ESAIM: PROCEEDINGS AND SURVEYS, Vol. ?, 2026, 1-10
Editors: Will be set by the publisher

QUANTUM APPROACH FOR ELECTRICAL SYSTEM MAINTENANCE SCHEDULING PROBLEM

Qinyan Zhou[1], Djahou N. Tognon[2], Sofiane Chalal[3], Noureddine Henka[4], Sami Tazi[5], Fikri Hafid[6] and Titouan Carette[7]

Abstract. Maintenance scheduling is essential to ensure the reliability of power transmission systems while limiting operational risk and respecting technical and resource constraints. Increasing system complexity due to asset aging and renewable integration challenges classical optimization approaches traditionally used by transmission system operators. This work focuses on the optimization stage of a risk-based maintenance planning framework, where predefined risk indicators guide the construction of feasible schedules.

After discussing the limitations of classical mixed-integer programming formulations in terms of scalability and computational effort, we investigate the applicability of quantum optimization methods to the maintenance scheduling problem. In particular, we explore the use of the Quantum Approximate Optimization Algorithm (QAOA) and Grover Adaptive Search to minimize operational risk under realistic constraints. While current quantum hardware limits practical deployment, this study provides insights into the potential of quantum approaches as complementary tools for future large-scale power system maintenance planning.

Résumé. La planification des opérations de maintenance est essentielle pour garantir la fiabilité des réseaux de transport d'électricité tout en limitant le risque opérationnel et en respectant les contraintes techniques et de ressources. La complexité croissante des systèmes électriques, due au vieillissement des actifs et à l'intégration accrue des énergies renouvelables, met à l'épreuve les approches d'optimisation classiques traditionnellement utilisées par les gestionnaires de réseau de transport. Ce travail se concentre sur l'étape d'optimisation d'un cadre de planification de maintenance fondé sur le risque, dans lequel des indicateurs de risque prédéfinis guident la construction de plannings réalisables.

Après avoir discuté les limites des formulations classiques en programmation linéaire en nombres entiers mixtes en termes de passage à l'échelle et de charge computationnelle, nous étudions l'applicabilité de méthodes d'optimisation quantique au problème de planification de la maintenance. En particulier, nous explorons l'utilisation de l'algorithme d'optimisation approximative quantique (QAOA) et de la recherche adaptative de Grover afin de minimiser le risque opérationnel sous des contraintes réalistes. Bien que les limitations actuelles du matériel quantique restreignent les applications pratiques, cette étude apporte des éléments de réflexion sur le potentiel des approches quantiques comme outils complémentaires pour la planification future de la maintenance des grands réseaux électriques.

[1] UMA, ENSTA Paris, Polytechnic Institute of Paris, France
[2] Université d'Orléans, Université de Tours, CNRS, IDP, UMR 7013, Orléans, France
[3] L2S, CentraleSupélec, Université Paris-Saclay, France
[4] R&D, RTE France
[5] R&D, RTE France
[6] R&D, RTE France, Centre Borelli, ENS Paris-Saclay
[7] LIX, CPHT, CNRS, Inria, École polytechnique, Institut Polytechnique de Paris, Palaiseau, France

Introduction

Maintaining a reliable electricity supply is one of the primary responsibilities of transmission system operators such as RTE (Réseau de Transport d'Electricité). This reliability depends not only on real-time operations but also on the careful planning of maintenance activities that ensure the long-term health of the grid. Some interventions can be carried out while the lines remain energized, whereas others require temporary disconnection. Each planned outage must therefore be scheduled so that system security and electricity delivery remain uncompromised. Although the network is generally designed to withstand unexpected contingencies, simultaneous breakdowns or poorly coordinated outages can still lead to severe disruptions, including large-scale blackouts.

The increasing complexity of the power system makes this task even more challenging. The ongoing energy transition, the aging of network assets, and the large-scale integration of renewable generation profoundly affect grid operation and flexibility. As a result, maintenance plans that were feasible under past operating conditions may no longer be realistic in the future. Anticipating such evolutions is essential: understanding how future grid configurations constrain maintenance planning helps RTE adapt its operational and investment strategies today, rather than reactively in the years to come.

To support this forward-looking approach, RTE has developed a three-stage framework. First, potential risks are quantified under various future scenarios. Second, these risk measures are incorporated into optimization models to identify feasible and efficient maintenance schedules. Finally, the resulting plans are validated through simulation and expert assessment.

The present study focuses on the optimization stage of this framework. Given predefined risk values, we aim to compute maintenance schedules that minimize operational risk while respecting technical, temporal, and resource constraints.

1. Related Works

1.1. Classical optimization approaches for power system maintenance scheduling

Maintenance scheduling in power transmission networks is a well-established research topic and is typically formulated as a combinatorial optimization problem subject to technical, temporal, and security constraints. Classical approaches model maintenance scheduling using mixed-integer linear programming (MILP), in which outage decisions, maintenance windows, resource capacities, and network reliability requirements are encoded as binary and linear constraints. For instance, Froger et al. provide an extensive survey of maintenance scheduling problems in the electricity industry, covering both generating unit maintenance and transmission maintenance, and discuss MILP models alongside heuristic and metaheuristic solution methods [7].

MILP formulations offer exact or bounded solutions under deterministic settings and are useful for integrating maintenance decisions with operational constraints such as generation dispatch and network flow feasibility. A concrete example of a network-focused MILP for transmission maintenance scheduling is presented in a doctoral study that formulates the Transmission Maintenance Scheduling (TMS) problem with connectivity constraints and use of decomposition algorithms to manage computational effort [15].

A particularly relevant recent contribution is the PSCC 2020 study by [4], which proposes a three-stage, risk-based framework. In this framework, operational risk metrics are first quantified under multiple future grid scenarios and then integrated into a MILP scheduling model featuring intervention duration, resource consumption, and exclusion constraints between conflicting outages.

These classical approaches exhibit several key limitations:

Scalability. MILP models become computationally intractable as the number of assets, decision periods, and scenario counts increases due to the exponential growth in binary variables and constraint interactions. Even with advanced solvers and decomposition techniques, solving large power system maintenance scheduling problems can be costly in time and memory. While classical MILP work focuses on solving instances of moderate size, large-scale transmission networks remain challenging to optimize directly with off-the-shelf approaches [15].

Handling uncertainty. Extensions that explicitly incorporate operational uncertainty — e.g., in future load, generation, or failure probabilities — often adopt stochastic MILP formulations with scenario trees or chance constraints that significantly increase problem size. For instance [13] formulate a two-stage stochastic mixed-integer program for integrated maintenance and operations scheduling under failure scenarios, which underscores both the expressivity and the computational burden of uncertainty-aware optimization.

Heuristic and metaheuristic dependencies. To address computational tractability for large-scale or highly complex problems, many researchers have turned to heuristic and metaheuristic methods. For example, teaching-learning-based optimization (TLBO) and other population-based methods have been applied to integrated maintenance scheduling problems combining generation and transmission factors, where exact methods struggle with large solution spaces [1]. Metaheuristics provide flexible, scalable alternatives but lack optimality guarantees and may have difficulty satisfying strict hard constraints without additional mechanisms.

In operational environments, the scalability and robustness challenges of classical optimization often lead transmission system operators to couple mathematical models with expert judgment, simplified heuristics, or decomposition strategies. As power systems become increasingly complex — due to higher renewable integration, evolving reliability requirements, and larger grid footprints — these classical methods face growing strain, motivating exploration of alternative computational paradigms.

1.2. **Quantum optimization methods for combinatorial scheduling problems**

As classical methods confront scalability and robustness limits, quantum computing has emerged as a potential alternative for tackling large-scale combinatorial optimization problems. Among quantum optimization algorithms, the Quantum Approximate Optimization Algorithm (QAOA) introduced by Farhi et al. aims to approximate solutions to binary optimization problems by leveraging quantum superposition and interference [6].

QAOA's hybrid quantum–classical nature makes it appealing for near-term noisy intermediate-scale quantum (NISQ) devices, and it has been investigated for problems related to power systems, such as simplified unit commitment or network reconfiguration (see e.g., Ajagekar et al. and Egger et al.) [2].

Another class of quantum methods builds on Grover's search algorithm, which offers a theoretical quadratic speedup for unstructured search problems. When adapted to optimization via iterative or adaptive search strategies, Grover-based methods can be used to explore large combinatorial spaces more efficiently than classical enumeration [8, 11].

1.3. **Limitations and outlook**

Despite their theoretical promise, current quantum optimization approaches are not yet ready for direct deployment in operational power system scheduling. Present hardware limitations—such as limited qubit counts, noise, and circuit depth constraints—restrict the size and complexity of problems that can be practically encoded and solved with meaningful solution quality. Furthermore, mapping real-world maintenance scheduling constraints and risk measures into quantum cost functions or oracles remains an active research challenge.

Nevertheless, quantum algorithms represent a fundamentally different computational paradigm that may help overcome combinatorial scaling issues in the longer term. By exploring quantum optimization methods today, researchers and system operators can identify problem structures amenable to quantum advantage and prepare for future advancements in quantum hardware and algorithm design.

In this context, the present work extends classical risk-aware maintenance formulations and investigates the applicability of QAOA and Grover Adaptive Search to compute maintenance schedules that minimize operational risk under technical, temporal, and resource constraints.

