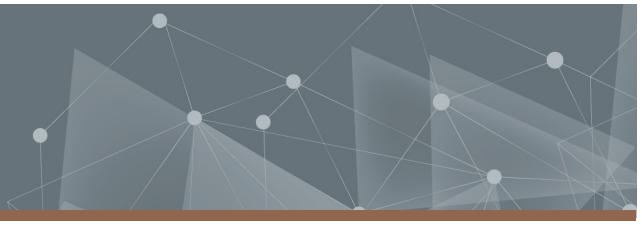
CHALMERS
UNIVERSITY OF TECHNOLOGY



Quantum Optimization of Physician Scheduling For Maximal Healthcare Capacity

A Comparative Study of Classical and Quantum Approaches to Combinatorial Workforce Scheduling

Master's thesis in Systems, Control and Mechatronics

NATHALIE LARSSON & SONJA KANEROT

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

Quantum Optimization of Physician Scheduling for Maximal Healthcare Capacity

A Comparative Study of Classical and Quantum Approaches to Combinatorial Workforce Scheduling

NATHALIE LARSSON
SONJA KANEROT

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Systems, Control and Mechatronics
Chalmers University of Technology
Gothenburg, Sweden 2025

Quantum Optimization of Physician Scheduling for Maximal Healthcare Capacity
A Comparative Study of Classical and Quantum Approaches to Combinatorial Workforce Scheduling
NATHALIE LARSSON
SONJA KANEROT

Examiner and Supervisor: Martin Fabian, Department of Electrical Engineering

Master's Thesis 2025
Department of Electrical Engineering
Division of Systems, Control and Mechatronics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Printed by TeknologTryck
Gothenburg, Sweden 2025

Quantum Optimization of Physician Scheduling for Maximal Healthcare Capacity
A Comparative Study of Classical and Quantum Approaches to Combinatorial Workforce Scheduling
NATHALIE LARSSON & SONJA KANEROT
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Physician scheduling is a critical challenge in healthcare systems, demanding a balance between operational efficiency, fairness, and individual preferences. This thesis investigates the use of quantum computing, specifically the Quantum Approximate Optimization Algorithm (QAOA), as a novel approach to solving the Physician Scheduling Problem (PSP), a known combinatorial optimization task. Classical methods, Mixed Integer Linear Programming (MILP) with Gurobi and Satisfiability Modulo Theories (SMT) with Z3, are implemented as benchmarks and used to establish feasible, constraint-satisfying solutions. The PSP is formulated as a Quadratic Unconstrained Binary Optimization (QUBO) problem, which serves as input to the QAOA algorithm. This is then executed on both quantum simulators and IBM's real quantum hardware. A modular scheduling framework is developed to encode fairness, availability, preferences, and contractual work extent into the objective functions, enabling both short- and long-term optimization scenarios. Comparative evaluations reveal that while classical solvers consistently yield feasible schedules, QAOA demonstrates potential for competitive solution quality despite current hardware limitations.

Keywords: physician scheduling, quantum optimization, QAOA, constraint satisfaction, Gurobi, Z3, QUBO, healthcare operations, hybrid solvers.

Acknowledgements

We would like to express our deepest gratitude to individuals and institutions who have supported and contributed to this thesis project.

First, we acknowledge support from the WACQT Quantum Technology Testbed operated by Chalmers Next Labs, which is funded by the Knut and Alice Wallenberg Foundation. Here, we also express our gratitude to Pontus Vikstål for his invaluable technical guidance and for providing access to quantum hardware resources. His expertise and insights into theoretical and practical quantum computing significantly shaped the direction and feasibility of our quantum optimization framework.

We are also sincerely grateful to Professor Justin F. Schneiderman at the Department of Clinical Neuroscience, Institute of Neuroscience and Physiology, Sahlgrenska Academy, University of Gothenburg, and Innovationsplattformen, Sahlgrenska University Hospital, Region Västra Götaland, for his support and perspective on the clinical relevance of scheduling in healthcare.

Special thanks go to Dr. Pontus Pfannenstill, MD and abdominal wall surgeon at Sahlgrenska University Hospital, for providing practical insights into real-world physician scheduling challenges and for sharing perspectives on how AI and algorithmic solutions could be integrated into clinical operations.

Finally, we wish to acknowledge our examiner and supervisor Martin Fabian at Chalmers University of Technology for his ongoing support, feedback, and encouragement throughout this project.

Nathalie Larsson & Sonja Kanerot, Gothenburg, May 2025

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

CP	Constraint Programming
ILP	Integer Linear Programming
MILP	Mixed Integer Linear Programming
NSP	Nurse Scheduling Problem
OMT	Optimization Modulo Theories
PSP	Physician Scheduling Problem
QA	Quantum Annealing
QAOA	Quantum Approximate Optimization Algorithm
QC	Quantum Computing
QUBO	Quadratic Unconstrained Binary Optimization
SMT	Satisfiability Modulo Theories

Nomenclature

Below is the nomenclature of indices, sets, parameters, variables, and other terms that have been used throughout this thesis.

Indices

p	Index representing a physician
s	Index representing a shift

Sets

$\mathcal{P}$	Set of physicians
$\mathcal{S}$	Set of shifts
$\mathcal{F}$	Set of feasible schedules

Parameters

λ	Penalty weight for a specific objective component (e.g., fairness, preference, extent)
$\bar{T}$	Average number of shifts per physician
d_s	Demand for shift s

Variables

$x_{p,s}$	Binary decision variable: 1 if physician p is assigned to shift s, 0 otherwise
T_p	Total number of shifts assigned to physician p
u_p	Auxiliary variable representing fairness deviation for physician p

Q	QUBO matrix representing the quadratic objective function
H_C	Cost Hamiltonian used in quantum optimization
$\hat{x}$	Final binary solution vector (bitstring) from quantum sampling

Other Terms

Work rate	Ratio representing how close a physician's workload is to their contracted extent
Extent	Contracted employment level of a physician (e.g., 100%, 75%, 50%)
Shift attractiveness	Metric used to weight the importance of a shift based on physician preferences and availability
HC-cost	Cost value calculated from the Hamiltonian, used to evaluate QAOA solutions
SWAP gate	A quantum gate that exchanges the states of two qubits, often required for non-adjacent qubit interactions
Circuit depth	The number of sequential quantum gate layers in a circuit; higher depth increases noise and error risk
Ising model	A model from physics representing binary spin interactions, structurally similar to QUBO formulations
Estimation shots	Number of samples taken during cost expectation estimation in QAOA
Sampling shots	Number of final measurement samples used to extract QAOA solutions

Contents

List of Figures

List of Tables

1

Introduction

This chapter introduces the context, motivation, and scope of the project. It begins by outlining the complexity and importance of physician scheduling in modern healthcare, emphasizing both operational and human-centered challenges. The background provides an overview of traditional approaches and highlights the emerging potential of quantum computing, particularly the Quantum Approximate Optimization Algorithm (QAOA), as a promising method for solving combinatorial scheduling problems. A review of relevant literature follows, summarizing classical and quantum techniques as well as the health implications of shift work. The chapter then defines the specific aim of the study, presents the research questions, and outlines the delimitations that constrain the problem space for the scope of this project.

1.1 Background

Scheduling healthcare professionals, particularly physicians, is a complex and high-stakes task with direct implications for hospital efficiency, staff well-being, and patient safety [1]. The schedules must satisfy numerous constraints, including adhering to legal working hours, medical qualifications, workload balance, and individual preferences. This complexity is increased by the dynamic and unpredictable nature of healthcare environments, where emergencies, absences, and fluctuating demand often require rapid rescheduling.

