can be determined by $\lfloor \log_2(s_{\max}^k) \rfloor + 1$, where $s_{\max}^k$ is the upper bound for s_m^k obtained by solving the linear program

$$s_{\max}^k = \max_{x,y} \quad b_k' - B_k x \tag{26}$$

$$\text{s.t.} \quad Ax + Gy \leqslant b, \tag{27}$$

$$Bx \leqslant b', \tag{28}$$

$$x \in [0,1]^n, \tag{29}$$

$$y \in \mathbb{R}_+^p. \tag{30}$$

Let π_1^k be a positive penalty coefficient associated with the kth MP constraint. The QUBO reformulation of the constraints (4) is thus given by

$$H_M = \sum_{k=1}^{m_1} \pi_1^k \left(B_k x + s_m^k - b_k\right)^2. \tag{31}$$

Minimizing the Hamiltonian H_M to zero ensures that the constraints $Bx \leqslant b'$ are satisfied. Otherwise, a cost on any deviation from zero, violating the constraint, is added.

2. Optimality and feasibility cuts

Let C be the number of Benders cuts added during the process and let $1 \leqslant k \leqslant C$ denote the kth Benders cut. We denote by $v^{o,k}$ and $v^{f,k}$ the binary vectors of lengths $Q_2^k + R_2^k$ and $Q_3^k + R_3^k$, respectively, used in reformulating the kth optimally and feasibility cut (14) or (15), depending on the type of the cut added in the kth iteration. Here Q_2^k and Q_3^k denote the numbers of qubits allocated for the integer parts of slack variables s_o^k and s_f^k, respectively. The terms R_2^k and R_3^k correspond to the number of qubits used for the fractional part of s_o^k and s_f^k, respectively. These can be obtained based on the desired precision.

Following the same steps used for the MP constraints, we obtain

$$s_o^k = \sum_{i=0}^{Q_2^k-1} 2^i v_{o,k}^i + \sum_{j=1}^{R_2^k} 2^{-j} v_{Q_2+j}^{o,k}, \tag{32}$$

$$s_f^k = \sum_{i=0}^{Q_3^k-1} 2^i v_i^{f,k} + \sum_{j=1}^{R_3^k} 2^{-j} v_{Q_3+j}^{f,k}. \tag{33}$$

Let Π_2^k be a positive penalty coefficient associated with the optimality cut and Π_3^k the one associated with the feasibility cuts. Using (32) and (33), we obtain the Hamiltonians

$$H_O = \pi_2^k \left[H_\phi + \left(\mu_o^k\right)^T Ax + s_o^k - b^T \mu_o^k\right]^2, \tag{34}$$

$$H_F = \pi_3^k \left[\left(r_f^k\right)^T Ax + s_f^k - b^T r_f\right]^2. \tag{35}$$

Here μ_o^k represents the dual value and r_f^k denotes the dual ray, both of which are obtained by solving the kth SP. Note that each SP has the potential to yield either a dual value, in cases where it is feasible, or an extreme ray, if it is infeasible. Consequently, for any given SP, only one of the Hamiltonians (34) or (35) is applicable. The QUBO formulation of the OP is given by the sum of the Hamiltonians

$$H_P = H_\phi + H_c + H_M + H_O + H_F. \tag{36}$$

C. Reformulation discussion

In the process of converting the MP into the QUBO model, several algorithmic challenges arise.

1. Qubit count limitation and convergence of the algorithm

The process of quadratizing the variable ϕ and the constraints in the BD (13)–(16) necessitates the use of additional qubits. The total number of qubits of the initial MP is given by

$$t = n + P + D + N + \left(\sum_{i=1}^{m_2} Q_1^i + R_1^i\right).$$

As the algorithm progresses, this number can significantly increase, primarily due to the generation of Benders cuts. Each iteration k increases the number of qubits by $Q_2^k + R_2^k$ or $Q_3^k + R_3^k$, depending on the type of the cut. A critical consideration in this process is the current limitations of quantum computing hardware, particularly in terms of qubit availability. To ensure that the computation remains feasible on existing quantum computers, it is crucial to accurately estimate the upper bounds of the real variable ϕ as well as slack variables s. These estimations directly impact the number of qubits required. As previously established, we utilize the linear relaxation of the OP to calculate upper bounds.

Addressing the linear relaxations defined in Eqs. (21)–(25) and (26)–(30) tightens the upper bounds of $\phi_{\max}$ and the slack variables $s_{\max}^k$ compared to the method described by Zhao *et al.* [13]. This improvement results from incorporating additional valid constraints (22) and (23) into the linear programs, which also leads to tighter values for $\phi_{\max}$ and $s_{\max}^k$.

By providing tighter upper bounds, the algorithm reaches the optimal solution more quickly as the bounds limit the solution space the algorithm needs to explore, thus accelerating convergence. Additionally, solving these linear continuous programs is generally computationally easy using classical methods.

2. Qubit count vs numerical precision

An accurate estimation of qubit count is essential to avoid numerical precision issues. In fact, a bad estimation of this count can lead to significant numerical precision issues. For instance, an underestimation of ϕ could stop the algorithm before its effective end, resulting in poor-quality solutions.

Conversely, an overestimation of ϕ can cause almost endless loops, or the generation of no good Benders cuts, thus resulting in unfeasible solutions. Therefore, it is important to find a balance between precision and qubit availability.

3. Penalty values and solution efficiency

Finally, the selection and tuning of penalty values in the QUBO model are also important. These values guide the quantum algorithm toward optimal or near-optimal solutions. However, the process of setting static penalty weights for various types of problems is not trivial. This is because values that are too small will lead to infeasible solutions, while values that are too large may lead to slower convergence. Many studies have explored different methods of setting penalty weights within the context of QUBO formulations [47–49]. The study of the best penalty configuration is not within the scope of this work. Finding the optimal penalties for the QUBO is still an open research topic.