2. Optimization Problem: Mean Cost

In this section, we first introduce the problem, notation and input parameters of the Stochastic Scheduling Problem (SSP) as applied to maintenance operations in electrical transmission grids. For more detailed technical specifications and the practical context faced by RTE, the French electricity transmission operator, we refer the reader to the ROADEF Challenge [14]. We then formulate the optimization problem as a Mixed-Integer Linear Program (MILP). Finally, we derive the corresponding Quadratic Unconstrained Binary Optimization (QUBO) formulation of the original problem.

2.1. **Inputs and Notations**

We consider a planning horizon discretized into T time steps. Depending on the desired temporal resolution, each time step may represent one day ($T = 365$ per year) or one week ($T = 53$ per year). Let I denote a finite set of maintenance activities to be scheduled over this horizon. These activities, referred to as interventions in the challenge, differ in both duration and required resources. Each intervention $i \in I$ must be scheduled exactly once within the planning horizon and is assumed to be non-preemptive, i.e., once it starts, it continues (excluding non-working days) until completion. We denote by $\Delta_{i,t}$, the duration of intervention i if it starts at time t. As previously mentioned, the duration may vary between interventions, i.e., $\forall i \in I, \exists i', \Delta_{i,t} \neq \Delta_{i',t}$. When an intervention $i \in I$ starts at time $start_i \in H$, it remains active until its completion; such interventions are referred to as active interventions. Finally, all interventions must be scheduled within the planning horizon so that each is completed by the end of the period. We refer to this requirement as the completion constraint.

Different interventions may require distinct skills or teams of workers, each specialized in specific types of maintenance tasks. The available resources are limited and may vary over time. Let C denote the set of resources. The quantity of resource $c \in C$ used at time $t \in H$ by intervention $i \in I$, starting at time $start_i \in T$, is denoted by $r_{c,t}^{i,start_i}$. For each resource $c \in C$ and each time step $t \in H$, the total resource consumption is constrained by lower bounds u_t^c and upper bounds l_t^c representing the minimum and maximum available capacities, respectively.

It is also important to note that when an intervention is performed, the electrical component on which it occurs must, in most cases, be temporarily disconnected from the grid. For instance, when a transmission line undergoes maintenance, it must generally be taken out of service. Such disconnections can weaken the overall grid structure, as the power flow previously carried by the disconnected line is redistributed across alternative paths, potentially overloading other lines. This situation introduces operational risks for RTE, given that the transmission system operator's operational doctrine requires that the power flow on each line remain below its thermal limit—even under contingency conditions, i.e., when any single line is disconnected.

Although the probability of such contingencies is low, they can result from extreme weather events or accidental failures, and if not properly managed, may trigger cascading failures leading to large-scale blackouts. Consequently, any planned disconnection for maintenance must ensure that the loss of another line does not compromise grid security. The normal operating state of the grid is denoted by N, while the state simulated under the loss of any single line is referred to as $N-1$. Therefore, the scheduling of interventions must account for both the risk associated with the N and $N-1$ network states.

To quantify the operational risk associated with maintenance activities, RTE performs extensive simulations of both N and $N-1$ network states under multiple scenarios representing different consumption and production profiles across time steps. Let S_t,denote the set of scenarios at time t, and $S = \cup_{t \in H} S_t$ the set of all scenarios. The risk, expressed in euros, associated with performing intervention $i \in I$ starting at time $start_i$ under scenario $s \in S$ and evaluated at time $t \in H$, is denoted by $\text{risk}_{i,\text{start}_i}^{s,t}$,. For certain interventions, this risk varies seasonally—for example, it is typically lower during summer, when electricity demand is reduced, than during winter peaks.

Furthermore, certain interventions cannot be scheduled simultaneously. For instance, two parallel transmission lines supplying the same consumers cannot both be disconnected at once, as this would interrupt service. Similarly, other lines, even if not in parallel, may be interdependent and thus restricted from concurrent maintenance. These mutual exclusion relationships are represented by a set $\mathcal{E}$ of triplets (i, i', t), indicating that interventions i and i' cannot be active simultaneously at time t.

2.2. MILP formulation

A maintenance schedule can be represented as a list of starting times for all interventions. In this work, the solution is modeled by a binary matrix $x\{0,1\}^{|I|\times|H|}$, defined as follows:

- $x_{i,t} = 1$ if intervention i is active at time t, 0 otherwise, size: $|I| \times |H|$ (need to moved to QUBO part)
- $y_{i,t} = 1$ if intervention i starts at time t, 0 otherwise, size: $|I| \times |H|$ (need to moved to QUBO part)
- $z_{i,k} = k$-th binary digit of the starting time of i (useful for encoding), size: $|I| \times \lceil \log_2 |H| \rceil$ (need to be moved to QUBO part)

This binary formulation enables a direct connection between the scheduling decision variables, the operational constraints, and the risk-based objective function.

The objective function evaluates a given planning solution by minimizing the expected operational risk over all possible scenarios. Recall that the risk is a random variable for which only a finite number of realizations are available. Hence, the original formulation corresponds to a stochastic linear programming problem. For simplicity—and because our goal is to implement the scheduling problem on a quantum computer—we consider the expected (mean) value of the risk across all scenarios. The mean risk thus represents the expected financial cost over the planning horizon, capturing both temporal and scenario-based variability:

$$\text{Mean Cost} = \frac{1}{T} \sum_{t\in H} \frac{1}{|S_t|} \sum_{s\in S_t} \sum_{i\in I_t} \text{risk}^{s,t}_{i,\text{start}_i}.$$

This quantity measures the average financial exposure associated with a given maintenance plan. Minimizing it yields schedules that optimally distribute interventions over time to reduce the overall operational risk of the grid.

The constraints of the problem can be organized as follows:

2.2.1. *Planning Horizon*

The planning is established over a one-year period, discretized into T time steps:

$$H = \{1, \dots, T\}, \quad T \in \mathbb{N}.$$

Depending on the required temporal resolution, each time step may represent one day ($T = 365$) or a week ($T = 53$).

2.2.2. *Interventions*

The set of interventions is denoted by I. Each intervention $i \in I$ must be scheduled exactly once and is **non-preemptive**, i.e., once started, it continues (excluding non-working days) until completion.

The set of maintenance interventions is denoted by I. Each intervention $i \in I$ must be scheduled exactly once within the planning horizon and is assumed to be non-preemptive, i.e., once started, it continues (excluding non-working days) until completion.

- **Duration:** $\Delta_{i,t}$ denotes the duration of intervention i if it starts at time t. The duration may vary with t (e.g. shorter in summer, longer in winter).
- **Active interventions:**

$$I_t = \{i \in I \mid \text{start}_i \le t < \text{start}_i + \Delta_{i,\text{start}_i}\}.$$

- **Completion constraint:**

$$\text{start}_i + \Delta_{i,\text{start}_i} \le T + 1.$$

2.2.3. *Resources*

Each intervention consumes specific **resources** (e.g.,workforce teams or specialized equipment) drawn from a finite set C.

- $r_{c,t}^{i,\text{start}_i}$: workload of resource c required by intervention i at time t when started at start_i.
- u_t^c : maximum available amount of resource c at time t.
- l_t^c : minimum required usage of resource c at time t.

Resource availability is enforced by the following constraint:

$$l_t^c \leq \sum_{i \in I_t} r_{c,t}^{i,\text{start}_i} \leq u_t^c, \quad \forall c \in C,\ t \in H.$$

2.2.4. *Exclusion Constraints*

Certain pairs of interventions cannot overlap in time because because of geographical proximity or electrical dependencies. Let $\mathcal{E}$ denote the set of exclusion triplets (i_1, i_2, t), meaning that if i_1 is active at t, then i_2 cannot be.

$$(i_1, i_2, t) \in \mathcal{E} \Rightarrow (i_1 \in I_t \Rightarrow i_2 \notin I_t).$$

2.2.5. *Problem Definition*

The objective of the planning problem is to determine the optimal starting times of a set of maintenance interventions over a discrete time horizon, to minimize the **mean risk** across all scenarios and time steps while satisfying operational, resource and exclusion constraints.

Formally, the optimization problem is defined as follows:

$$\begin{aligned}
\min_{\text{start}_i} \quad & \frac{1}{T} \sum_{t \in D} \left(\frac{1}{|S_t|} \sum_{s \in S_t} \sum_{i \in I_t} \text{risk}_{i,\text{start}_i}^{s,t} \right) = \min_{\text{start}_i} \frac{1}{T} \sum_{t \in D} \sum_{i \in I_t} \overline{\text{risk}^t}_{i,\text{start}_i} && \text{(1a)}\\
\text{s.t.} \quad & \text{start}_i \in H, \quad \forall i \in I && \text{(1b)}\\
& \text{start}_i + \Delta_{i,\text{start}_i} \leq T + 1, \quad \forall i \in I && \text{(1c)}\\
& l_t^c \leq \sum_{i \in I_t} r_{c,t}^{i,\text{start}_i} \leq u_t^c, \quad \forall c \in C, \forall t \in H && \text{(1d)}\\
& (i_1, i_2, t) \in \mathcal{E} \Rightarrow (i_1 \in I_t \Rightarrow i_2 \notin I_t) && \text{(1e)}
\end{aligned}$$

We define the average risk of intervention i starting at time $start_i$ and evaluated at time t as:

$$\overline{\text{risk}^t}_{i,\text{start}_i} = \frac{1}{|S_t|} \sum_{s \in S_t} \text{risk}_{i,\text{start}_i}^{s,t}$$

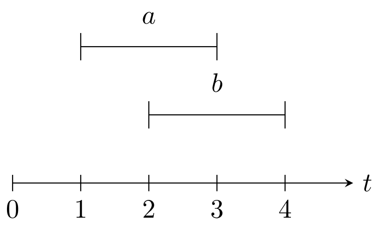
This formulation represents a stochastic optimization problem, where the mean value serves as an estimator of the expected risk. In this work, we focus on minimizing the mean risk over all scenarios for simplicity and to enable implementation on quantum hardware. Alternative stochastic optimization approaches—such as quantile-based optimization or worst-case (robust) optimization—could also be employed, depending on the risk tolerance of the operator.