Traditionally, physician scheduling has been managed manually or through classical optimization algorithms. While these methods can produce feasible schedules, they struggle to scale efficiently as the problem size and constraint complexity increases [2]. The Physician Scheduling Problem (PSP) is combinatorial by nature; the number of possible configurations grows exponentially with the number of physicians, shifts, and constraints. This has motivated extensive research into more robust computational methods for PSP [3]–[7].

Quantum Computing (QC) offers a new computing approach that uses the principles of quantum mechanics, such as superposition and entanglement, to perform complex optimization tasks potentially more efficiently than classical methods [2]. One of the most promising quantum algorithms for combinatorial optimization is QAOA, first

proposed by Farhi et al. [8]. QAOA is a hybrid algorithm that combines quantum operations with classical optimization loops to approximate solutions to problems formulated as Quadratic Unconstrained Binary Optimization (QUBO) problems. Since scheduling can be naturally expressed in this format, QAOA has received growing attention for its potential in workforce rostering applications.

Recent studies have demonstrated the feasibility of applying quantum optimization to healthcare settings. For example, QAOA and quantum annealing have been applied to the Nurse Scheduling Problem (NSP) with promising results [9], [10], and to logistics-related challenges such as tail-assignment [11]. These early applications suggest that quantum methods could effectively explore large, highly constrained scheduling spaces and offer competitive solutions [12]–[15].

In this context, developing and evaluating a QAOA-based approach for physician scheduling represents a promising research direction. By comparing its performance with classical methods and incorporating real-world scheduling constraints, this project aims to explore how QC can contribute to more flexible, scalable, and efficient scheduling in modern healthcare systems.

1.2 Literature Review

Early studies emphasized the mathematical difficulty of the problem. As mentioned by Van den Bergh et al. [16] and Ernst et al. [17], healthcare staff scheduling in an NP-hard optimization problem, which means that finding exact solutions becomes computationally infeasible as the problem size increases. These studies also reviewed common optimization frameworks used in this domain, including Integer Linear Programming (ILP), Constraint Programming (CP), and various metaheuristics.

Constraint programming has shown strong potential for handling scheduling constraints while offering flexibility. In a real-world application, Weil et al. [18] developed a CP-based system to manage nurse scheduling. This was used at hospital pilot sites and was designed to handle requirements such as shift coverage, legal constraints and personal preferences. Similarly, Caprara et al. [19] applied mixed-integer programming to nurse rostering and demonstrated how fairness and staff preferences can be incorporated into the optimization model. Burke et al. [20] introduced a two-phase strategy to simplify nurse scheduling by separating workload distribution from shift assignment, which improved the model's adaptability to both organizational and personal constraints.

Beyond deterministic methods, metaheuristic algorithms such as simulated annealing and genetic algorithms have been successfully used for large or flexible scheduling problems. Socha et al. [21] showed that combining simulated annealing with CP led to more efficient solutions in exam timetabling, and similar approaches have since been applied to healthcare settings. These techniques are particularly useful when the number of feasible solutions is large, and exact methods become too slow.

While much of the technical literature focuses on the mathematical formulation and solution of the scheduling problem, it is essential to recognize that optimization models must also account for the human aspects of shift work. A growing amount of research highlights the health consequences of poorly designed shift schedules. Costa [22] reported that shift work is associated with chronic sleep disorders, gastrointestinal problems, and a higher risk of cardiovascular disease. Di Muzio et al. [23] found that nurses working rotating shifts had reduced sleep quality and higher fatigue levels compared to those on fixed day shifts. More recently, Ferri et al. [24] observed that rotating shift nurses experienced more symptoms of burnout and digestive issues than their daytime counterparts.

The impact of shift work also extends to cognitive performance and patient safety. Wagstaff and Sigstad Lie [25] conducted a systematic review and concluded that irregular or extended work hours can reduce alertness and increase the likelihood of errors. Karhula et al. [26] highlighted how unpredictable shift patterns contribute to increased stress and absenteeism among healthcare staff. These findings stress the importance of including health-related considerations when designing scheduling algorithms.

Modern scheduling systems increasingly aim to account for staff preferences, recognizing that satisfaction and retention are influenced by how well individual needs are met. Goetschalckx and Jacobs-Blecha [27] demonstrated how preferences could be included as soft constraints, allowing optimization models to balance fairness with feasibility.

In the area of physician scheduling, Erhard et al. [28] provided a thorough overview of the current state of the art, emphasizing the importance of fairness, cyclic schedules, and proper rest periods. More recently, Kraul et al. [29] explored how resilient break scheduling can help reduce physician burnout while maintaining service coverage. From a policy perspective, the Swedish Medical Association's report *Tid till vård ger vård i tid* [30] emphasizes the importance of work-life balance for physicians and links better scheduling with improved patient care.

Emerging technologies such as QC have introduced new ways of thinking about optimization. Blekos et al. [31] reviewed the QAOA and its application to scheduling problems formulated as QUBO. Although practical quantum advantage is still limited by hardware constraints and noise, QAOA represents a promising direction for future research, especially for large combinatorial problems.

1.3 Aim

The aim of this study is to explore how QC can be used to solve the PSP by formulating it as a QUBO problem. The study focuses on using QAOA, which is run on both IBM quantum simulators and hardware. To evaluate its performance, the schedules generated by the quantum approach are compared to those produces by the classical solvers Gurobi and Z3. By working with small-scale scheduling

examples that fit within the limits of today's quantum devices, the goal is to create a practical proof of concept showing how QC can help generate fair and constraint-satisfying schedules in healthcare settings.

1.4 Research Questions

The research seeks to answer the following questions:

- How does the performance of a quantum-based approach (QAOA on IBM Quantum hardware and simulators) compare to classical optimization methods such as Satisfiability Modulo Theories (SMT) using Z3 and simplex using Gurobi in terms of scalability, solution quality, and constraint satisfaction?

- What are the current limitations and challenges of implementing quantum scheduling solutions in a real-world healthcare setting?

1.5 Delimitations

Due to limited time and computational resources, this study focuses on simplified instances of the physician scheduling problem involving a reduced number of staff and shifts. The scheduling scenarios are intentionally small to fit within the qubit limits of current quantum hardware, typically around 20 qubits before the amount of noise makes the results redundant.

The optimization model uses binary decision variables, which are best suited for quantum algorithms like QAOA. This choice simplifies the problem by focusing on whether or not a physician is assigned to a specific predefined shift, rather than considering continuous factors such as start times or shift durations.

Only one quantum algorithm (QAOA) is explored, and its performance is compared with two classical solvers, Z3 and Gurobi, on small instances. For larger scenarios, Z3 was excluded due to computational constraints, and only Gurobi was used as a classical baseline. The aim is not to provide a full-scale scheduling solution but rather to demonstrate a practical proof of concept.

2

Theory

This chapter provides the theoretical foundation for the classical and quantum optimization methods used in this study. It begins by outlining the principles of classical optimization, including standard mathematical formulations and commonly used solver paradigms such as Mixed Integer Linear Programming (MILP) and Satisfiability Modulo Theories (SMT). The distinction between hard and soft constraints is discussed, along with the structure of multi-objective functions and how trade-offs are handled in real-world applications. The chapter then introduces key concepts in quantum computing, explaining how qubits differ from classical bits through phenomena such as superposition and entanglement, and how these are represented mathematically using Hilbert spaces and geometrically via the Bloch sphere. Fundamental components of quantum computation—such as Pauli operators, quantum gates, and quantum circuits—are introduced to establish a basis for understanding quantum algorithms. The final part of the chapter focuses on quantum optimization, describing the Quadratic Unconstrained Binary Optimization (QUBO) framework, its reformulation into a cost Hamiltonian suitable for quantum circuits, and the Quantum Approximate Optimization Algorithm (QAOA), which serves as the quantum approach to solving the physician scheduling problem in this work.