Future directions addressing the presented challenges will be discussed in the Conclusion (see Sec. VIII). The next section presents the methodology to solve the QUBO in a neutral-atom QPU.

V. QUANTUM ALGORITHM DESIGN

The development of a quantum algorithm in a computer based on neutral atoms includes two major steps: register embedding and pulse shaping. In this section we present the algorithms developed for these steps.

A. QUBO embedding strategy

The register embedding involves placing atoms at specific locations within a register having its own distance constraints. The aim is to find the placement aligning the interaction matrix U, created based on the distances between the placed atoms and a device-specific constant, as closely as possible to the predefined QUBO matrix.

Formally, the problem can be defined as follows. Given (i) a register defined by a set of positions P, respecting a minimum distance between each pair of positions and a maximum distance of any position from a reference point c, referred to as the center of the register; (ii) a set V of n atoms to be placed inside the register; (iii) an n-dimensional QUBO matrix Q, we define $\mathcal{P}$ as the set of injective mappings from V to P, with each mapping $\phi : V \to P$ in $\mathcal{P}$ associated with a position $p \in P$ for each atom $v \in V$. The placement of atoms on a subset of positions yields an interaction matrix U_ϕ, whose components are defined by $u_{ij} = \frac{C_6}{r_{ij}^6}$, where C_6 is a device-dependent constant and r_{ij} denotes the distance between the atoms i and j induced by the placement ϕ.

The register embedding problem (REP) consists of selecting the best placement $\phi^* \in \mathcal{P}$ that minimizes the distance between the interaction matrix U and the QUBO matrix Q:

$$\min_{\phi \in \mathcal{P}} \sum_{(i,j) \in V^2, i \neq j} |q_{ij} - u_{ij}|. \tag{37}$$

The REP is an NP-hard problem. To address this complexity, we develop the heuristic presented in Algorithm 2. The algorithm starts with a random selection of an atom from the set V.

ALGORITHM 2. Register embedding algorithm.

Input: V ▷ Set of atoms.
Input: P ▷ Set of positions.
Input: Q ▷ QUBO matrix.
Randomly select an atom u from V
$P_a \leftarrow (u, c)$ ▷ P_a is the set of atom-position pairs representing the placement of atoms. Initially contains (u, c), which represents the placement of the randomly chosen atom u on the center of the register c.
$U \leftarrow 0_{\mathbb{R}_{n \times n}}$ ▷ initially no interactions
$P \leftarrow P \setminus \{c\}$ ▷ Remove c from P
$V \leftarrow V \setminus \{u\}$ ▷ Remove atom u, already embedded
while $V \neq \emptyset$ **do**
    Select atom u from V
    Initialize min_sum $\leftarrow \infty$
    For each atom p in P:
        $sum \leftarrow 0$
        For each position v such that there exists $(v, p_v) \in P_a$:
            $sum \leftarrow sum + |Q_{u,v} - U_{p,\Pi(v)}|$
        If $sum <$ min_sum then:
            min_sum $\leftarrow sum$
            best_position $\leftarrow p$
        $P_a \leftarrow P_a \cup \{u,$ best_position$\}$ ▷ Place u on p
        $P \leftarrow P \setminus \{$best_position$\}$ ▷ Position best_position no longer available
    $V \leftarrow V \setminus \{u\}$ ▷ Remove atom u, already embedded
end while
return P

The selected atom is placed at the center of the register c and the process continues by iteratively evaluating and embedding the remaining atoms.

At each iteration, the algorithm examines all available positions for the selected atom. It computes the total deviation from the desired QUBO matrix. The position that yields the lowest deviation is chosen as the best placement for that atom. For each atom u in V, we compute, the position of the lowest deviation is the one that minimizes the sum of absolute value differences between the elements of the QUBO matrix Q and the interaction matrix U. The interaction matrix U is dynamically updated as each atom is placed, reflecting the current state of interactions in the register.

Once a position is selected, it is removed from the set P of available positions and the atom is removed from the set V of unplaced atoms. The process is repeated until all atoms are placed, resulting in a configuration that finds a solution of (37), minimizing the distance between the interaction and QUBO matrices.

It is worth noting that deviations between U and Q are critical because they can lead to suboptimal or even infeasible solutions for the MP. Such outcomes can result in weak or no good Benders cuts. Specifically, weak cuts are those that do not significantly tighten the value of ϕ, thereby failing to accelerate the convergence toward the best solution. No good cuts are invalid for the original problem, potentially rendering the OP itself infeasible. These aspects highlight the importance of achieving an accurate register embedding method, as they directly impact the quality and feasibility of the solution to the OP.

B. Variational algorithm for pulse shaping

This section presents a variational algorithm for optimizing pulse parameters. Our objective is to identify the optimal settings for pulse parameters for a register embedded using the heuristic detailed in Sec. V A: the maximum amplitude $\Omega_{\max}$, the initial detuning δ_{init}, the final detuning δ_{final}, and the pulse duration T, all within predetermined bounds. Since this procedure aims to find the optimal shape for the laser pulses that control the quantum system, it will be referred to as pulse shaping.