2.2.6. *Illustrative Example*

Let $I = \{a, b\}$ be two intervention to be scheduled, each lasting for 2 time steps. assume that the available resources are sufficient to allow both interventions to be active simultaneously. The planning horizon $H = \{0, 1, 2, 3, 4\}$.

A possible planning representation is given below:

t	I_t
0	$\emptyset$
1	$\{a\}$
2	$\{a, b\}$
3	$\{a, b\}$
4	$\{b\}$



This example illustrates the overlap between interventions a and b across the time horizon, as well as their corresponding active intervals. The proposed schedule satisfies all previously defined operational and resource constraints.

2.3. **QUBO Formulation**

Once the scheduling problem is well-defined and expressed mathematically, there is no guarantee that quantum computers can solve it efficiently. For most quantum computing approaches, the problem must be reformulated as a QUBO model. This requires expressing the objective as:

$$\min_{x \in \{0,1\}^n} x^T Q x$$

where:

- $Q \in R^{n \times n}$ is a symmetric real-valued matrix
- $x \in \{0, 1\}^n$ is a vector of binary decision variables

The structure of Q is critical because it influences the physical implementation on quantum hardware. In certain architectures, Q encodes the interaction between qubits (or particles). For example:

- On Pasqal machine, performance improves when Q exhibits a unidisk topology.
- On D-Wave systems, the topology of Q should align perfectly with the machine's native connectivity graph.

For the scheduling optimization problem, let: $I = \{\text{interventions}\}, H = \{1, \ldots, T\}, |I| = n, |H| = T$. An intervention $i \in I$ starting at time $t \in H$, has a duration $\Delta_{i,t}$, and denotes $x_{i,t}$ a $\{0, 1\}$-variable

$$x_{i,t} = \begin{cases} 1 & \text{if intervention } i \text{ start at time } t \\ 0 & \text{otherwise.} \end{cases}$$

Therefore, for each intervention, we introduce a binary variable indicating its possible starting time across all time slots. The decision vector is defined as: $x = [x_{1,1}, x_{1,2}, \ldots, x_{1,T}, x_{2,1}, x_{2,2}, \ldots, x_{n,1}, \ldots, x_{n,T}]^T$. This representation imposes an additional constraint: for any $i \in I$, exactly one starting time must be selected. Equivalently, this can be expressed as:

$$\forall i \in I,\ \exists! t \in H,\ \text{such that}\ \ x_{i,t} = 1 \Leftrightarrow \sum_{\tau \in H} x_{i,\tau} = 1,\ \forall i \in I$$

Following this formulation, the cost function and constraints will be rewritten to incorporate the binary structure of x.

2.3.1. *Cost Function*

In the cost function, the indicator for the start of each intervention is now represented by the binary variable $x_{i,t}$, which specifies whether intervention i begins at time t. Therefore:

$$\begin{aligned}
\sum_{t\in H}\sum_{i\in I}\overline{\text{risk}^t}_{i,\text{start}_i} &= \sum_{t\in H}\sum_{i\in I}\sum_{t'\in H} x_{i,t'}1_{t\in[t',t'+\Delta_{i,t'}]}\overline{\text{risk}^t}_{i,\text{start}_i}\\
&= \sum_{i\in I}\sum_{t'\in H} x_{i,t'}\Big(\sum_{t\in[t',t'+\Delta_{i,t'})}\overline{\text{risk}^t}_{i,\text{start}_i}\Big)\\
&= \sum_{i\in I}\sum_{t'\in H} x_{i,t'}\tilde{Q}(i,t')\\
&= x^T\tilde{Q}x
\end{aligned}$$

Thus, the cost function can be expressed in QUBO form, enabling direct mapping to quantum optimization frameworks.

2.4. **Constraints**

The QUBO formulation does not natively support explicit constraints. A common approach is to relax constraints into the objective function using penalty terms weighted by factors that balance feasibility and optimality. This requires expressing each constraint in a quadratic form and incorporating it into the cost function.

Unique intervention: Under the binary formulation, each intervention must start exactly once. The constraint is written as:

$$\sum_{t'\in H} x_{i,t'} = 1, \ \ \forall i \in I$$

This is a linear constraint. Any linear expression can be represented in QUBO form by embedding its coefficients on the diagonal, since: $a^Tx = x(\text{diag}(a))x^T$ Thus, the relaxed penalty term becomes:

$$\lambda_1 \sum_{i\in I}\left(\sum_{t'\in H} x_{i,t'} - 1\right)^2$$

where $\lambda_1 > 0$ controls the strength of the constraint relative to the original cost function.

Don't start late: To ensure that all interventions are scheduled within the reserved time horizon, we introduce a constraint that prevents late starts. Formally: $\text{start}_i + \Delta_{i,\text{start}_i} \leq T+1$. In terms of QUBO variables, we have $x_{i,\text{start}_i} = 1$, and $x_{i,t'} = 0, \forall t' \in H\backslash\{\text{start}_i\},$. Let us denote $\tau = \text{start}_i$, then

$$\tau = \sum_{\tau'\in H}\tau' x_{i,\tau'},$$

and the constraint becomes:

$$\Delta_{i,\tau} + \underbrace{\tau}_{\sum_{\tau'\in H}\tau' x_{i,\tau'}} \leq T+1$$

which is exactly,

$$\sum_{\tau'\in H}(\tau + \Delta_{i,\tau} - T)x_{i,\tau} \leq 1 \tag{2}$$

However, instead of encoding this as a QUBO penalty term, we apply constraint propagation during variable construction. Specifically, for each intervention i, any start time that would cause the intervention to exceed the planning horizon is excluded. Thus: $\text{start}_i + \Delta_{i,\text{start}_i} \leq T+1, \quad \forall i \in I$. This implies for each intervention i, starting times that would cause it to exceed the total time must be disallowed. In terms of the binary variables $x_{i,t}$, we can have: $x_{i,t} = 0 \quad \text{for all } t > T + 1 - \Delta_{i,t}$. This means we only define binary variables for valid start times:

$$\mathcal{X} = \{(i,t) \in I \times H \mid t \leq T + 1 - \Delta_{i,t}\}$$

By removing invalid variables from the QUBO model, this constraint is satisfied by construction, reducing the number of bits and improving encoding efficiency.

The constraint of resource: The resource constraint is expressed as an inequality between lower and upper bounds

$$l_t \leq \sum_{i \in I_t} r_t^{i,\text{start}_i} \leq u_t, \quad \forall t \in H$$

Since QUBO does not directly handle inequalities, we introduce slack variables to convert them into equalities. These slack variables saturate the inequality until it becomes an equality. Thus, we rewrite the constraint as:

$$\sum_{i \in I_t} r_t^{i,\text{start}_i} + s_t^u = u_t, \qquad \sum_{i \in I_t} r_t^{i,\text{start}_i} - s_t^l = l_t$$

where $s_t^u, s_t^l \geq 0$ are integer slack variables. Each slack variable is encoded in binary as:

$$s_t^u = \sum_{k=0}^{K_t^{(u)}-1} 2^k z_{t,k}^{(u)}, \qquad s_t^l = \sum_{k=0}^{K_t^{(l)}-1} 2^k z_{t,k}^{(l)}, \quad z_{t,k}^{(u)}, z_{t,k}^{(l)} \in \{0,1\}$$

with

$$K_t^{(u)} = \lceil \log_2(u_t) \rceil + 1, \quad K_t^{(l)} = \left\lceil \log_2 \left(\sum_{i \in I_t} r_t^{i,\text{start}_i} - l_t \right) \right\rceil + 1,$$

The resource constraint penalty term is then:

$$\sum_{t \in H} \left[\lambda_t^{(u)} \left(\sum_{i \in I_t} r_t^{i,\text{start}_i} + \sum_{k=0}^{K_t^u - 1} 2^k z_{t,k}^{(u)} - u_t \right)^2 + \lambda_t^{(l)} \left(\sum_{i \in I_t} r_t^{i,\text{start}_i} - \sum_{k=0}^{K_t^l - 1} 2^k z_{t,k}^{(l)} - l_t \right)^2 \right]$$

General QUBO formulation: After transforming the cost function and all constraints into QUBO terms, the general QUBO formulation for the scheduling problem is given by:

$$\begin{aligned} \min_{x,z} \Big(& \sum_{i \in I} \sum_{\tau \in H} x_{i,\tau} \tilde{Q}(i,\tau) - \lambda_1 \sum_{\mathcal{E}} x_{i,\tau} x_{j,\tau'} + \lambda_2 \sum_{i \in I} \sum_{t + \Delta_{i,t} > T} x_{i,t} \\ & + \lambda_3 \sum_{i \in I} \Big(\sum_{t \in H} x_{i,t} - 1 \Big)^2 + \sum_{t \in H} \Big\{ \lambda_4^t \Big(\sum_{i \in I_t} r_t(i,\tau) + \sum_{k=0}^{K_t^{(u)}} 2^k z_{t,k}^u - u_t \Big)^2 + \lambda_5^t \Big(\sum_{i \in I_t} r_t(i,\tau) - \sum_{k=0}^{K_t^{(l)}} 2^k z_{t,k}^l - l_t \Big)^2 \Big\} \Big) \end{aligned} \tag{3}$$

where:

- λ_1 penalizes mutual exclusion violations,
- λ_2 penalizes late starts,
- λ_3 enforces unique start times per intervention,
- λ_4^t, λ_5^t enforce upper and lower bounds on resource usage at time t,

- x are binary scheduling variables,
- z are binary slack variables for resource constraints.