2.1 Foundations of Classical Optimization

Optimization is, at its core, a mathematical process of seeking the best solution within a set of feasible solutions [32]. It provides structured methodologies to solve problems defined by three primary components: objective functions, decision variables, and constraints [33]. Mathematically, a general optimization problem can be formulated as

$$\min_{x} \quad f(x) \quad \text{subject to} \quad g_i(x) \leq 0, \quad h_j(x) = 0,$$

where $x \in \mathbb{R}^n$ denotes the decision variables, $f(x) : \mathbb{R}^n \to \mathbb{R}$ is the objective function to be minimized, and $g_i(x)$ and $h_j(x)$ represent inequality and equality constraint functions respectively [34].

A feasible solution must satisfy all the constraints, while an optimal solution additionally minimizes or maximizes the objective function. In many practical applica-

tions, strict feasibility may be difficult to achieve, requiring relaxation mechanisms or trade-off strategies between different goals.

2.1.1 Mixed Integer Linear Programming

Mixed Integer Linear Programming (MILP) is a subclass of optimization problems that has linear constraints and objective function, and restricts some of the variables to be integers [35]. Formally, an MILP problem is expressed as

$$\min_x \quad c^T x \quad \text{subject to} \quad Ax \leq b, \quad x_i \in \mathbb{Z} \text{ or } x_i \in \mathbb{R}.$$

Here, $c \in \mathbb{R}^n$ is the cost vector, $A \in \mathbb{R}^{m \times n}$ is the constraint coefficient matrix, and $b \in \mathbb{R}^m$ is the constraint right-hand side vector. A special case of MILP is the binary integer program, where decision variables satisfy $x_i \in \{0, 1\}$. Binary MILP models are particularly important for assignment problems, where each variable represents a yes-or-no decision, such as whether a worker is assigned to a task or not.

MILP models are known to be NP-hard, meaning that there is no known polynomial-time algorithm that solves all problem instances optimally. Despite this theoretical difficulty, modern solvers can often handle MILPs of practical size very efficiently. State-of-the-art solvers, such as Gurobi, use a combination of algorithmic techniques to solve MILPs to optimality or near-optimality [36].

2.1.2 Constraint Programming and SMT Optimization

In CP, relationships between variables are defined by constraints, and variable assignments are made to satisfy these constraints [37]. Emphasis is placed on feasibility rather than explicit cost optimization in CP. A related approach is SMT, which builds on boolean satisfiability by adding support for things like arithmetic, arrays, and other mathematical rules [38]. This allows SMT solvers to handle more complicated constraints that are hard to describe using only linear equations. A recent study [39] shows that SMT solvers like Z3 can outperform MILP solvers like Gurobi in less constrained personnel scheduling scenarios, although Gurobi performs better on highly constrained or infeasible instances.

2.1.3 Constraint Classes and Objective Structures

In classical optimization, there are usually two kinds of constraints: hard constraints and soft constraints [40]. Hard constraints are strict rules that must always be followed for a solution to be valid. Soft constraints, on the other hand, describe things that are preferred but not required. These can be broken if necessary, but doing so results in a penalty.

The objective function describes what the solver should try to optimize [32]. It might aim to minimize cost, distribute work evenly, or match personal preferences as much as possible. In many real-world problems, the objective function includes several different goals at once. These goals are combined into one function using

weights, which reflect how important each goal is. The solver then looks for a solution that gives the best overall result according to this combined score.

2.2 Introduction to Quantum Computing

As computer technology evolves, the pursuit of constructing computers with increasingly small fundamental components continues [41]. In classical computers, these fundamental components are called transistors, and can be of the size of only a few atoms. However, when transistors reach sizes that are even smaller than atoms, the classical rules of physics no longer apply [42]. Instead, quantum physics takes over, introducing phenomena like superposition and entanglement. QC uses these phenomena to enable faster and more efficient processing, unlocking new possibilities in computation.

2.2.1 Qubits: Fundamental Units of Quantum Information

The *qubit*, short for quantum bit, is the quantum counterpart to the classical bit and expands the binary notation of 0 and 1 to the possibility of being a combination of these states, a property fundamental to quantum mechanics and a key difference between quantum systems and classical systems [42]. The most common way to represent the qubit is using the bra-ket notation, also known as the Dirac notation:

$$\langle bra|ket\rangle = \langle\psi|\phi\rangle$$

where ket, $|\phi\rangle$, represents the state of the quantum system and is a vector in a complex vector space, and bra, $\langle\psi|$, is its dual corresponding to the Hermitian conjugate, or complex conjugate transpose, of the ket [43].

For a two-dimensional quantum system, such as a single qubit, the relationship between the bra and ket can be expressed as:

$$c_1|\alpha_1\rangle + c_2|\alpha_2\rangle \Longleftrightarrow c_1^*\langle\alpha_1| + c_2^*\langle\alpha_2| \tag{2.1}$$

where c_i are complex coefficients (probability amplitudes) and the asterisk denotes the complex conjugate [42].

2.2.2 Superposition and Measurement

As mentioned above, what differentiates qubits from classical bits the most is their ability to be in two states at once, a phenomenon known as superposition [42]. Once they are measured, they immediately collapse into one of the two states, with probabilities determined by their superposition. This unique property allows a qubit to store much more information than a classical bit.

A qubit can be mathematically represented as a linear combination of two orthonormal basis states, $|0\rangle$ and $|1\rangle$, which form the computational basis [43]:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{2.2}$$

where α and β are complex numbers satisfying the normalization condition:

$$|\alpha|^2 + |\beta|^2 = 1 \tag{2.3}$$

This ensures the total probability of measuring the qubit in one of the basis states is 1 [43]. Upon measurement, the probabilities of collapsing to $|0\rangle$ and $|1\rangle$ are $|\alpha|^2$ and $|\beta|^2$, respectively. This measurement process, known as wavefunction collapse, destroys the qubit's superposition.

Superposition and measurement can also be understood through the double-slit experiment, illustrated in Figure 2.1.

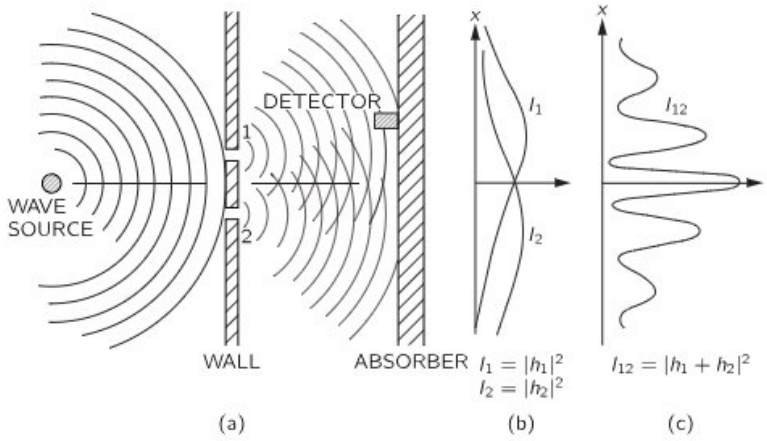


Figure 2.1: The double-slit experiment demonstrating superposition and interference. (a) Waves pass through two slits, creating interference. (b) Intensity distributions for single-slit openings. (c) Resulting interference pattern when both slits are open, demonstrating superposition. See Fig. 1-2 of [44].

In quantum mechanics, a qubit in superposition can be described as a combination of probability amplitudes. Measuring the qubit collapses this state, just as observing which slit the particle passes through collapses the interference pattern.