An iterative optimization procedure is established, progressively refining these parameters to enhance the pulse's efficiency. Initially, the average value of the parameters is used to construct a pulse, with a chosen shape (such as an interpolated waveform). This pulse is then executed on the register, which is initialized in the Rydberg state $|\psi_0\rangle$. The evolution of the quantum state under the influence of the pulse is described by an effective Hamiltonian H_{eff}, as expressed in Eq. (1). This effective Hamiltonian governs the dynamics of the system and is not necessarily unitary. The final state $|\psi_f(\delta_{\text{init}}, \delta_{\text{final}}, T)\rangle$ after applying the pulse, during a time T, is given by

$$|\psi_f(\delta_{\text{init}}, \delta_{\text{final}}, T)\rangle = H_{\text{eff}}|\psi_0\rangle.$$

If the system consists of M atoms, the final state will typically be a normalized superposition of basis states, each uniquely corresponding to a binary bit string of length M,

$$|\psi_f\rangle = \sum_{i=1}^{2^M} a_i|b_i\rangle,$$

where $\sum_i |a_i|^2 = 1$ and

$$|b_i\rangle = \left|b_i^1\right\rangle \otimes \cdots \otimes \left|b_i^M\right\rangle, \quad b_i^j = |0\rangle \text{ or } |1\rangle.$$

Achieving perfect knowledge of the quantum state $|\psi_f\rangle$ (and thus the coefficients a_i) would require an exponential number of resources as the system scales. Therefore, the state is usually only approximately known through repeated measurements. Each measurement of the state $|\psi_f\rangle$ results in the extraction of one of the bit strings b_i with probability $|a_i|^2$. Collecting N samples of the state yields a set of pairs $\{(b_i, w_i^{(N)})\}_{i=1,\ldots,2^M}$, where $w_i^{(N)}$ denotes the number of times the bit string b_i was measured out of N tries.

To each bit string b_i, a cost $C(b_i) = H_P(b_i)$ can be assigned according to the objective value of the QUBO H_P (36) for b_i. The effectiveness of the optimization is evaluated by calculating the expectation value of the problem Hamiltonian H_P over all samples. Specifically, the average cost from all the samples is computed as

$$\langle C\rangle = \frac{1}{N}\sum_i w_i C(b_i).$$

Gradient boosted regression trees (GBRTs) [50], a refined numerical optimization technique, are then applied to identify the parameter set that minimizes the cost function, thereby finding the optimal parameters that achieve the minimum average cost. This sampling and optimization cycle is executed repeatedly, a total of p times. The parameters and sample collection that yield the most favorable cost will be selected.

ALGORITHM 3. Pulse optimization algorithm.

Input: Ω_{bounds}, δ_{bounds}, T_{bounds} ▷ Parameter intervals for Rabi frequency, detuning, and pulse duration.
Input: *Register*, H_P, p ▷ Quantum register, problem Hamiltonian, number of iterations p.
Initialize params as the initial solution (the average of parameter bounds).
for $i = 1$ to p **do**
  cost, *samples* ← EvaluateSequence(*params*)
  if *cost* < best cost found so far **then**
    Update *best_params* and *best_samples* with current *params* and *samples*.
  end if
end for
params ← Generate new parameters based on GBRT optimization or initial parameters for the first iteration.
return *best_params*, *best samples*.
function EVALUATESEQUENCE(params)
  Generate the pulse sequence with *params*.
  Apply the sequence to the quantum system.
  Measure the outcome to collect samples.
  Calculate the average cost $\langle C\rangle$.
  return the average cost $\langle C\rangle$, samples.
end Function

The choice of p is important, as it affects both the solution's quality and the overall cost of the iterative process; an optimal iteration count controls the balance between achieving a satisfactory solution and maintaining reasonable computational resources. The algorithm's pseudocode is delineated in Algorithm 3.

Figure 2 shows an example of an optimized pulse corresponding to a gradual parameter evolution over an extended duration T. This is particularly used by the quantum adiabatic algorithm [51], which emerges as a potent strategy for efficiently addressing the optimization problem.

VI. HYBRID QUANTUM-CLASSICAL BENDERS ALGORITHM

This section presents the overall BD hybrid quantum-classical algorithm, which uses elements discussed in Sec. III and the study from Secs. IV and V. The algorithm combines quantum and classical computing resources to address MILP problems using the BD algorithm. The sequence of operations

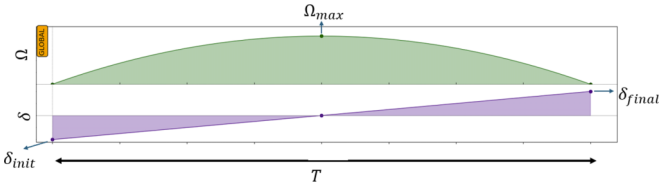

FIG. 2. Shape of an optimized pulse. The change in Rabi frequency Ω and global detuning δ during the pulse T is demonstrated. The Ω exhibits a peak at $\Omega_{\max}$ while δ varies linearly from δ_{init} to δ_{final}.

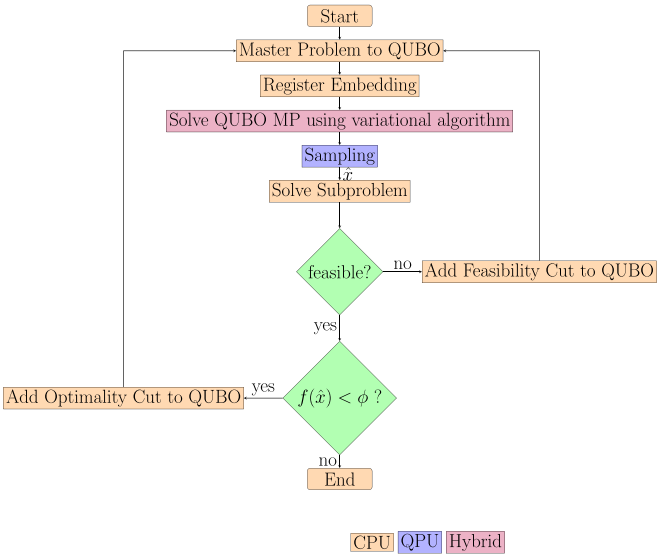

FIG. 3. Hybrid BD quantum-classical algorithm flowchart, showing the sequence of steps in our hybrid quantum-classical algorithm. Starting with the reformulation of the MP into a QUBO, the process employs register embedding to spatially arrange atoms for QUBO encoding. This is followed by the application of the variational algorithm, where pulse parameters are optimized for quantum sampling. The QPU then samples solutions, with the best candidate $\hat{x}$ selected based on the costs. If the SP is infeasible, a feasibility cut is introduced into the MP. If feasible, an optimality cut is applied if necessary $[f(\hat{x}) < \phi)]$. The process iterates between these steps until the best solution is found, resulting in the final values of x and y.

and decision-making processes involved in this algorithm is represented in the flowchart of Fig. 3.