2.5. Penalty terms and slack variables in QUBO

The sensitivity analysis presented in Appendix A clearly illustrates that the validity and quality of the solutions strongly depend on the choice of the penalty coefficients. While sufficiently large values of λ enforce feasibility, they may also distort the optimization landscape and hinder the optimization of the original objective function. This empirical observation motivates a more general discussion on the role of penalty terms in QUBO formulations and the limitations of standard approaches based on slack variables.

We settled here for the most pragmatic approach, which led us to a directly testable QUBO formulation. To do so, we had to introduce slack variables, the standard technique to turn inequality constraints into equalities. However, this leads to a dramatic increase of the number of qubits that we can't usually afford both for classical simulations or tests on near term quantum computer. Thus, it is desirable to consider ways to handle inequalities in QUBO without relying on slack variables. Formally, the general problem is as follows: we want to minimize a function $f : X \to \mathbb{R}$ over a finite set X such that $h(x) \leq 0$ for a linear constraint function $h : X \to \mathbb{R}$, we are looking for a penalty function $g : X \to \mathbb{R}$ such that the minimum of $f + g$ is attained by a solution to the original problem. The most obvious way to ensure this is to impose:

$$\sup_{x,h(x)\leq 0} f(x) + g(x) < \inf_{x,h(x)>0} f(x) + g(x)$$

Thus, we are sure that it is always better to prefer a solution satisfying the constraint to a solution not satisfying them. A stronger condition is: $\sup_{x,h(x)\leq 0} f(x) + \sup_{x,h(x)\leq 0} g(x) < \inf_{x,h(x)>0} f(x) + \inf_{x,h(x)>0} g(x)$ or equivalently:

$$\sup_{x,h(x)\leq 0} f(x) - \inf_{x,h(x)>0} f(x) < \inf_{x,h(x)>0} g(x) - \sup_{x,h(x)\leq 0} g(x)$$

Notice that if the gap $\sup_{x,h(x)\leq 0} f(x) - \inf_{x,h(x)>0} f(x)$ is zero, then there is no continuous g satisfying the condition, happily, the fact that X is finite ensures that such gap always exists. The choice we would like to make is $g(x) = \Lambda H(h(x))$ where H is the Heaviside step function, defined as 1 if its input is non-negative and else 0, and Λ is a positive constant bigger than the gap $\sup_{x,h(x)\leq 0} f(x) - \inf_{x,h(x)>0} f(x)$.

Sadly, in the QUBO setting g is required to be quadratic in x, so we are reduced to try to find a quadratic function satisfying the condition on the relevant intervals, it is always possible when we have only one linear inequality but as soon as we have more they might even be no way to find such a quadratic function. An approach to tackle this problem has been proposed in [12], where they cut a Taylor expansion of a convenient function (in their case a negative exponential) at order two and hope the approximation is good enough to have the right properties. This approach is very relevant to the case of a PUBO (Polynomial Unconstrained Binary Optimisation) problem, when g is allowed to be any polynomial. Then we can obtain Taylor approximations as precise as we need, at the price of more entanglement in order to reach the higher degrees in the Hamiltonian. But with only order two available there is no theoretical reasons to expect good results in general.

The previous discussions only ensure the satisfaction of the constraints. In practice, it might happen that the penalty constants we choose are so big that the optimization algorithm only focuses on satisfying the constraints and becomes inefficient in optimizing the objective function. Sorting this out would require extensive benchmarks, preferably on simpler problems than the scheduling task that concerns us in this paper. Thus, in what follows, we will opt for an empirical approach to choose the penalty terms, at the price of obtaining from time to time solutions that violate the constraints.

3. Classical Solver and QAOA Solver

The motivation for employing both classical and quantum-approximate solvers arises from their complementary characteristics and roles in optimization research. In this section, the classical solver used is the `dimod.ExactSolver`, which performs an exhaustive enumeration of all binary configurations in the QUBO model. Although this brute-force approach is computationally expensive and limited to small problem sizes, it guarantees the globally optimal solution and therefore provides a reliable baseline for validation. On the other hand, the Quantum Approximate Optimization Algorithm (QAOA) offers a heuristic quantum-classical hybrid approach that leverages quantum superposition and interference to explore the solution space more efficiently. While QAOA does not guarantee global optimality, it scales more favorably for larger and more complex QUBO instances. By comparing results from both methods, we can quantify the performance gap between exact and approximate optimization, evaluate the feasibility of QAOA on realistic scheduling problems, and demonstrate the potential of hybrid quantum–classical pipelines for future large-scale combinatorial optimization tasks.

3.1. Toy Example

To illustrate the modeling and solving pipeline, we consider a simplified instance of the scheduling problem with reduced dimensions and relaxed constraints. This toy example enables explicit formulation of the QUBO representation and direct comparison between classical and quantum solvers.

We define the following instance:

- $T = 5$: the total number of days in the planning horizon.
- $I = \{a, b, c\}$, with $|I| = 3$: the set of interventions to be scheduled.
- $\Delta_i = 2$ for all $i \in I$: the duration (in days) required for each intervention.
- r_i: the number of workers required per day for intervention i, with $r_a = 2$, $r_b = 2$, and $r_c = 3$.
- $u_t = 4$: the maximum number of workers available on day t.
- $l_t = 0$: the minimum number of workers required on day t (here set to 0 for simplicity).
- $\mathcal{E} = \emptyset$: the set of precedence constraints between interventions (empty in this toy example).

Given the duration and daily resource limit, there exist only two feasible global plans: either start interventions $\{a, b\}$ first followed by $\{c\}$, or conversely, start $\{c\}$ first followed by $\{a, b\}$. We deliberately assign risk values $\tilde{Q}(i, t)$ such that one of these two plans becomes the unique global optimum.

The QUBO formulation without resource constraint of this instance is expressed as:

$$\min_{x\in\{0,1\}^{15}} \Big(\sum_{i\in\{a,b,c\}} \sum_{\tau=1}^{5} x_{i,\tau}\tilde{Q}(i,\tau) + \lambda_2 \sum_{i\in\{a,b,c\}} \sum_{t>3} x_{i,t} + \lambda_3 \sum_{i\in\{a,b,c\}} \big(\sum_{t=1}^{5} x_{i,t} - 1\big)^2 \Big),$$

which penalizes late start times (via λ_2) and enforces single-start constraints (via λ_3).

3.2. Matrix representation of the QUBO

The final quadratic binary optimization problem takes the form:

$$\min_{x\in\{0,1\}^{15}} x^T Q x = x^T(\tilde{Q} + \lambda_2\bar{Q} + \lambda_3\hat{Q})x, \tag{4}$$

where Q is decomposed into three structured matrices capturing risk, lateness penalty, and single-start enforcement.

- **Risk matrix $\tilde{Q}$:** The risk associated with each intervention starting at each time step is encoded in the diagonal matrix

$$\tilde{Q} = \operatorname{diag}(\tilde{Q}(a,1), \tilde{Q}(a,2), \tilde{Q}(a,3), \tilde{Q}(a,4), \tilde{Q}(a,5), \tilde{Q}(b,1), \ldots, \tilde{Q}(c,5)),$$

  where $\tilde{Q}(i, t)$ represents the risk of starting intervention i at time t.
- **Lateness penalty matrix $\bar{Q}$:** To penalize late starts (here, times $t = 4, 5$), we define a diagonal matrix

$$\bar{Q} = \operatorname{diag}(0, 0, 0, 1, 1,\ 0, 0, 0, 1, 1,\ 0, 0, 0, 1, 1).$$

- **Single-start penalty matrix $\hat{Q}$:** Each intervention i has a 5×5 block enforcing that it starts exactly once:
$$\hat{Q}_i = \begin{pmatrix} -1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 \end{pmatrix}.$$
Equivalently, the diagonal entries are -1 and the off-diagonal entries are 1.
The full single-start penalty matrix is block-diagonal:
$$\hat{Q} = \begin{pmatrix} \hat{Q}_a & 0 & 0 \\ 0 & \hat{Q}_b & 0 \\ 0 & 0 & \hat{Q}_c \end{pmatrix}.$$

For this toy example, classical exact solvers are capable of enumerating all feasible configurations and therefore can reliably identify the global optimum of the QUBO model. In contrast, the performance of QAOA is highly sensitive to the choice of penalty coefficients associated with the constraints. In our formulation, multiple penalty parameters λ_i are used to enforce distinct constraints such as single-start times and late-start penalties. As highlighted in [12], using unbalanced penalization schemes can improve the encoding of inequality constraints in combinatorial optimization problems for quantum algorithms.