This ability of superposition makes qubits far more powerful than classical bits, as they encode probabilistic information rather than fixed values. By further exploiting interference and superposition, quantum algorithms can achieve significant computational advantages over classical approaches. Notable examples include Shor's algorithm, which efficiently factors large integers and threatens the security of classical cryptographic systems, and Grover's algorithm, which provides a quadratic speedup for unstructured search problems. However, quantum computations do not produce deterministic results in a single run; instead, they yield probabilistic outcomes that require repeated sampling to extract meaningful solutions. Moreover, their practical usefulness is still limited by noise, decoherence, and the difficulty of scaling quantum hardware.

2.2.3 Hilbert Space and Inner Product

Qubits exist in a Hilbert space, $\mathcal{H}$, which is a complex vector space with an inner product [43]. This inner product allows calculations of overlaps and probabilities. For any two states $|\psi\rangle$ and $|\phi\rangle$, the inner product is:

$$\langle\psi|\phi\rangle = c \tag{2.4}$$

where c is a possibly complex number. This inner product provides a measure of the overlap between the two states. If the states are orthogonal, the inner product is zero.

2.2.4 The Bloch Sphere: A Geometric Representation

A qubit can also be visualized geometrically using the Bloch sphere [42], where the state of $|0\rangle$ corresponds to the north pole, $|1\rangle$ corresponds to the south pole, and an arbitrary single-qubit state can be represented as points on the surface of the sphere [43]. With this, the state of a qubit can be written as:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{2.5}$$

where $\theta \in [0, \pi]$ determines the latitude of the sphere and $\phi \in [0, 2\pi]$ determines the longitude. This representation is particularly useful for visualizing quantum gates and quantum operations.

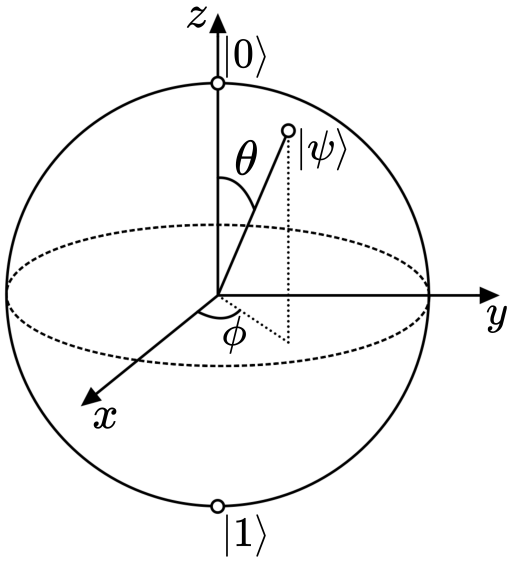


Figure 2.2: The Bloch sphere representation of a qubit state, illustrating the geometric visualization of the quantum state. The north pole corresponds to $|0\rangle$, the south pole to $|1\rangle$, and any point on the surface represents a superposition of these states. The angles θ and ϕ define the latitude and longitude of the state, respectively.

2.2.5 Pauli Operators

Quantum computing works by manipulating the states of the qubits, which can be visualized as a rotation in the bloch sphere. These manipulations can be expressed in terms Pauli operators, and the three fundamental types of Pauli operators are Pauli-X, Pauli-Y and Pauli-Z operators[43]. These unitary operators can be expressed in matrix form

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

For interaction between two qubits, the tensor product of two Pauli operators can act on both of them, similar to how a regular logical gate can act on two regular bits. A commonly used such tensor product is the Pauli-ZZ operator, defined by

$$Z \otimes Z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In this study, the operator $\otimes$ denotes the tensor product between two vectors or matrices. The Pauli-ZZ operator measures whether two qubits are in the same state or in different states, resulting in +1 or -1, respectively.

2.2.6 Multi-Qubit Systems and Entanglement

For systems with more than one qubit, the combined state is described by the tensor product of the individual qubits [42]. For two qubits, the state resides in a four-dimensional Hilbert space, $\mathbb{C}^4$:

$$\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\} \tag{2.6}$$

Entanglement describes the phenomenon where the quantum states of two or more particles become interdependent, i.e. the state of one particle cannot be fully described without referencing the state of the other(s), regardless of the physical distance between them [42]. Mathematically, this is expressed as:

$$|\psi_{AB}\rangle \neq |\psi_A\rangle \otimes |\psi_B\rangle \tag{2.7}$$

where $|\psi_{AB}\rangle$ is the joint state of two particles A and B, and the inequality signifies that the state is not separable into individual states of $|\psi_A\rangle$ and $|\psi_B\rangle$.

Entanglement creates complex correlations between qubits that are essential for quantum parallelism and quantum error correction [42]. Quantum parallelism refers to a quantum system's ability to process multiple states simultaneously. For instance, when two qubits are entangled, their joint state represents all possible combinations of their basis states at once. This allows quantum computers to operate over an exponentially large state space compared to classical systems. However, it is not entanglement alone that provides quantum advantage. It is the combination

of superposition, entanglement, and quantum interference that enables certain algorithms to solve problems more efficiently than their classical counterparts. Quantum computations are inherently probabilistic, and their results must often be sampled repeatedly to extract meaningful or high-confidence outcomes. These features highlight both the potential and the limitations of current quantum systems.

2.2.7 Quantum Gates and Circuits

In classical computing, logical operations are performed using classical gates, e.g. AND, OR, NOT, which manipulate bits. Similarly, QC uses quantum gates, which operate on qubits by transforming their quantum states [43]. Unlike classical gates, many quantum gates are reversible and are represented mathematically by unitary matrices.

A quantum gate transforms the state of a qubit by applying a specific operation, corresponding to one or a linear combination of the Pauli operators. If the initial state of the qubit is given by Eq. 2.2, the gate performs a unitary operation U, resulting in the new state:

$$|\psi'\rangle = U|\psi\rangle. \tag{2.8}$$

Here, U is a unitary matrix, meaning it satisfies $U^\dagger U = I$, where $U^\dagger$ denotes the Hermitian adjoint (or conjugate transpose) of U. This condition ensures that the transformation is reversible and preserves the total probability, which is a fundamental requirement for all quantum operations.

Quantum circuits consist of quantum gates arranged in a sequence or network to perform computations. A quantum circuit starts with an initial state of qubits, typically $|0\rangle^{\otimes n}$ for n qubits, applies a series of quantum gates, and concludes with a measurement to extract the results.

2.2.8 Noise and Circuit Depth

As previously described, entanglement is one of the strengths of quantum computing, since it allows faster computing through quantum parallelism. However, the quantum hardware that exists today is limited in connectivity, meaning that the number of connections between qubits is limited. Consequently, ideal problems for current quantum circuits are therefore sparse, with few variable interactions [45].

Another challenging aspect of qubit interactions is the existence of SWAP gates, which are used in cases where connections between two qubits that are not adjacent in the physical circuit, are required[46]. These SWAP gates work by communicating state information between non-neighboring qubits, requiring an additional step that is vulnerable to physical disturbances, here referred to as *noise*.

In this study, circuit depth refers to the total number of sequential operations performed on the qubits, and is a problem-specific metric. However, this theoretical circuit is distinct from the physical circuit, which refers to how the qubits are ar-

ranged and connected physically in the quantum hardware. A larger circuit depth means more qubit operations, and hence a greater risk of disturbances, or *noise*, in the physical system, affecting the result[45].

Circuit depth also correlates with the risk of qubit decoherence, which means that the superposition of the qubit is disrupted and the state collapses to the state $|0\rangle$ or $|1\rangle$, similar to the states of a regular bit [45]. This is also a type of error in the sense that a physical phenomenon interfere with the results, and is thus also categorized as noise.