The algorithm uses the decomposition of the OP, as introduced in (2)–(6), into a MP and a SP. Initially, the MP (13)–(18) is reformulated into a QUBO model (36). Subsequently, the register embedding heuristic, detailed in Algorithm 2, is employed. This heuristic arranges atoms in a spatial configuration that closely aligns the interaction matrix with the QUBO.

Following the register embedding, the variational algorithm is applied. In this step, pulse parameters are tuned according to the established register configuration. Next the QPU is used as a sampler. Multiple measurements are performed on the final quantum states. These measurements yield various potential solutions for the OP, each with an associated probability of occurrence and cost. The solution $\hat{x}$, which yields the lowest cost, is selected for further processing.

The next phase involves classical computing methods to solve the SP (7)–(9). The feasibility of its solution is assessed. If the SP is infeasible, a feasibility Benders cut (15) is generated and added into the MP, redirecting the process to the QUBO reformulation. Conversely, if the solution is feasible, a subsequent check is conducted to compare the optimal objective value of the SP $f(\hat{x})$ with the value of the variable ϕ. In case $f(\hat{x})$ is lower, an optimality cut (14) is added to the MP and the algorithm revisits the QUBO reformulation step. If not, the process stops, outputting the latest values of x and y as the solution obtained by the hybrid BD algorithm.

VII. NUMERICAL EXPERIMENTS

In this section we present our numerical experiments. The primary objective is to provide a POC. Moreover, on a set of arbitrary MILP small instances, we conduct a comparative study between our hybrid BD solver and classical BD, with the MP solved using simulated annealing.

A. Proof of concept

For the POC, we use the same example presented in [13]. The MILP associated with the problem uses two binary variables $x_1, x_2 \in \{0, 1\}$ and four non-negative continuous variables $y_1, y_2, y_3, y_4 \geqslant 0$. The matrix description of the MILP is

$$A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ -1 & 0 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad G = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

$$B = [-1 \quad -1], \quad b' = [-1],$$

$$c^T = [-15 \quad -10], \quad h^T = [8 \quad 9 \quad 5 \quad 6],$$

In the presented use case, we set the penalty coefficients to the penalty of the master objective function $\pi_{\text{obj}} = 1$ and penalties of the constraints of the master constraints as well as optimality and feasibility Benders constraints, respectively, to $\pi_1 = \pi_2 = \pi_3 = 100$. Table I outlines the execution state of the BD algorithm applied to the example described above. The table includes MP QUBO, the literal expression of the initial MP; MP solution, the solutions obtained from the quantum sampler, i.e., values of variables x, s, and ϕ and the objective-function value, denoted by obj; SP solution, the solution to the SP, i.e., the objective value, values of variable y solution, and its dual μ; generated penalty, the literal expression of the penalty (if any); and type of penalty, the type of the penalty (if any).

Solving the use case using the hybrid BD algorithm, the final objective value is equal to the optimal solution optimal objective value determined by CPLEX. Moreover, as shown in Table I, our hybrid BD algorithm with neutral atoms concludes within two iterations, which marks an improvement compared to the five iterations required by the hybrid Benders algorithm presented in [13], using D-Wave. This faster convergence can be attributed to the fact that our quantum sampler helps in providing stronger Benders cuts. The neutral-atom QPU generates better MP solutions. Using these solutions, the SP yields tighter values of the left-hand side of optimality cuts (14). Consequently, the variable ϕ converges more rapidly, allowing us to confirm optimality in fewer iterations.

The distinctive advantages of using neutral atoms in our BD algorithm are primarily grounded in the superior problem encoding capabilities of neutral-atom QPUs. In D-Wave systems, the embedding procedure involves linking chains of

TABLE I. Trace of the BD algorithm solving process on the MILP.

	Iteration 1	Iteration 2
MP QUBO	$-(15x_1^2+10x_2^2)+100(-x_1^2-x_2^2+2^0 s_{m\_1\_1}+1)^2$ $+(2^4w_1^2+2^3w_2^2+2^2w_3^2+2^1w_4^2+2^0w_5^2)$	MP(it1)+ generated penalty
MP solution	$\Phi : 31.5$ obj : $21.5x_1 : 0$, $x_2 : 1, s_{m\_1\_1} : 0$	$\Phi : 17.0$ obj : $2.0\,x_1 : 1$, $x_2 : 0, s_{m\_1\_1} : 0$, $s1\_1 : 0$
SP solution	Objective value: 11.0 $y_1 : 0.0,\ y_2 : 0.0,\ y_3 : 1.0,\ y_4 : 1.0$ $\mu_1 : 5.0,\ \mu_2 : 0.0,\ \mu_3 : 0.0,\ \mu_4 : 6.0,$ $\mu_5 : -3.0,\ \mu_6 : -3.0,\ \mu_7 : 0.0,\ \mu_8 : 0.0$	Objective value: 17.0 $y_1 : 1.0,\ y_2 : 1.0,\ y_3 : 0.0,\ y_4 : 0.0$ $\mu_1 : 8.0,\ \mu_2 : 0.0,\ \mu_3 : 0.0,\ \mu_4 : 9.0,$ $\mu_5 : 0.0,\ \mu_6 : 0.0,\ \mu_7 : 0.0,\ \mu_8 : 0.0$
Generated penalty	$100[2^4w_1+2^3w_2+2^2w_3$ $+2^1w_4+2^0w_5+3.0(-x_1)+3.0(-x_2)$ $+2^0 s1\_1-(5.0+6.0)]^2$	
Type of penalty	Optimality	

qubits with a strongly ferromagnetic coupling ($K=-2$) to simulate a single logical variable. Additionally, these logical variables are then coupled with Q_{ij}, which is constrained within the narrow range $[-1, 1]$ (for D-Wave, $Q_{ij}=J_{ij}$) [52]. This restrictive range necessitates rescaling all couplings to fit within it, which can compromise the fidelity of the problem representation.