Our numerical experiments indicate that improper selection of λ_i may lead to either a high probability of infeasible solutions or suboptimal feasible solutions. In particular, the probability distribution of sampled QAOA solutions over feasible versus infeasible configurations is strongly influenced by the relative magnitudes of these penalties.

To further explore quantum-enhanced approaches, we consider extending the search using Grover-based quantum algorithms. From a mathematical perspective, this could allow for amplitude amplification of feasible solutions in the solution space, potentially increasing the success probability of obtaining valid schedules while reducing sensitivity to penalty coefficients.

Overall, this toy example demonstrates that while classical methods guarantee optimality, QAOA requires careful tuning of penalty parameters, and hybrid approaches combining amplitude amplification techniques may provide a more robust alternative for constrained combinatorial problems.

An interesting direction for future work is the deployment of the proposed QUBO formulation on quantum annealing hardware, such as D-Wave systems. Given the native support of QUBO models, quantum annealing could provide additional insights into large-scale scheduling problems.

3.3. Numercial examples

The table corresponds to Part (a) of the algorithm 1, presenting the classical solver approach, while Graphs A and B illustrate Part (b) of the algorithm 1, which presents the QAOA solver approach.

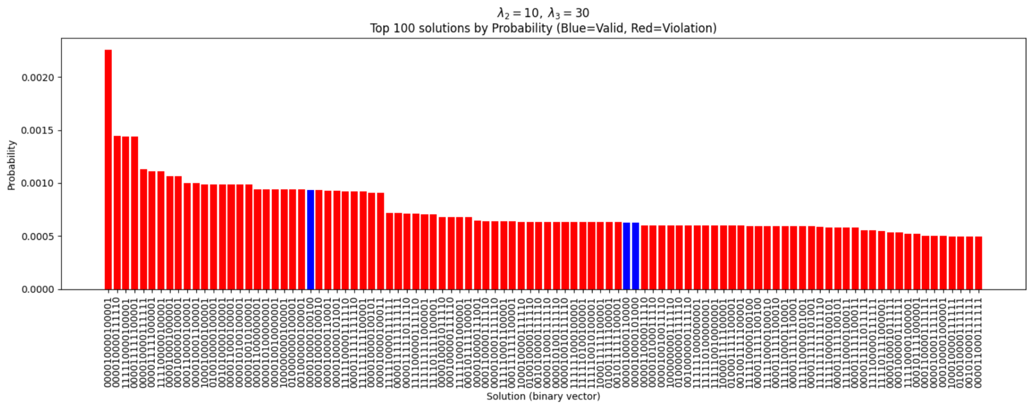
The 15-bit binary strings in the graphs A and B represent the initiation of interventions over time. Let $I = \{a, b, c\}$ denote the set of interventions to be scheduled, with $|I| = 3$. Each bit in the binary string encodes the initiation status of an intervention, where 1 denotes that the intervention has commenced, and 0 denotes that it has not. The first five bits correspond to intervention a, the middle five bits to intervention b, and the last five bits to intervention c. In the experiments, the penalty parameters are set to $\lambda_2 = 10$ and $\lambda_3 = 30$.

Algorithm 1: Classical and Quantum Scheduling Solvers for Binary QUBO Scheduling

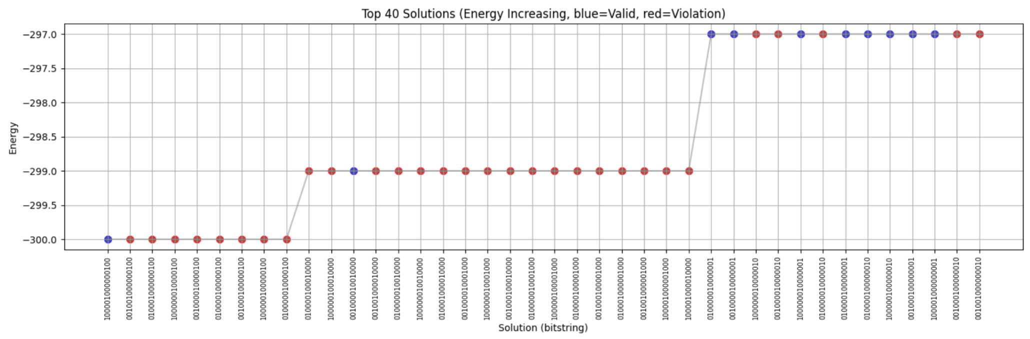
```
Input:  Interventions $I = \{a, b, c\}$ with durations $\Delta_i$ and worker demands $r_i$;
  Total days $T$; worker limit per day $L$;
  QUBO penalty weights $\lambda_2, \lambda_3$ and cost $\tilde{Q}(i, t)$
Output: Feasible schedule(s) minimizing QUBO objective under worker limits
Variable Definition
$x_{i,t} \in \{0, 1\}$: 1 if intervention $i$ starts on day $t$, 0 otherwise
$idx(i, t)$: linear index of $x_{i,t}$ in QUBO matrix $Q$
$W[d, idx(i, t)]$: number of workers used on day $d$ if $i$ starts at $t$
Part (a): Classical ExactSolver Approach
    Step 1: Construct QUBO
    Initialize $Q \leftarrow 0_{|I|\cdot T\times|I|\cdot T}$
    foreach intervention $i \in I$ do
        foreach day $t = 1, \ldots, T$ do
            $Q[idx(i, t), idx(i, t)] \leftarrow Q[idx(i, t), idx(i, t)] + \tilde{Q}(i, t)$
            if $t > T - \Delta_i$ then
                $Q[idx(i, t), idx(i, t)] \leftarrow Q[idx(i, t), idx(i, t)] + \lambda_2$
        Add single-start penalty $\lambda_3(\sum_t x_{i,t} - 1)^2$ to $Q$
    Step 2: Solve QUBO Classically
    Use exact solver (e.g., dimod.ExactSolver) to enumerate all binary configurations
    Obtain solutions $x$ and corresponding energies $f(x)$
    Step 3: Post-filter by Worker Limits
    Construct worker matrix $W$ as defined above
    Sort solutions by energy $f(x)$ ascending
    foreach sample $x$ in top-K solutions do
        Compute total workers per day: $totals = W \cdot x$
        if all $totals \leq L$ then
            Accept $x$ as feasible
    return Feasible schedule(s) with minimum QUBO energy
Part (b): QAOA Solver Approach
    Step 1: Construct QUBO
    (Same as in Part (a)) to obtain matrix $Q$ representing the cost function
    Step 2: Quantum Optimization via QAOA
    Convert $Q$ into a QuadraticProgram
    Initialize QAOA ansatz with depth $p$ and classical optimizer (e.g., COBYLA)
    Use MinimumEigenOptimizer to estimate low-energy states
    Obtain solution samples $\{x_i, f(x_i), p(x_i)\}$, where $p(x_i)$ is the sampling probability
    Step 3: Post-filter by Worker Limits
    Construct worker matrix $W$ as above
    Sort samples by probability $p(x_i)$ descending
    foreach sample $x_i$ in top-K candidates do
        Compute daily totals: $totals = W \cdot x_i$
        if all $totals \leq L$ then
            Accept $x_i$ as feasible
    return Feasible low-energy schedule(s) found via QAOA sampling
```

Table 1. Candidate solutions identified by the classical solver, showing the start times, feasibility, and any violations of worker-capacity constraints.

Solution	Energy	A start	B start	C start	Feasible	Violations
1	-60.0	Day 3	Day 2	Day 3	×	Day 3: 7; Day 4: 5
2	-60.0	Day 1	Day 3	Day 3	×	Day 3: 5; Day 4: 5
3	-60.0	Day 3	Day 3	Day 3	×	Day 3: 7; Day 4: 7
4	-60.0	Day 1	Day 2	Day 3	×	Day 3: 5
5	-60.0	Day 2	Day 1	Day 3	×	Day 3: 5
6	-60.0	Day 3	Day 1	Day 3	×	Day 3: 5; Day 4: 5
7	-60.0	Day 2	Day 2	Day 3	×	Day 3: 7
8	-60.0	Day 2	Day 3	Day 3	×	Day 3: 7; Day 4: 5
9	-60.0	Day 1	Day 1	Day 3	✓	None
10	-59.0	Day 1	Day 3	Day 2	×	Day 2: 5; Day 3: 5



(a) Probabilities of the top 100 solutions found by QAOA. Blue bars indicate solutions meeting the worker limits; red bars indicate violations.



(b) Energy of the top 40 solutions found by QAOA. Blue points correspond to solutions that satisfy the worker limits, while red points correspond to solutions that exceed the worker limits.

4. Grover algorithm

Grover's algorithm is a quantum search algorithm that provides a theoretical quadratic speed-up for unstructured search problems. For example, when finding the minimum value in an unordered table, Grover's algorithm can approximately reach the optimal solution with a complexity of $O(\sqrt{N})$, rather than the classical complexity of $O(N)$ (see [10]). Beyond simple search, variants of Grover's algorithm have been proposed to tackle more sophisticated tasks such as combinatorial optimization [3,5,9]. One such variant is the Grover Adaptive Search (GAS) method, which leverages Grover iterations within an adaptive strategy to iteratively improve candidate solutions.