2.3 Quantum Optimization

This section presents the mathematical foundation for formulating the physician scheduling problem as a quantum optimization task. It begins with an overview of the Quadratic Unconstrained Binary Optimization (QUBO) model, a widely used representation for combinatorial problems. The section then describes how QUBO problems are mapped onto quantum hardware through the cost Hamiltonian, using techniques inspired by the Ising model. Finally, it introduces the Quantum Approximate Optimization Algorithm (QAOA), a variational quantum algorithm used to solve the reformulated problem on both simulated and real quantum devices.

2.3.1 Quadratic Unconstrained Binary Optimization

QUBO stands for Quadratic Unconstrained Binary Optimization and is a standard mathematical framework for expressing combinatorial optimization problems when using quantum computing [47].

A QUBO problem formulation can be expressed as

$$f(\mathbf{x}) = \mathbf{x}^\top Q \mathbf{x} \tag{2.9}$$

where $\mathbf{x}$ is a vector containing the binary decision variables of the problem, and Q is an upper triangular matrix where each element represents the weight of a pairwise interaction between the decision variables in $\mathbf{x}$. The diagonal elements in Q are the linear coefficients of each decision variable. Each off-diagonal term, on the other hand, represents the interaction of two different decision variables and is active unless one of the decision variables is equal to zero. This pairwise interaction corresponds to the previously mentioned phenomenon called entanglement.

As the name suggests, the QUBO framework does not support constraints in the classical sense. Instead, Q encodes both the objective and the constraints by assigning a penalty term to each constraint and adding it to the objective.

2.3.2 Cost Hamiltonian

In QC, the problem objective is implemented using quantum gates. These gates consist of Pauli-Z operators, whose eigenvalues correspond to +1 and -1 for the $|0\rangle$ state and $|1\rangle$ state respectively. This reveals a fundamental mismatch between the QUBO formulation, where decision variables are binary, $x \in \{0, 1\}$, and the requirements of quantum circuits. Therefore, an alternative mapping of the problem is needed. In the QAOA framework, the objective must be expressed as an operator which is diagonal in the computational basis [8]. This format, often referred to as the cost Hamiltonian (H_C), encodes the optimization problem using Pauli operators with associated coefficients instead of a quadratic matrix.

To support this translation, the similarity between the QUBO framework and the Ising model is commonly exploited [48]. The Ising model, originally developed to describe atomic spin interactions on a lattice, bears a close structural resemblance to how qubits interact within a quantum circuit and thus provides a natural bridge for reformulating classical optimization problems into quantum-compatible forms.

In order to map the expanded QUBO sum:

$$x^\top Q x = \sum_i Q_{ii} x_i + \sum_{i<j} Q_{ij} x_i x_j$$

where $x_i \in \{0, 1\}$, to an Ising Hamiltonian with $z_i \in \{-1, 1\}$, the substitution of

$$x_i = \frac{1 - z_i}{2}$$

leading to

$$x_i x_j = \frac{(1 - z_i)(1 - z_j)}{4}$$

is performed. Here, z_i is a Pauli-Z operator acting on qubit i. This results in an Ising Hamiltonian

$$H_c = \sum_i c_i Z_i + \sum_{i<j} c_{ij} Z_i Z_j \tag{2.10}$$

where:

$$c_i = -\frac{Q_{ii}}{2} - \sum_{j \neq i} \frac{Q_{ij}}{4}, \quad c_{ij} = \frac{Q_{ij}}{4}.$$

2.3.3 Quantum Approximate Optimization Algorithm

Quantum Approximate Optimization Algorithm, or QAOA, is used for combinatorial optimization problems and utilizes the same phenomenon as Quantum Annealing (QA), namely the adiabatic theorem, which states “A physical system remains in its instantaneous eigenstate if a given perturbation is acting on it slowly enough and if there is a gap between the eigenvalue and the rest of the Hamiltonian’s spectrum” [8]. In QA, a simple Hamiltonian is initially used for exploration of the solution space. This general Hamiltonian is then gradually transformed into the cost

Hamiltonian H_C, containing the specific problem objective. QAOA uses a similar approach, but instead of gradually transforming the exploring Hamiltonian (here called mixer Hamiltonian H_M) into the cost Hamiltonian, QAOA uses a sequence of layers that each apply the two Hamiltonians alternately. The result is a Trotterized, or time-discretized, version of the QA process, where each layer corresponds to a discrete time step. Hence, the theoretical realism of the model increases with n_layers, although this is often not the case in practice due to the amount of noise also increasing with n_layers in quantum hardware [8].

Exploration of the solution space is managed by the mixer Hamiltonian H_M, while the cost Hamiltonian contains the objective function. The mixer Hamiltonian, sometimes referred to as B, is often set to QAOA is the X-basis mixer, which performs Pauli-X operations on the qubits [8]. Consequently, the mixer Hamiltonian introduces superposition to the problem since it rotates the amplitudes of the qubits, making it possible to exist in a state between $|0\rangle$ and $|1\rangle$.

Once the Hamiltonians are established, a theoretical quantum circuit called an *ansatz* can be constructed, with the mathematical notation:

$$|\phi(\boldsymbol{\beta},\boldsymbol{\gamma})\rangle = U_M(\beta_{n\_layers})\cdot U_C(\gamma_{n\_layers})\cdot \ ... \ \cdot U_M(\beta_2)\cdot U_C(\gamma_2)\cdot U_M(\beta_1)\cdot U_C(\gamma_1) \quad (2.11)$$

where

$$U_M(\beta_\ell) = e^{-i\beta_\ell H_M}, \quad U_C(\gamma_\ell) = e^{-i\gamma_\ell H_C}, \quad \ell \in \{1, .., n\_layers\}.$$

Each factor represents an operator enforcing one of the Hamiltonians in one of the circuit's layers, with layer-specific parameters $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. In turn, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ represent the angles of rotation for each operator to apply to the states of the qubits.

Unfortunately, the gates of the theoretical QAOA circuit are typically not directly transferable to a quantum circuit on the hardware, which has limitations in what sequences of gates can be applied. Hence, the circuit is replaced with one that is more hardware compatible by switching infeasible gates with a feasible sequence of corresponding gates, preserving the logic of the theoretical circuit. This procedure is called *transpilation* and is typically accomplished automatically using imported functions from libraries such as Qiskit [49]. As there can be several transpiled quantum circuits that each correspond to the same theoretical ansatz, this process is typically stochastic, yielding different solutions each time it is run.

The parameters $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are optimized with respect to the expectation value of the cost Hamiltonian given by the variational state prepared by QAOA. The expectation value is computed using the Estimator primitive in Qiskit, and can be expressed mathematically as

$$\langle C\rangle = \langle\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})|H_C|\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})\rangle \quad (2.12)$$

and its certainty depends on the number of solutions that are sampled.

An optimization algorithm that is commonly used for this angle optimization is COBYLA, short for Constrained Optimization by Linear Approximation, is [50]. This solver does not depend on gradients, which fits the nonlinear nature of QAOA problems. COBYLA is deterministic, although it heavily depends on the initial parameters in the context of QAOA, because of the non-convex character of the problem which often contains numerous local minima.

Once the parameters $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are set, solutions are sampled from the final circuit using the Sampler primitive in Qiskit. The result is a distribution of many solutions in the form of sequences of bits, so-called bitstrings, where each bit represents the binary value of a decision variable. In an ideal circuit where better solutions have higher probabilities, the best solution is found among the most frequently occurring solutions, given a sufficiently large number of samples.