By contrast, our approach with neutral atoms leverages long-range Rydberg interactions, allowing for a much more faithful embedding. The Rydberg dipole-dipole interactions, characterized by C_6/r^6, provide a mechanism to fine-tune the interaction distances. This ability enables a versatile adjustment of interaction strengths over a broad spectrum due to the power-law decay, offering significant advantages in problem encoding. For our nine-variable all-to-all connected QUBO, this means we can handle the MILP problem's extensively varying coupling values without the need to rescale them, thus preserving the fidelity of the problem representation. This is particularly beneficial as our QUBO model features an exponentially broad range of coupling values, which poses a challenge in D-Wave systems where lower J_{ij} values can be drowned in thermal noise. Furthermore, the architecture of neutral atoms facilitates coherent annealing processes, where the dephasing and depolarizing timescales exceed the annealing period. This ensures that the system's evolution is predominantly isolated from the thermal environment, enhancing computational performance and stability.

The enhanced performance of our algorithm is not only attributable to the quantum hardware employed but also significantly influenced by our preprocessing strategy, which optimizes algorithm convergence. As detailed in Sec. IV, by solving linear relaxations and integrating the OP valid constraints, our strategy tightens the upper bounds of $\phi_{\max}$ and the slack variables $s^k_{\max}$. By providing a tighter upper bound, we help the algorithm reach the optimal solution more quickly because the tighter bounds limit the solution space the algorithm needs to explore, thus accelerating convergence. Additionally, this approach reduces the quantum resource requirements by decreasing the number of qubits needed for encoding, thereby enhancing resource utilization.

This use case demonstrates the potential of neutral-atom QPUs when integrated into a hybrid BD algorithm; it confirms the feasibility of applying such quantum computational resources and shows their ability to enhance performance, outperforming current state-of-the-art solutions.

B. Numerical results

We examine now the efficiency of the hybrid BD algorithm on several MILPs instances.

1. Description of MILP and implementation features

The instance generation process of our experimentation covers 450 MILPs randomly generated. We vary the number of variables and constraints, as well as the structure and coefficients of the constraint matrices, as follows.

(a) Variables x and y. The number of binary variables x ranges from 2 to 5, while the number of continuous variables y varies from 2 to 10. This setting is motivated by the exploration of different problem scales while respecting the capacity of the simulation in terms of the number of qubits.

(b) Constraint matrices A and G and vector b. Matrix A is populated with nonpositive random values and matrix G with non-negative values. This setting is chosen to yield positive slack variables, thereby reducing the number of required qubits needed in converting optimality and feasibility cuts to the QUBO formulation. The vector b is randomly generated with non-negative random values. These parameters define the set of constraints (3), whose number varies from 5 to 14.

(c) Matrix B and vector b′. Matrix B consists of a single row filled with $1'$s and vector b' is a random positive number strictly less than the size of x. This imposes a special constraint that interdicts choosing the solution where all the x variables are equal to 1. These parameters are related to the constraints (4). Only one constraint of this type is considered.

(d) Coefficients c and h. These are randomly generated vectors with non-negative random values, contributing to the variability in the objective function.

The randomness in these parameters generates various instances. We group the results based on the number of qubits used during the whole process and average the output for each

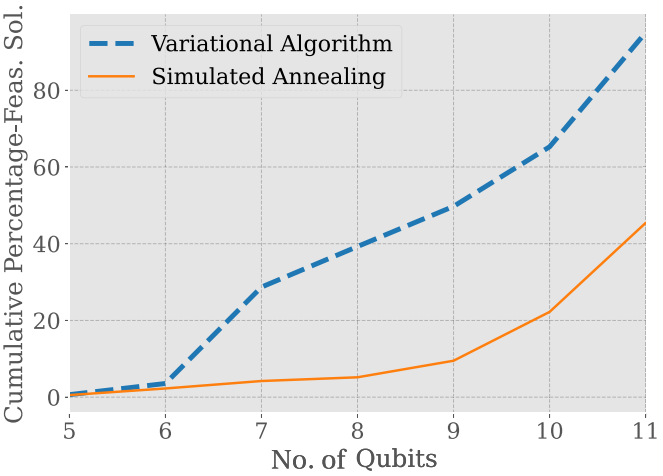

FIG. 4. Percentage of feasible solutions per number of qubits.

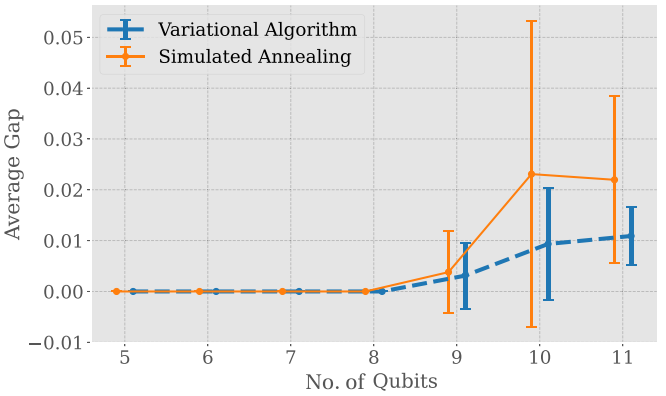

FIG. 5. Average gap per number of qubits.

qubit count. The evaluation of the algorithm performance is based on three key metrics.