Grover's algorithm was first introduced in [10]. It aims to solve the following search problem: given a function $f : \{0,1\}^n \to \{0,1\}$, find an input x^* such that $f(x^*) = 1$. Concretely, this optimal solution is amplified so that it obtains a higher probability amplitude than the others. To achieve this, we need to design an oracle that evaluates candidate solutions. The Grover algorithm can therefore be summarized as follows:

```
Algorithm 2: Grover algorithm
  Input:  State initialization into a uniform superposition; $|s\rangle = \frac{1}{N}\sum_{x=0}^{N-1}|x\rangle$
  Output: High amplitude of the marked element(s) so that a measurement yields the optimal solution
          with high probability
1 Variable Definition
2 Oracle that flips the phase of marked states; Diffusion operator that inverts amplitudes about the
    average; Grover iterations repeated approximately $k \approx \frac{\sqrt{N}}{m}$ where $m$ in the number of multiplicity of
    the candidate solution in the data.
```

4.1. Scheduling problem encoding

We first encode the optimization problem (1) in a binary form, such as QUBO, PUBO, or others, in order to apply GAS. In (1), we seek the optimal schedule of n interventions over T days. The optimal schedule specifies only the starting day of each intervention.

The encoding presented in section 2.3 represents the optimal schedule as $x = ([x_i]_i^n)$, where each $x_i \in \{0,1\}^T$. The vector $x_i = [x_{i,1}, \dots, x_{i,T}]$ encodes the starting date of the i-th intervention as follows: $x_{i,t} = 1$ if the intervention starts on day t, and $x_{i,t'} = 0$ for all $t' \neq t$. This encoding simplifies the design of both the cost function and the constraints. However, the decision vector x lies in $\{0,1\}^{nT}$ and therefore requires nT qubits, which can be quite demanding in practice.

We now consider a new encoding. Let $x = ([x_i]_{i=1}^n)$, where each x_i is the binary representation of the starting date of the i-th intervention, i.e., $x_i \in \{0,1\}^{\lceil \log_2 T \rceil}$. This encoding guarantees that constraint (1b) is satisfied by construction, since the decimal representation of x_i falls within $[1,T]$. Therefore, the optimization (1) reads:

$$\min_{x \in \{0,1\}^{n\lceil \log_2 D \rceil}} \left(\sum_t \sum_i \sum_s \overline{risk_{i,s}^t} \prod_j x_{i,j}^{s_j} (1 - x_{i,j})^{(1-s_{i,j})} \right) \tag{5a}$$

$$\sum_s (s + \Delta_{i,s}) \prod_k x_{i,k}^{s_k} (1 - x_{i,k})^{(1-s_k)} \leq D + 1, \quad \forall i, \tag{5b}$$

$$\text{s.t.} \quad l_t^c \leq \sum_{i \in I} \sum_{0 \leq t-s \leq \Delta_{i,s}} r_{c,t}^{i,s} \prod_k x_{i,k}^{s_k} (1 - x_{i,k})^{(1-s_k)} \leq u_t^c, \quad \forall c, \forall t, \tag{5c}$$

$$(0 \leq t - s \leq \Delta_{i,s} \wedge 0 \leq t - r \leq \Delta_{j,r}) \Rightarrow \prod_k x_{i,k}^{s_k} (1 - x_{i,k})^{(1-s_k)} x_{j,k}^{r_k} (1 - x_{j,k})^{(1-r_k)} = 0, \tag{5d}$$

$$\forall (i,j,t) \in \mathcal{E}, \forall s, r \in \{0,1\}^{\lceil \log_2 D \rceil}. \tag{5e}$$

Notice that we use here r and s to both denote natural numbers and their binary representation.

The general formulation thus leads to a PUBO (Polynomial Unconstrained Binary Optimization) problem, since the cost function is expressed as a polynomial in binary variables. For instance, if $T = 365$, then $d = \lceil \log_2 T \rceil = 9$, and the decision variable $x \in \{0,1\}^{dn}$. To illustrate this encoding, we consider the following toy example. Let three interventions $I = \{i_1, i_2, i_3\}$, be scheduled over four days, indexed from day 0 to day 3. In this case, we have $n = 3$, $T = 4$, and consequently $d = 2$. The binary decision variable x therefore belongs to $\{0,1\}^6$ given as

$$x = [x_1, x_2, x_3] = [x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}, x_{3,1}, x_{3,2}].$$

We denote the starting date τ_i of the i-th intervention by the binary vector $x_i = (x_{i,1}, x_{i,2})$, where the binary representation $\tau_i = x_{i,1}x_{i,2}$ corresponds to the decimal value $2x_{i,1} + x_{i,2}$.

The constraints are the following :

- (5b): each intervention has a duration of 2 consecutive days and must be completed within a time window of 5 days: $\Delta_{i,s} = 2$, $\forall i, s$. Combined with the risk associated with each intervention, this yields table 2

$x_{1,1}$	$x_{1,2}$	τ_1	$\overline{\text{risk}_{\tau_1}}$	$x_{2,1}$	$x_{2,2}$	τ_2	$\overline{\text{risk}_{\tau_2}}$	$x_{3,1}$	$x_{3,2}$	τ_3	$\overline{\text{risk}_{\tau_3}}$
0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	1	1	0	0	1	1	0
1	0	2	0	1	0	2	0	1	0	2	1
1	1	3	2	1	1	3	2	1	1	3	3

Table 2. Risks of the interventions (toy example)

- (5c): here there is non minimum required usage of resource c at time t, we only have the maximum number of workers per day: $r_t^c = 4$ $\forall c, \forall t$.
  - On each day, at most 4 workers are available.
  - Interventions i_1 and i_2 each require 2 workers per day.
  - Intervention i_3 requires 3 workers per day.

  Denoting $\varphi_t(x)$ the workers limit at t, the workers limit constraint are given by:

  $$\varphi_t(x) \leq 2, \quad t = 0, 1, 2, 3,$$

  where $\varphi_3(x) = x_{1,1}x_{1,2} + x_{2,1}x_{2,1} + x_{3,1}x_{3,2}$, $\varphi_2(x) = x_{1,2} + x_{2,2} + x_{3,2} - \varphi_3(x)$, $\varphi_1(x) = x_{1,1} + x_{2,1} + x_{3,1} - \varphi_3(x)$ and $\varphi_0(x) = \varphi_3(x) - \varphi_2(x) - \varphi_1(x) + 3$
- (5a): the cost function becomes

  $$\text{cost}(x) = 4x_{1,1}x_{1,2} + 4x_{2,1}x_{2,2} - 2x_{3,1} - 2x_{3,1}x_{3,2}.$$

To address the optimization problem in (5), an oracle circuit is devised that jointly encodes both the cost function and all relevant constraints into a single quantum operation. By integrating these elements within the same oracle, Grover's amplitude amplification selectively enhances states corresponding to feasible candidate solutions while attenuating those associated with configurations that violate one or more constraints.

The total number of qubits required for this oracle depends on the structure of the specific optimization instance. In particular, it is determined by the number of decision variables and the precision of their binary encoding, which sets the dimensionality of the search space. The number, complexity, and nonlinearity of the constraints also play a key role, since each constraint typically requires additional circuitry for verification. Beyond these primary registers, ancilla qubits are needed to support intermediate arithmetic operations, comparisons, and logical checks for evaluating both the cost function and the constraints. These ancilla qubits enable reversible computation and allow the oracle to be uncomputed when required, thereby preserving quantum coherence.

Consequently, the total qubit count can vary significantly between problem instances and may increase rapidly with problem size or constraint complexity, underscoring the need for careful oracle design. Striking a balance between faithful problem encoding and economical qubit usage is crucial to keeping resource overhead within practical limits. In practice, this motivates the use of problem-specific encodings, circuit optimizations, and qubit-recycling techniques to reduce both the depth and width of the oracle circuit, thereby enhancing the practicality of Grover-based optimization on near-term and future quantum architectures. In the next section, we design an oracle for the toy example introduced here and determine the corresponding number of qubit requirements.

4.2. **Oracle for the toy example**

To solve the toy example problem presented in Section 4.1, we build up a quantum circuit shown in Figure 2. In that circuit we implement GAS with two iterations. The important part of the circuit is designing the Oracle that intervenes. To design such an Oracle, we follow the description presented in [9] Section 3.2. Therein, the cost function and the constraints are directly encoded so that at each iteration a feasible candidate solution may be determined for a given threshold.

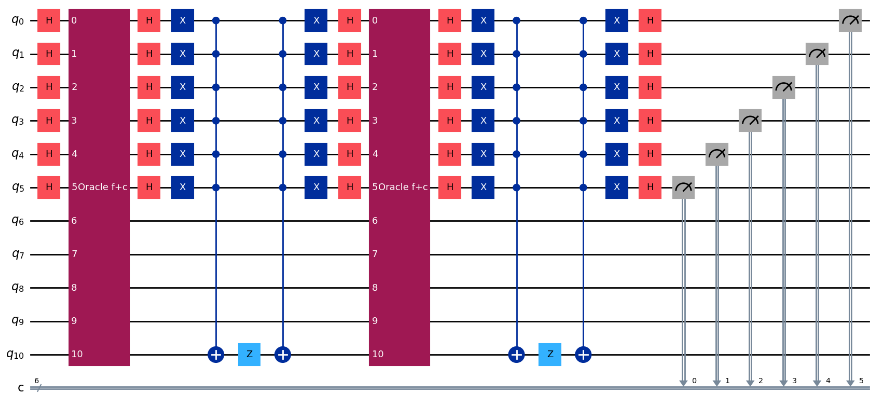


FIGURE 2. Quantum circuit using GAS for solving the toy example with two iterations.