3

Method

This chapter describes the methodology used to develop and evaluate classical and quantum optimization approaches for solving the Physician Scheduling Problem (PSP). It begins with the formulation of classical models using the Gurobi and Z3 solvers, including the definition of binary decision variables, a weighted objective function capturing soft constraints such as preference satisfaction, fairness, extent, and memory, and the specification of hard constraints related to demand fulfillment and physician availability. The quantum approach is then introduced, based on the Quantum Approximate Optimization Algorithm (QAOA), which reformulates the scheduling problem into a cost Hamiltonian derived from a QUBO representation. Due to hardware limitations, an iterative, shift-by-shift optimization scheme was adopted, incorporating dynamically updated fairness and extent measures. The chapter also outlines the transpilation and parameter optimization processes used for both simulated and real quantum hardware, as well as the postprocessing techniques applied to decode and assess the generated schedules. Finally, the evaluation procedure is explained, focusing on runtime, constraint satisfaction, and fairness metrics to enable a robust comparison between classical and quantum methods.

3.1 Data Collection and Processing

In order to construct a model of the PSP, various types of realistic exemplifying data were needed. A physician at Sahlgrenska who is involved in scheduling was therefore interviewed in this process to ensure that the model correspond to the real-life scheduling situation at Sahlgrenska, to the largest possible extent given the limitations of the project. From this guidance, synthetic data was automatically generated, and variables such as employment extent and personal shift preferences were set as shown in Table 3.1. The choice of using synthetic data was made in part for convenience, since the model was developed to handle increasingly complex rules over time, and partly for ethical reasons.

As the realism of the model was highly prioritized, real calendar dates were used, and information about which days were classified as Swedish holidays was gathered using the Python library Holidays [51]. To ensure there were feasible solutions to the problem, the number of required workers each shift was set based on the overall

number of workers and their work extent, so that these could be combined without contradicting each other.

Table 3.1: Overview of Physician Scheduling Information

Attribute	**Description**
Name	Common Swedish names were used as examples
Title	The used titles were ST, UL, ÖL and AT (see Appendix B for definitions)
Employment Extent	The possible extents were 50, 75 and 100%
Preferred Shifts	Individual for all workers, selected randomly from all scheduled dates
Unpreferred Shifts	Individual for all workers, selected randomly among non-preferred shifts
Unavailability	Individual for all workers, selected randomly among non-preferred shifts

In a real-life scheduling situation, the demand for physicians with a certain title can vary between situations and hospital wards. As a necessary simplification, the model assumed a demand equal to a fourth of the total demand, for each title, except for the title AT which had no specified demand. These numbers were however not based on real data, since they were thought of as customizable parameters to be set according to the specific real life situation the model would later be applied to.

A concern in the shift assignment is the scheduling of Swedish holidays such as Christmas Eve or Midsummer. Since these are celebrated by a majority of Swedish people, and thereby viewed as generally more attractive days off, the case where the same person is assigned all these holidays would not be fair. The solution was a parameter called $attractiveness_s$, calculated for shift $s \in S$, as:

$$attractiveness_s = \frac{prefer_s - prefernot_s}{demand_s + unavailable_s + 0.1} \tag{3.1}$$

where $prefer_s$ represents the number of workers who prefer the shift $s \in S$, $prefernot_s$ represents the number of workers who prefer not working, $demand_s$ the target number of physicians for that shift, and lastly, $unavailable_s$ represents the number of workers marked unavailable for that shift. The constant 0.1 ensures that zero division is avoided in all cases.

This way, the fairness in preference satisfaction could be assessed on a more advanced level than just using the number of satisfied preferences. Another advantage was that the fairness could be specifically tailored for each hospital ward, based on local preference patterns.

3.1.1 Two Test Designs

Since the project aimed to compare both the qualities of the scheduling solutions and the scalability, two different tests were designed. The first, henceforth called Month Scheduling, was to simply schedule 15 physicians from 1st to 28th of June, enabling comparison of to what extent the constraints were met in a scenario close to reality. The other test called Stress Test was designed to evaluate how the performances were affected by an increasing number of decision variables. In this test design, the cost distributions of the quantum solutions were also compared to the same number of randomly generated bitstrings, in order to determine whether the results were meaningful.

3.2 Classical Optimization Models

Classical optimization methods are widely used for solving complex scheduling problems, including the physician scheduling problem. These approaches rely on formulating the problem as a mathematical program that can be solved using constraint-based or mixed-integer optimization solvers. This section outlines the classical formulation used in this study and describes how constraints and scheduling goals are encoded through decision variables, an objective function, and associated penalties.

3.2.1 Gurobi and Z3 Model

The classical models used here are implemented with the Gurobi and Z3 solvers. Both models share the same structure and optimization formulation, where the physician scheduling problem is expressed as a constrained optimization task. The core of the model consists of binary decision variables indicating physician-shift assignments, an objective function that balances multiple soft constraints, and a set of hard constraints that must always be satisfied. The objective function includes penalty terms that quantify undesired properties of a schedule, such as unfair workload distributions or violations of stated preferences. By minimizing the total penalty, the solvers generate feasible and as fair schedules as possible under the given constraints.

3.2.1.1 Decision Variables

The decision variables are defined as

$$x_{p,s} \in \{0, 1\}, \quad \forall p \in \mathcal{P}, \quad \forall s \in \mathcal{S},$$

where $x_{p,s} = 1$ indicates that physician p from a set $\mathcal{P}$ of physicians is assigned to shift s from a set $\mathcal{S}$ of shifts, and $x_{p,s} = 0$ otherwise. All physician-shift pairs are instantiated as variables, regardless of physician preferences.

3.2.1.2 Objective Function

The objective is to minimize a weighted sum of penalty terms representing soft constraints:

$$\min \quad \lambda_{\text{pref}} C_{pref} + \lambda_{\text{fair}} C_{fair} + \lambda_{\text{memory}} C_{memory} + \lambda_{\text{extent}} C_{extent}$$

where

- C_{pref} is penalties for violating stated preferences
- C_{fair} is the penalties for unfair workload distribution
- C_{memory} is the penalty for overall assignment load used to regularize over-assignment
- C_{extent} is penalties for mismatches between assigned work and contracted extent

The penalty terms are defined as follows:

Preference Penalty

$$C_{pref} = \sum_{p,s} \begin{cases} -x_{p,s} & \text{if preference}_{p,s} = 1, \\ +x_{p,s} & \text{if preference}_{p,s} = -1, \\ 0 & \text{otherwise} \end{cases}$$

Here, a preference of 1 means that the shift is preferred, -1 is non-preferred, and 0 is neutral.

Fairness Penalty

The fairness penalty captures the deviation of each physician's shift count from the average. Let

$$u_p \geq |T_p - \bar{T}|,$$

where T_p is the total number of shifts assigned to physician p, and $\bar{T}$ is the mean number of shifts per physician. Then:

$$C_{\text{fair}} = \sum_{p} u_p$$

Memory Penalty

The memory term penalizes physicians proportionally to their cumulative number of assigned shifts, discouraging repeated over-assignment:

$$C_{memory} = \sum_{p \in \mathcal{P}} \sum_{s \in \mathcal{S}} x_{p,s}$$

Extent Penalty

The extent penalty is based on the physician's target work rate and the number of days passed at the shift s.

$$C_{extent} = \begin{cases} -\alpha_{p,s}\, x_{p,s} & \text{if work rate}_p < 1, \\ +\alpha_{p,s}\, x_{p,s} & \text{if work rate}_p \geq 1, \end{cases}$$

Where $work\ rate_p$ is a physician-specific parameter and equals their number of shifts worked divided by the target number of shifts, stated by their work extent. The target number of shifts is based on the assumption that an extent of 100% corresponds to five shifts per week.

In turn, $\alpha_{p,s}$ is given by:

$$\alpha_{p,s} = \min\left(\frac{\text{days passed}_s}{7},\ 1\right) |1 - \text{work rate}_p| \,.$$

and is a measure of how shift assignment to one individual should be prioritized compared to the others. This equation encourages dynamic adjustment of assignment pressure based on how far each physician is from their contractual work extent. The idea is to gently push the scheduling in the direction of balancing work effort across physicians over time.