(i) Percentage of feasible instances. This metric evaluates the algorithm's ability to find feasible solutions for the tested instances.

(ii) Gap to optimality. This metric indicates the quality of the solution obtained by the algorithm. It is defined as gap $= \frac{\text{obj(algo)}-\text{obj(opt)}}{\text{obj(opt)}}$, where obj is the objective-function value. The gap represents the distance of the solution provided by the algorithm from the optimal solution. The optimal solution is computed considering the OP in its compact formulation before being reformulated for a BD.

(iii) Number of iterations. This metric serves as an indicator of the time and energy consumption of the algorithm.

For the computational implementation and analysis of the generated MILP instances, the programming is conducted in PYTHON. The quantum pulses are simulated using Pulser [53], which is used for designing and emulating quantum protocols on neutral-atom devices. The variational algorithm resolution is conducted by SCIKIT-OPTIMIZE [54], a PYTHON library for optimization that is well suited for quantum algorithm parameter tuning. Finally, the compact formulation of the OP was solved to optimality using the CPLEX solver [22], a high-performance mathematical programming solver.

2. Results

We benchmark our algorithm with a fully classical BD where we solve the MP using simulated annealing. It is important to note that our test limits the number of qubits to 11, which is a threshold set by the simulation constraints of current quantum simulator capabilities.

The graphic in Fig. 4 illustrates the comparative performance of our hybrid BD with the variational algorithm and classical BD using simulated annealing, in terms of the cumulative percentage of feasible solutions, with respect to the number of qubits. It is evident that the variational algorithm surpasses simulated annealing consistently throughout the observed qubit range. Our variational algorithm demonstrates a pronounced increase in the cumulative percentage of feasible solutions, as the qubit count increases, achieving more than 95% with 11 qubits. In contrast, simulated annealing shows a more moderate progression.

Figure 5 shows the average gap to optimality (and the standard deviation with 95% confidence interval) for both hybrid BD with the variational algorithm and fully classical BD with simulated annealing as a function of the number of qubits. The gap to optimality serves as a crucial measure of solution quality. A smaller gap is indicative of a solution that is closer to the optimal one. The results presented here were conducted on MILPs where both methods, variational algorithm and simulated annealing, provided feasible solutions (thus, on the 45% of feasible MILPs given by simulated annealing). This ensures the fact that the gap is defined for both the variational algorithm and simulated annealing. It can be seen that, overall, both methods deliver solutions of good quality. The maximum average gap is attributed to simulated annealing and is equal to 2.3%. Nonetheless, observations indicate that the variational algorithm maintains a relatively stable average gap, suggesting a robust ability to generate solutions close to optimal across different qubit counts. Conversely, simulated annealing exhibits a comparable performance at lower qubit counts but deteriorates as the count becomes greater than 10. This performance degradation shows the potential scalability issues with simulated annealing when faced with increased problem complexity, represented by higher qubit counts, and consequently the potential ability of our algorithm to produce a high-quality solution when scaling up.

Figure 6 presents the average number of iterations (and the standard deviation with 95% confidence interval) for the hybrid BD with variational algorithm and the fully classical BD with simulated annealing as a function of the number of qubits. Iterations reflect the computational effort and, by extension, time and energy expenditure of the algorithms. Both algorithms show an increase in iterations with more qubits. This is explained by the complexity generated by the number of qubits. The simulated annealing count increases at ten qubits, indicating possible inefficiencies at this problem size. In contrast, the variational algorithm displays a moderate increase, showing a more stable scaling performance. This is particularly remarkable at 11 qubits where the number of iterations decreases.

In conclusion, the numerical results on this type of MILP show the efficiency of our hybrid BD approach in comparison

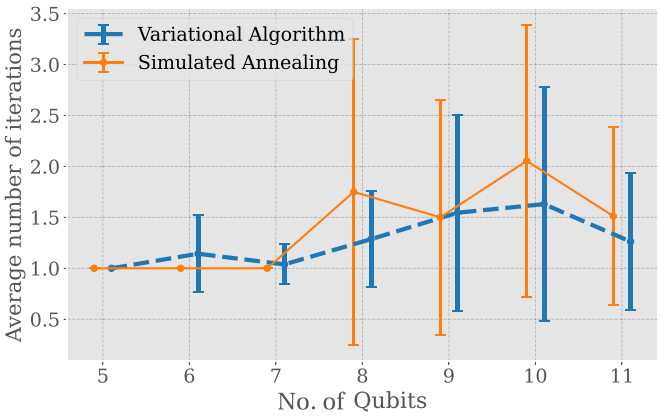

FIG. 6. Average number of BD iterations per number of qubits.

to the fully classical Benders method with simulated annealing. The results affirm that the hybrid BD generates a higher percentage of feasible solutions for various qubit counts and maintains a closer distance to optimal solutions. Moreover, the hybrid BD algorithm shows a more stable performance with respect to the number of iterations, especially with a larger number of qubits. These results show the potential of the hybrid BD algorithm with neutral atoms to efficiently handle larger and more complex problems, even within the current limitations of quantum computational resources. Future work should aim to extend these investigations beyond the 11-qubit threshold as advancements in quantum computing technology become available. It is also important to note that extending these preliminary tests to more complex MILP scenarios may reveal instances where the classical BD with simulated annealing outperforms the hybrid BD with the variational algorithm, due to qubit limitations and/or algorithmic convergence issues. In such cases, further research would be conducted. This will be discussed in Sec. VIII.