The oracle which is represented as the long red rectangle on Figure 2 operates through five main computational phases that work in concert to evaluate the optimization problem and mark feasible solutions. Each phase serves a specific purpose in the overall oracle logic, and the oracle is designed so that all intermediate computational results are reversed and cleaned up at the end, leaving only the marker qubit affected.

- Decision variable x : this register contains the six input qubits that represent the decision variables of the optimization problem. These qubits hold the binary encoding of which task starts on which day. These qubits are not modified by the oracle—they serve as inputs that determine the values of the constraint and objective functions.
- Cost and constraints functions register: the oracle needs to compute and store intermediate polynomial values. Separate quantum registers are allocated for each major computation: the cost function register (cost(x) : 4 qubits) to store the cost function value, and four constraint registers ($\varphi_0(x), \varphi_1(x), \varphi_2(x)$ and $\varphi_3(x)$: $5, 4, 4$ and 3 qubits respectively) to store the values of each constraint function. These registers will temporarily contain the quantum superpositions representing the computed values. The size of each register is determined by the range of possible values—for example, if the cost function can range from 0 to 15, then 4 qubits are needed to represent this range. So 20 qubits is needed to these functions.

- Flag qubits: after computing the function and constraint values, the oracle must determine whether each value satisfies its respective threshold. For this purpose, five flag qubits are allocated to indicate whether each of the four constraints is satisfied. Each flag qubit will be set to 1 if its corresponding condition is true, and 0 otherwise.
- Marker qubit: a single marker qubit (m) serves as the target of the oracle's phase flip. This qubit will be flipped (from 0 to 1 or 1 to 0) only if all constraints and the objective are satisfied simultaneously.
- Ancilla qubits: finally, temporary storage qubits (ancillas) are allocated to assist in intermediate computations, particularly for evaluating monomials. These ancillas will be used in computations and then reset to their initial state before the oracle completes.

The oracle first evaluates the pseudo-Boolean polynomial functions using a Quantum Fourier Transform (QFT) approach. Each monomial is computed with multi-controlled-X gates to obtain the product of the relevant variables and then encoded into the target register through controlled phase rotations, with angles proportional to the term's coefficient and the qubit position. Ancilla qubits used for intermediate computations are uncomputed to free resources and minimize noise. After all monomials are processed, an inverse QFT returns the register to the computational basis, yielding the final computed values of the functions.

Next, the oracle checks each value against its threshold using quantum comparators, transforming the registers into Fourier space and applying phase rotations to implement subtraction. The most significant bit of the result is used to set flag qubits indicating whether each constraint or the cost function is satisfied. Finally, a multi-controlled-X gate flips the marker qubit only if all flags are 1, marking feasible solutions. All intermediate computations, including the comparators and function evaluations, are uncomputed, ensuring a clean oracle that leaves only the marker qubit affected.

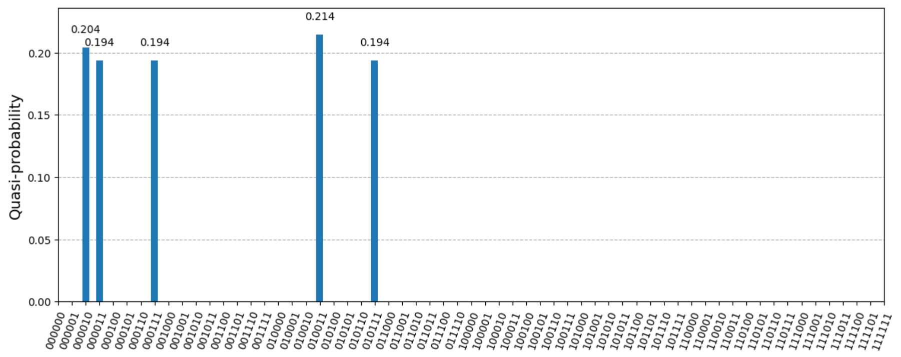


Figure 3. Probability of reaching a feasible solution

The quantum circuit can be simulated using the `qiskit` library developed by IBM[1] . This library enables the simulation of quantum circuits on classical computers. However, such simulations require significant memory resources, typically available only on high-performance computing clusters. In our experiments 32 qubits at least is needed to simulate the quantum circuit. The available computing resources did not provide sufficient memory to simulate this full circuit. To address this limitation, we encoded the Oracle classically using the `numpy` library, thereby reducing the number of qubits required in the quantum simulation. With this approach, only 12 qubits (6 to encode x the decision variable, 6 ancilla as details on Figure 2) are needed to simulate the quantum circuit, while the Oracle is evaluated entirely using classical computation. The resulting hybrid circuit is then simulated using `qiskit`. We perform two Grover iterations with 8192 shots. The simulation results are shown in Figure 3. We show the probability of appearance of each feasible candidate for the 8192 shots. We observe that five feasible solutions that have high probability are:

[1]https://www.ibm.com/quantum/qiskit

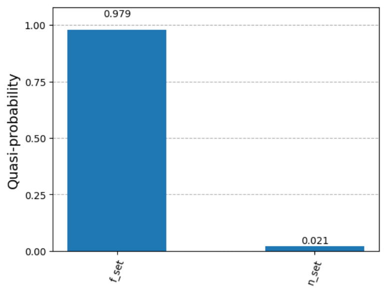


FIGURE 4. Probability of reaching a solution in the feasible solution set: `f_set` is feasible set (that contains feasible solutions) and `n_set` is non-feasible set.

- $x = (0, 0, 0, 0, 1, 0)$: interventions i_1 and i_2 start on the first day (day 0 in our notation), and intervention i_3 starts on the third day (day 2),
- $x = (0, 0, 0, 0, 1, 1)$: interventions i_1 and i_2 start on the first day, and intervention i_3 starts on the fourth day,
- $x = (0, 0, 0, 1, 1, 1)$: intervention i_1 starts on the first day, intervention i_2 on the second day, and intervention i_3 on the fourth day,
- $x = (0, 1, 0, 0, 1, 1)$: intervention i_1 starts on the second day, intervention i_2 on the first day, and intervention i_3 on the fourth day,
- $x = (0, 1, 0, 1, 1, 1)$: interventions i_1 and i_2 start on the second day, and intervention i_3 starts on the fourth day.

Using these results, we estimate the probability of obtaining one of the feasible solutions by GAS. The corresponding result is shown in Figure 4. We observe that the GAS algorithm reaches a feasible solution with a probability of 97.9. These results indicate that GAS is effective in solving the binary optimal control problem defined in (5), at least for problem instances of moderate size. The high success probability highlights the potential of quantum-inspired or hybrid quantum–classical approaches for combinatorial optimization problems.

However, simulating the complete formulation of the problem on classical hardware remains challenging, as it requires high-performance computing resources with substantial memory capacity, particularly when the number of qubits increases. This limitation currently restricts the scalability of the approach in simulation environments. Future work will therefore focus on the analysis of qubit requirements, the design of more efficient Oracle constructions, and the extension of the method to general large-scale optimization problems, with the goal of enabling more scalable and resource-efficient simulations.

CONCLUSION

This work explored risk-based maintenance scheduling for power transmission systems through the lens of quantum optimization, motivated by the growing computational challenges faced by transmission system operators. As network size, operational constraints, and risk exposure continue to increase, classical mixed-integer optimization methods are reaching practical scalability limits, calling for alternative computational paradigms.

By reformulating the maintenance scheduling problem as a Quadratic Unconstrained Binary Optimization (QUBO) model, this study enabled the use of Quantum Approximate Optimization Algorithm (QAOA) and Grover Adaptive Search. Numerical experiments on reduced-size instances demonstrate that quantum algorithms can effectively explore complex combinatorial solution spaces and identify low-risk schedules, providing a concrete proof of concept for quantum-assisted maintenance planning.

However, this work also highlights several fundamental limitations of current QUBO-based formulations. First, constraint enforcement through penalty terms introduces a delicate trade-off: penalty constants must be sufficiently large to ensure feasibility, yet small enough to avoid overwhelming the objective function. In practice, this balance is difficult to achieve and often requires empirical tuning, which may lead to suboptimal solutions or, in some cases, constraint violations. This issue is particularly critical in quantum optimization, where excessive penalties can mask the objective landscape and significantly degrade algorithmic performance.

Second, the use of slack variables to encode inequality constraints, especially resource bounds, leads to a substantial increase in the number of binary variables and therefore in the number of required qubits. While this expansion is largely manageable in classical computing, it becomes a major bottleneck in quantum settings, where qubit availability is severely limited and circuit depth must be carefully controlled. As a result, the practical scalability of the proposed QUBO formulation is currently constrained not only by hardware limitations, but also by the structural complexity of the constraint encoding itself.

These limitations point to clear and promising directions for future research. Improving constraint encoding to reduce or eliminate slack variables, developing adaptive or theoretically grounded penalty selection strategies, and designing hybrid formulations that selectively offload constraints to classical components could significantly enhance the viability of quantum optimization for real-world maintenance scheduling. More broadly, identifying problem structures that allow for native or constraint-light quantum formulations will be key to unlocking meaningful quantum advantage.