If a physician's work rate is less than 1, it means they are currently under-assigned relative to their expected number of shifts. In this case, assigning them is rewarded by a negative penalty, thereby promoting their selection in upcoming shifts. Conversely, if their work rate is at or above 1, they are meeting or exceeding their expected workload, and any additional assignment results in a positive penalty.

The weighting factor $\alpha_{p,s}$ modulates the strength of this pressure based on two factors: (1) the deviation from the ideal workload ($|1-\text{work rate}_p|$), and (2) the progression of the schedule, increasing gradually with time via the term $\min\left(\frac{\text{days passed}_s}{7}, 1\right)$. This design ensures that early assignments are flexible, while fairness pressure increases as the schedule unfolds, allowing the model to correct imbalances incrementally without overreacting too early.

3.2.1.3 Constraints

The following hard constraints are imposed:

Demand Satisfaction and Unavailability

Each shift must be assigned exactly the required number of physicians. This is enforced using the following constraint:

$$\sum_p x_{p,s} = d_s, \quad \forall s \in S$$

where d_s denotes the required number of physicians for shift s.

To reduce model size, the decision variable $x_{p,s}$ is only created if physician p is available for shift s, which is indicated by $preference_{p,s} \neq -2$.

3.3 Quantum Optimization Framework

This section describes how the optimization problem is formulated and solved within a quantum framework, by explaining the chosen algorithm and the process of applying it. An iterative optimization approach to support schedules that cover longer time frames, is also specified as an adaptation to the specific problem.

3.3.1 Algorithm Selection

The selection of an appropriate quantum optimization algorithm is crucial to solving PSP efficiently and allowing a realistic and fair performance comparison with classical approaches. QAOA was selected because it supports combinatorial optimization problems and because of its hybrid quantum-classical nature. It works by applying layers that alternately enforce a mixer Hamiltonian and a cost Hamiltonian using sequences of quantum gates. Each layer has two parameters that are determined using classical optimization routines. The cost Hamiltonian, H_C, is derived directly from the QUBO formulation:

$$H_C = \sum_{p,s} W_{(p,s)}\, q_s q_s, \tag{3.2}$$

where $W_{(p,s)}$ encodes the weight of interactions between schedule assignments. The algorithm alternates between applying H_C and the X-basis mixer Hamiltonian, H_M, that encourages exploration of the solution space:

$$H_M = \sum_{p} \sigma_x^p. \tag{3.3}$$

3.3.2 Iterative Optimization

Because the current quantum hardware has limited capacity in the number of qubits they can handle effectively for this type of problem, the schedules are generated iteratively, shift by shift, instead of all shifts at once. This approach make it possible to yield a long enough schedule for the results to be meaningful in terms of fairness and employment extent, but it also requires the algorithm to remember shift assignments from earlier optimizations to ensure fairness over time. To this end, information regarding the number of shifts worked by each physician is divided by the target number of shifts given by their employment extent and the total scheduled time, in order to get a parameter called *work_rate*. A higher work rate mean a lower probability of being assigned shifts, and vice versa.

Another iteratively recorded parameter is the *satisfaction score*, to ensure long term fairness among the physicians. If a physician p either prefer of prefer to not work

the shift s, the shift attractiveness and a term called *self_weight* is added to their satisfaction score if their preference is respected. If the preference is not respected however, the same terms is instead subtracted from their satisfaction score. The term *self_weight* is a constant that represents the importance of the individual's own preferences compared to the collectively determined shift attractiveness. For shift assignment to unavailable physicians, the shift attractiveness and 5 times the *self_weight* is subtracted from their satisfaction score, since being scheduled despite being unavailable is considered very undesirable.

3.3.3 Objective Function

As previously described in Section 2.3.1, the QUBO format requires any constraints to be added as penalties to the objective function rather than acting as separate hard constraints. Hence, the objective function consists of the sum of each objective and constraint. The sums are constructed using symbolic expressions, where each symbol corresponds to one of the binary decision variables x_{ps}. Each constraint is multiplied by a penalty term λ , that manages their importance compared to the rest of the total sum. The demand constraint is formulated as

$$\lambda_{demand} \left(\sum_{s=1}^{S} (\text{demand}_s - \sum_{p=1}^{P} x_{ps})^2 \right),$$

where λ_{demand} is the penalty term for the demand constraint and demand_s is the target number of physicians working shift s. Similarly, the constraint demanding the correct number of physicians with each title has the form

$$\lambda_{titles} \Big(\sum_{s=1}^{S} (ST_s - (\text{demand}_{STs})^2 + (UL_s - \text{demand}_{ULs})^2 + (\text{ÖL}_s - \text{demand}_{\text{ÖL}s})^2) \Big)$$

where ST_s, UL_s and $\text{Ö}L_s$ represent the number of workers with that respective title at shift s. Here, the title AT is left out intentionally for some flexibility, as the total demand each shift can vary. When it comes to the preferences, a factor called priority_p is used to enforce the long term satisfaction fairness. This factor is determined by

$$\text{priority}_p = \min \left(1 - \frac{\text{satisfaction}_p}{\text{satisfaction}_{max}},\ \text{min\_priority} \right)$$

where $\text{satisfaction}_{max}$ refers to the satisfaction value of the most satisfied physician, min_priority is a constant and each satisfaction score has previously been shifted so that the smallest score is equal to one, avoiding zero division. If a physician p prefers shift s, the term

$$\lambda_{prefer}(\text{priority}_p \cdot x_{p,s}^2)$$

is subtracted from the constraint sum to reward these shift assignments, and conversely, if p prefers to not work s, the same term is instead added to the constraint sum. Similar to this approach, the unavailability constraint consists of terms

$$\lambda_{unavailable} x_{p,s}^2$$

for the variables $x_{p,s}$ representing shifts where p is unavailable.

Lastly, the constraint enforcing the employment extent of each physician is constructed using a similar priority parameter as the one used for the preference constraint, but this one is determined by

$$\text{priority}_p = |\text{extent}_{importance} \cdot (1 - work\_rate_p)|,$$

where $work\_rate_p$ is the previously defined *work_rate* for a certain physician p, and $\text{extent}_{importance}$ is set according to

$$\text{extent}_{importance} = \min\left(\frac{days\_passed}{7}, 1\right).$$

This parameter is implemented so that the extent constraint is less important the first days of the schedule, since the algorithm would otherwise schedule too many physicians in the beginning because of their initially low *work_rate* values. The parameter priority_p is later used for rewarding all shifts to physicians who have a $work\_rate < 1$ by subtracting the term

$$\lambda_{extent}\left(\sum_{s=1}^{S} \text{priority}_p \cdot x_{p,s}^2\right),$$

and the same term is instead added for physicians with $work\_rate > 1$, penalizing shift assignment to them.

The sum of penalized constraints and the problem objective now consists of terms containing a maximum of two decision variables and a coefficient. From this state, the coefficient of each term is set as the element in the QUBO matrix, representing the interaction of the two respective decision variables. Since any remaining constant term would have no effect on the optimization, it is neglected.

From the QUBO matrix, substitutions are made as described in Section 2.3.2, resulting in the corresponding cost Hamiltonian H_C.

3.3.4 Finding the Optimal Quantum Circuit

The number of layers in the *QAOA* ansatz is set to two, since more layers would increase the risk of noise, as described in Section 2.2.8. Another reason is the relatively high connectivity of the problem, which means that there are many variable interactions in the objective and constraints, resulting in a relatively dense Q matrix, from Eq. 2.9. As described in Section 2.2.8, high connectivity increases the risk of needing to use SWAP gates, which in turn also increases the risk of noise.