VIII. CONCLUSION

In this study we assisted classical BD with neutral-atom computation, to address MILP problems. To this end, we designed a hybrid classical-quantum algorithm. We developed an automated procedure to transform the MP into a QUBO formulation. We also presented a heuristic for register embedding. In addition, we implemented a variational algorithm for pulse shaping. A POC has shown our method applicability, and preliminary numerical results have demonstrated the efficiency of our hybrid framework, which outperforms fully classical BD techniques. While this research shows the potential of hybrid quantum-classical BD algorithms, supported by neutral-atom computation, in addressing MILPs, it opens new avenues for future advancements in this area. It should be noted that while the results presented in this paper are encouraging, they apply only to a specific set of small instances. Expanding these findings and extending the scale of application are not included in the scope of this initial POC work.

Looking to the future, our objective is to scale and diversify our instances. We are aware of the potential challenges of this step, especially in terms of qubit resource limitations and algorithmic convergence. In case we encounter resource limitations in terms of the number of qubits, one promising avenue is to set a maximum number of qubits as a computational threshold and implement an iterative process that aims to guarantee the quality of the solution while respecting the qubit count limitation. More precisely, within each iteration, one can evaluate the current penalty terms and their contribution to the solution quality. Any penalty term not significantly influencing the solution can be removed to free up computational resources. This allows for the introduction of new penalty terms that may further refine the solution. This process should be repeated until the solution converges. By managing the number of qubits in this manner, we hope to achieve a balance between qubit utilization and solution quality. This prospective approach has the potential to enable us to solve complex problems within a limited quantum environment.

In order to address the potential convergence issues of the algorithm, we can consider multiple MP solutions in each iteration. To this end, we can build on the interesting work of [55], which is based on the concepts of multicuts introduced in [34]. This method involves generating multiple solutions for the MP and then selecting a specific subset of Benders cuts to not surcharge the MP. By solving a set covering problem, we can identify the minimal set of constraints necessary to exclude all suboptimal or infeasible MP solutions. This approach can speed up the algorithm convergence and at the same time reduce the number of qubits needed. Moreover, the task of identifying the optimal subset of constraints is equivalent to solving a MIS problem, making it well suited for execution on a neutral-atom QPU.

ACKNOWLEDGMENTS

We thank Constantin Dalyac for insightful discussions and Anna Joliot for her helpful outcome in some experimentation.

[1] V. T. Paschos, *Applications of Combinatorial Optimization* (Wiley, New York, 2014), Vol. 3.
[2] M. Bénichou, J.-M. Gauthier, P. Girodet, G. Hentges, G. Ribière, and O. Vincent, Experiments in mixed-integer linear programming, Math. Program. **1**, 76 (1971).
[3] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*, 5th ed. (Springer, Berlin, 2020).
[4] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey, Cutting planes in integer and mixed integer programming, Discrete Appl. Math. **123**, 397 (2002).
[5] J. F. Benders, Partitioning procedures for solving mixed-variables programming problems, Numer. Math. **4**, 238 (1962).
[6] F. Vanderbeck and M. W. Savelsbergh, A generic view of Dantzig–Wolfe decomposition in mixed integer programming, Operat. Res. Lett. **34**, 296 (2006).
[7] R. Rahmaniani, T. G. Crainic, M. Gendreau, and W. Rei, The benders decomposition algorithm: A literature review, Eur. J. Operat. Res. **259**, 801 (2017).
[8] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028.