In the second part, we implement the Grover algorithm to optimize the scheduling. This modeling approach allows us to obtain optimal results with strong practical and theoretical guarantees. It also demonstrates quantum supremacy in this context. However, the implementation is complex and challenging. Moreover, it is quite hard—or even impossible—to implement such an algorithm on current quantum hardware, which remains one of the main limitations.

Ultimately, this work positions maintenance scheduling as a compelling testbed for quantum optimization in power systems. While real-scale deployment remains a long-term objective, the insights gained here contribute to shaping the next generation of hybrid decision-support tools, where quantum and classical methods jointly address the growing complexity of risk-aware planning in future electricity networks.

References

[1] M. Abirami, S. Ganesan, S. Subramanian, and R. Anandhakumar. Source and transmission line maintenance outage scheduling in a power system using teaching learning based optimization algorithm. *Applied Soft Computing*, 21:72–83, 2014.

[2] Akshay Ajagekar and Fengqi You. Quantum computing for energy systems optimization: Challenges and opportunities. *Energy*, 179:76–89, 2019.

[3] William P Baritompa, David W Bulger, and Graham R Wood. Grover's quantum algorithm applied to global optimization. *SIAM Journal on Optimization*, 15(4):1170–1184, 2005.

[4] Guillaume Crognier, Pascal Tournebise, Manuel Ruiz, and Patrick Panciatici. Grid operation-based outage maintenance planning. *Electric Power Systems Research*, 190:106682, 2021.

[5] Christoph Durr and Peter Hoyer. A quantum algorithm for finding the minimum. *arXiv preprint quant-ph/9607014*, 1996.

[6] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014.

[7] Aurélien Froger, Michel Gendreau, Jorge E. Mendoza, Éric Pinson, and Louis-Martin Rousseau. Maintenance scheduling in the electricity industry: A literature review. *European Journal of Operational Research*, 251(3):695–706, June 2016.

[8] Austin Gilliam, Stefan Woerner, and Constantin Gonciulea. Grover adaptive search for constrained polynomial binary optimization. *Quantum*, 5:428, April 2021.

[9] Austin Gilliam, Stefan Woerner, and Constantin Gonciulea. Grover adaptive search for constrained polynomial binary optimization. *Quantum*, 5:428, 2021.

[10] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[11] Lov K. Grover. Quantum mechanics helps in searching for a needle in a haystack. *Physical Review Letters*, 79(2):325–328, July 1997.

[12] J A Montañez-Barrera, Dennis Willsch, A Maldonado-Romo, and Kristel Michielsen. Unbalanced penalization: a new approach to encode inequality constraints of combinatorial problems for quantum optimization algorithms. *Quantum Science and Technology*, 9(2):025022, April 2024.

[13] Bahar Cennet Okumusoglu, Beste Basciftci, and Burak Kocuk. A joint chance-constrained stochastic programming approach for the integrated predictive maintenance and operations scheduling problem in power systems, 2022.

[14] ROADEF & EURO. Roadef/euro challenge 2020: Maintenance planning problem. `https://roadef.org/challenge/2020/en/`, 2020. Consulté le [date d'accès].

[15] Mariana Faria Pires Gama Rocha. *Mixed-integer Programming Models for Maintenance Scheduling in Power Transmission Systems*. PhD thesis, Polytechnique Montréal, mai 2023.

A. Penalty coefficient sensitivity analysis

A.1. Determining the penalty coefficient λ

The penalty coefficient λ is a crucial parameter in the QUBO formulation that determines how strictly the constraints are enforced. The higher the value of λ, the more heavily the algorithm penalizes constraint violations.

In the Qiskit `QuadraticProgramToQubo` converter, the penalty coefficient λ is determined automatically depending on the nature of the problem's constraints. Specifically, for problems involving floating-point coefficients in the constraints, the default penalty value is set to a large number:

$$\lambda = 10^5$$

For problems with integer coefficients or linear constraints that do not involve floating points, the penalty coefficient is calculated as:

$$\lambda = 1 + (\text{lin\_b.upperbound} - \text{lin\_b.lowerbound}) + (\text{quad\_b.upperbound} - \text{quad\_b.lowerbound})$$

Here, `lin_b` refers to the linear terms in the objective function or constraints, and `quad_b` refers to the quadratic terms. The `upperbound` and `lowerbound` represent the respective range of values for these terms. This method of determining λ is detailed in the Qiskit documentation [?], which provides further insight into how penalty coefficients are adjusted based on the problem's bounds.

A.2. Impact of λ on solution validity

The unconstrained QUBO formulation of this instance is expressed as:

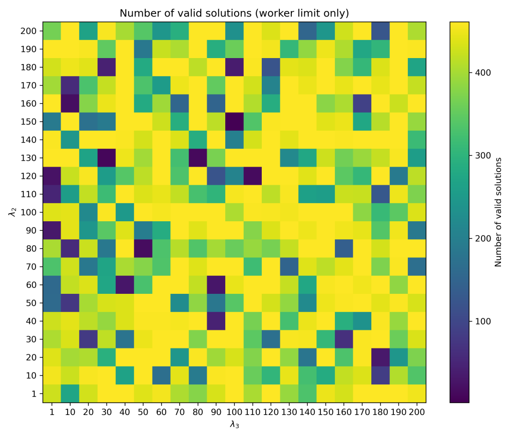
$$\min_{x\in\{0,1\}^{15}} [\sum_{i\in\{a,b,c\}} \sum_{\tau=1}^{5} x_{i,\tau}\tilde{Q}(i,\tau) + \lambda_2 \sum_{i\in\{a,b,c\}} \sum_{t>3} x_{i,t} + \lambda_3 \sum_{i\in\{a,b,c\}} (\sum_{t=1}^{5} x_{i,t} - 1)^2],$$

which penalizes late start times (via λ_2) and enforces single-start constraints (via λ_3).

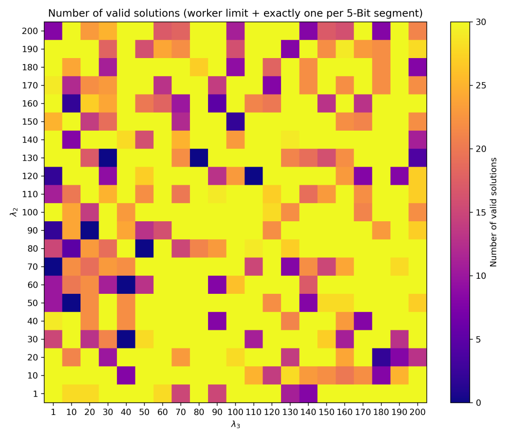
The penalty coefficients λ_2 and λ_3 control the strength of these constraints. Higher values of λ_2 and λ_3 force the model to adhere strictly to the early start and single start constraints, respectively, reducing the number of feasible solutions that violate the constraints. However, excessively large values of λ may lead to a reduction in solution diversity, potentially causing the solver to focus on solutions that satisfy the constraints but are suboptimal in terms of the original objective.

To find the optimal balance, a sensitivity analysis was performed, testing a range of values for λ_2 and λ_3. Figures 5 and 6 illustrate the relationship between different penalty coefficients λ_2 and λ_3 and the number of valid solutions obtained.

The first set of figures (Figure 5) shows the number of valid solutions under different constraint settings, while the second set (Figure 6) focuses on the top 200 high-probability solutions. In both cases, the x-axis and y-axis represent the values of λ_2 and λ_3, and the depth of the bars indicates the number of valid solutions.

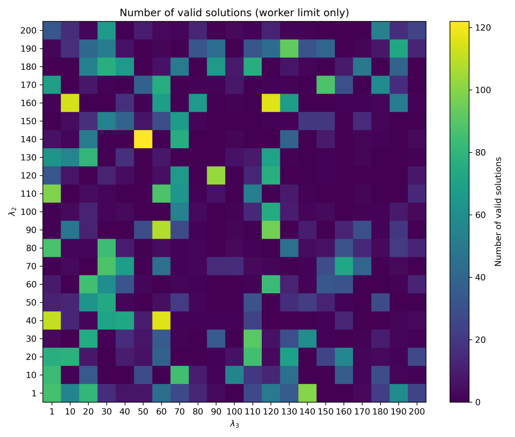


(A) Worker limit only

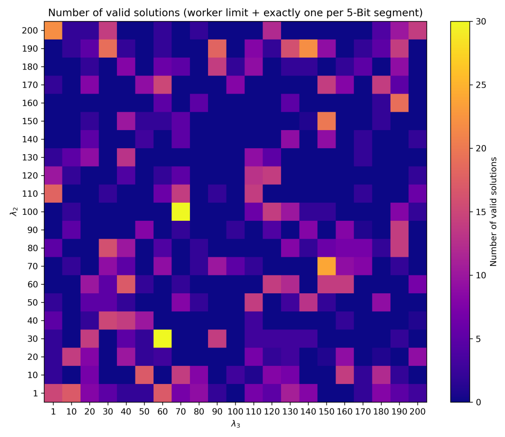


(B) Worker limit and single-start constraints

FIGURE 5. Number of valid solutions under single and combined constraints.



(A) Worker limit only



(B) Worker limit and single-start constraints

FIGURE 6. The number of valid solutions of the top 200 high-probability solutions under single and combined constraints.