[9] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, New J. Phys. **18**, 023023 (2016).
[10] G. Nannicini, Performance of hybrid quantum-classical variational heuristics for combinatorial optimization, Phys. Rev. E **99**, 013304 (2019).
[11] J. Li, X. Yang, X. Peng, and C.-P. Sun, Hybrid quantum-classical approach to quantum optimal control, Phys. Rev. Lett. **118**, 150503 (2017).
[12] W. da Silva Coelho, L. Henriet, and L.-P. Henry, Quantum pricing-based column-generation framework for hard combinatorial problems, Phys. Rev. A **107**, 032426 (2023).
[13] Z. Zhao, L. Fan, and Z. Han, in *Proceedings of the IEEE Wireless Communications and Networking Conference, Austin, 2022* (IEEE, Piscataway, 2022), pp. 2536–2540.
[14] F. Gao, D. Huang, Z. Zhao, W. Dai, M. Yang, and F. Shuang, Hybrid quantum-classical general benders decomposition algorithm for unit commitment with multiple networked microgrids, arXiv:2210.06678.
[15] C.-Y. Chang, E. Jones, Y. Yao, P. Graf, and R. Jain, On hybrid quantum and classical computing algorithms for mixed-integer programming, arXiv:2010.07852.
[16] L. Fan and Z. Han, Hybrid quantum-classical computing for future network optimization, IEEE Network **36**, 72 (2022).
[17] M.-T. Nguyen, J.-G. Liu, J. Wurtz, M. D. Lukin, S.-T. Wang, and H. Pichler, Quantum optimization with arbitrary connectivity using Rydberg atom arrays, PRX Quantum **4**, 010316 (2023).
[18] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, Superconducting qubits: Current state of play, Annu. Rev. Condens. Matter Phys. **11**, 369 (2020).
[19] I. Pogorelov, T. Feldker, C. D. Marciniak, L. Postler, G. Jacob, O. Krieglsteiner, V. Podlesnic, M. Meth, V. Negnevitsky, M. Stadler *et al.*, Compact ion-trap quantum computing demonstrator, PRX Quantum **2**, 020343 (2021).
[20] C. Antón, J. C. Loredo, G. Coppola, H. Ollivier, N. Viggianiello, A. Harouri, N. Somaschi, A. Crespi, I. Sagnes, A. Lemaitre *et al.*, Interfacing scalable photonic platforms: Solid-state based multi-photon interference in a reconfigurable glass chip, Optica **6**, 1471 (2019).
[21] F. Glover, G. Kochenberger, and Y. Du, A tutorial on formulating and using QUBO models, arXiv:1811.11538.
[22] IBM, CPLEX optimizer, 2023, available at https://www.ibm.com/fr-fr/analytics/cplex-optimizer.
[23] Gurobi Optimization, Gurobi—the fastest solver, 2023, available at https://www.gurobi.com.
[24] D. Bertsimas and J. Tsitsiklis, Simulated annealing, Stat. Sci. **8**, 10 (1993).
[25] G. Desaulniers, J. Desrosiers, and M. M. Solomon, *Column Generation* (Springer Science+Business Media, New York, 2006), Vol. 5.
[26] D. T. Do, N. Trieu, and D. T. Nguyen, Quantum-based distributed algorithms for edge node placement and workload allocation, arXiv:2306.01159.
[27] R. Wille, R. Van Meter, and Y. Naveh, in *Proceedings of the 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE), Florence* (IEEE, Piscataway, 2019), pp. 1234–1240.
[28] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, Found. Trends Mach. Learn. **3**, 1 (2010).
[29] J. Naoum-Sawaya and S. Elhedhli, An interior-point benders based branch-and-cut algorithm for mixed integer programs, Ann. Operat. Res. **210**, 33 (2013).
[30] T. G. Crainic, M. Hewitt, and W. Rei, *Partial Decomposition Strategies for Two-Stage Stochastic Integer Programs* (CIRRELT, Montreal, 2014), Vol. 88.
[31] Y. Yang and J. M. Lee, A tighter cut generation strategy for acceleration of benders decomposition, Comput. Chem. Eng. **44**, 84 (2012).
[32] C. I. Fábián and Z. Szőke, Solving two-stage stochastic programming problems with level decomposition, Comput. Manag. Sci. **4**, 313 (2007).
[33] A. J. Rubiales, P. A. Lotito, and L. A. Parente, Stabilization of the generalized benders decomposition applied to short-term hydrothermal coordination problem, IEEE Lat. Am. Trans. **11**, 1212 (2013).
[34] N. Beheshti Asl and S. MirHassani, Accelerating benders decomposition: Multiple cuts via multiple solutions, J. Comb. Optim. **37**, 806 (2019).
[35] N. Franco, T. Wollschläger, B. Poggel, S. Günnemann, and J. M. Lorenz, Efficient MILP decomposition in quantum computing for ReLU network robustness, arXiv:2305.00472.
[36] R. E. Tarjan and A. E. Trojanowski, Finding a maximum independent set, SIAM J. Comput. **6**, 537 (1977).
[37] B. N. Clark, C. J. Colbourn, and D. S. Johnson, Unit disk graphs, Discrete Math. **86**, 165 (1990).
[38] S. Stastny, H. P. Büchler, and N. Lang, Functional completeness of planar Rydberg blockade structures, Phys. Rev. B **108**, 085138 (2023).
[39] J. J. Sakurai and J. Napolitano, *Modern Quantum Mechanics*, 2nd ed. (Addison-Wesley, Reading, 2011), Chap. 5 provides a detailed discussion of the adiabatic theorem in quantum mechanics.
[40] W. da Silva Coelho, M. D'Arcangelo, and L.-P. Henry, Efficient protocol for solving combinatorial graph problems on neutral-atom quantum processors, arXiv:2207.13030.
[41] L. Novo, S. Chakraborty, M. Mohseni, and Y. Omar, Environment-assisted analog quantum search, Phys. Rev. A **98**, 022316 (2018).
[42] L. Henriet, L. Beguin, A. Signoles, T. Lahaye, A. Browaeys, G.-O. Reymond, and C. Jurczak, Quantum computing with neutral atoms, Quantum **4**, 327 (2020).
[43] J. F. Benders, Partitioning procedures for solving mixed-variables programming problems, Comput. Manag. Sci. **2**, 3 (2005).
[44] M. Balinski and A. W. Tucker, Duality theory of linear programs: A constructive approach with applications, SIAM Rev. **11**, 347 (1969).
[45] A. Schrijver, *Theory of Linear and Integer Programming* (Wiley, New York, 1998).
[46] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization* (Athena, Belmont, 1997), Vol. 6.
[47] M. Ayodele, in *European Conference on Evolutionary Computation in Combinatorial Optimization*, edited by L.

Pérez Cáceres and S. Verel, Lecture Notes in Computer Science Vol. 13222 (Springer, Cham, 2022), pp. 159–174.

[48] A. Verma and M. Lewis, Penalty and partitioning techniques to improve performance of QUBO solvers, Discrete Optim. **44**, 100594 (2022).

[49] M. D. García, M. Ayodele, and A. Moraglio, in *Proceedings of the Genetic and Evolutionary Computation Conference Companion, Kyoto, 2018*, edited by H. Aguirre (ACM, New York, 2022), pp. 184–187.

[50] P. Prettenhofer and G. Louppe, Gradient boosted regression trees in scikit learn, available at https://www.slideshare.net/slideshow/gradient-boosted-regression-trees-in-scikit-learn-gilles-louppe/32811158.

[51] T. Albash and D. A. Lidar, Adiabatic quantum computation, Rev. Mod. Phys. **90**, 015002 (2018).

[52] D-Wave Embedding, available at https://docs.ocean.dwavesys.com/en/stable/concepts/embedding.html.

[53] Pulser Development Team, Pulser documentation, available at https://pulser.readthedocs.io/en/stable/.

[54] scikit-optimize Developers, scikit-optimize: Sequential model-based optimization in Python, available at https://scikit-optimize.github.io/stable/.

[55] N. G. Paterakis, Hybrid quantum-classical multi-cut benders approach with a power system application, Comput. Chem. Eng. **172**, 108161 (2023).