This ansatz is implemented on both a quantum hardware backend provided by *IBM Quantum* [52] , and a high performance quantum circuit simulator called *Qiskit Aer*, also developed by IBM [49].

3.3.5 Real Quantum Hardware

The transpilation of the QAOA ansatz into a quantum circuit is done using Qiskit's *PassManager*, which allows custom optimization for circuit transformation and hardware adaptation[53]. Since this is a stochastic process, it is repeated ten times and the resulting circuit with the least number of two-qubit gates is chosen to mitigate noise from SWAP gates.

In an attempt to find the optimal circuit parameters $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, defined in Eq. 2.11, the *minimize* function from the *optimize* module in the Python library *SciPy* is used [54], in combination with the classical optimization algorithm COBYLA, presented in Section 2.3.3. As a consequence of the stochastic nature of the quantum circuit, the function values used for this minimization are expectation values generated by the Estimator primitive, given by Eq. 2.12.

As it was discovered that this approach had a tendency to end up in local minima, it is accompanied by an initial brute-force grid search using the function *brute*, also in the module optimize in SciPy [54]. In this procedure, the expected H_C costs of 16 combinations of initial parameters were estimated using the Estimator primitive, and the parameters with the lowest cost were used as initial $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ in the minimization process. Another modification made was ensuring that the chosen angles corresponded to the historically lowest cost, instead of always using the estimation that was made last.

3.3.6 Quantum Simulator Backend

The transpilation process for the quantum simulator is similar to the one for the quantum hardware, with the exception that the first circuit is used, rather than the best out of ten transpilation iterations. In the parameter optimization process of the simulated quantum circuit, the initial brute-force grid search is replaced by 20 sets of random initial parameters, from which the one that results in the lowest estimated cost is chosen.

3.3.7 Sampling Solutions

Once the circuit is transpiled and optimized, the Sampler primitive in Qiskit is used to sample 4000 solutions in the form of bitstrings. The result is a probability distribution where several solutions are sampled, some more frequently than others. The H_C-costs of the 50 most frequent solutions are compared, and the solution with the lowest corresponding H_C-cost is chosen and decoded from its bitstring format into an actual schedule.

3.3.8 Choice of Parameters

The parameter values used in the two test designs and for the various methods can be seen in Tables 3.2 - 3.5. For the Month Scheduling test, the number of physicians is set to 15, since this was the maximum capacity for the Z3 solver in one

optimization. The parameter *demand_total* is carefully chosen in consideration of the employment extents of the physicians, so that the demand satisfaction constraint is not contradicted by the soft constraint managing employment extents.

The quantum-specific parameters can be seen in Table 3.5, and a majority of these are set by testing a range of values and choosing the one that gave the most desirable results. For some parameters, this meant a trade-off between the computation time and solution quality. An example is the parameter *estimation_shots*, where a larger number gives more accurate expectation values, but also increases the computation time, especially since many estimates are required for each optimization.

Table 3.2: Universal Parameters

Name	**Value**	**Description**
λ_{demand}	3	Constraint penalties
λ_{pref}	5	
$\lambda_{\text{unavailable}}$	15	
λ_{extent}	8	
λ_{titles}	5	
$demand_{ST}$	$\frac{\text{demand}_s}{4}$	Title-Specific Shift Demands
$demand_{UL}$	$\frac{\text{demand}_s}{4}$	
$demand_{\text{ÖL}}$	$\frac{\text{demand}_s}{4}$	
$self\_weight$	1	Priority of p's preference in relation to $attractiveness_s$

Table 3.3: Parameters for the Month Scheduling test

Name	**Value**	**Description**
start_date	2025-06-01	Calendar dates
end_date	2025-06-28	(28 days including 10 holidays)
n_physicians	15	Total number of physicians
$demand_s$	Weekday: 11, Holiday: 5	Workers demanded on shift s

Table 3.4: Parameters for the Stress Test

Name	**Value**	**Description**
start_date	2025-06-22	Calendar dates
end_date	2025-06-28	(7 days including 2 holidays)
n_physicians	3, 4, 5, 6, 7, 10, 14	Increasing total number of physicians

Table 3.5: Quantum parameters

Name	Value	Description
n_layers	2	The number of layers in the QAOA circuits
$bounds_\beta$ $bounds_\gamma$	$(0, \pi)$ $(0, \frac{\pi}{2})$	The ranges of values explored in the angle optimization
Ns	Month Scheduling: 2 Stress Test:3	The resolution of the brute grid search
estimation_shots	Month Scheduling:2000 Stress Test:4000	The number of samples from which an average H_C cost is derived
sampling_shots	Month scheduling: 4000 Stress Test:100000	The number of solutions sampled from the final circuit
estimation_iterations	20	Number of random angle initializations used in Aer estimation process
n_candidates	50	Number of most frequent solutions to choose from
min_priority	0.01	Minimum preference priority

3.3.9 Postprocessing

From the resulting schedule, the satisfaction of constraints from both physician and shift coverage perspectives were evaluated. For each physician, the preference satisfaction was analyzed by the rate of their preferred shifts they were assigned, and the same for non-preferred shifts. To evaluate the fairness of the handling of the physicians' preferences in terms of the popularity of some specific dates, the satisfaction score (see Section 3.3.2), based on the total shift attractiveness, was used. The total number of hours each physician worked in comparison to their employment extent was also analyzed.

For shift coverage evaluation, the required number of workers per shift from each title was compared to the number of assigned workers from each title, resulting in two sums showing how many times too few respectively too many workers were assigned.

3.3.10 Evaluation

A challenging aspect of comparing the classical and quantum optimization methods is the difference in how they enforce constraints. This, in combination with their deterministic, respectively, stochastic natures, means that there is no obvious way to compare the methods using theoretical solution quality metrics. Instead, a comparison is made using several metrics related to what is considered important

in practice in a real hospital scenario. These included the total practical computation time, combined with how well the solution met certain aspects of the problem constraints. More specifically, for each schedule solution, the demand criteria is evaluated by computing the number of shifts where the wrong number of physicians were assigned. Similarly for the titles constraint, the sum of the deviations from the target values symbolizes the errors, and an average deviation over the titles is used. The averages of the previously defined preference satisfaction scores are also compared, and their variance indicate the fairness of the schedules. Employment extents were compared by averaging the distances from the work rates to 1, indicating the average error in percent. Lastly, the number of shift assignments to unavailable physicians were compared.

For the Stress Test, where multiple optimizations were made by each method for problems of increasing size, the H_C generated in the QAOA process was used as a metric, since the primary goal was to compare the quantum performance with randomly sampled solutions. The choice of including the H_C cost for the Gurobi solution was made to emphasize the difference in the problem formulations for the different methods, and to highlight how the results depend heavily on what metric is being used.

4

Results

This chapter presents the outcomes of applying classical and quantum optimization methods to the Physician Scheduling Problem (PSP). The evaluation focuses on key performance metrics, including feasibility, preference satisfaction, runtime, and memory usage for the classical solvers Z3 and Gurobi. For the quantum approach, results are reported for both a quantum simulator (Aer) and quantum hardware (IBM) using QAOA, with a focus on circuit properties, constraint satisfaction, and H_C-cost performance. Two test designs were used to assess the methods: a realistic month-long scheduling scenario and a Stress Test exploring scalability. The findings are presented to enable comparison across solvers and to highlight the current capabilities and limitations of quantum optimization in practical scheduling applications.

4.1 Classical Optimization Solvers

This section presents the outcomes of applying classical constraint-based optimization to the PSP using two solvers: Z3, an SMT-based solver, and Gurobi, a commercial mixed integer programming solver. Both solvers were evaluated with identical input data, incorporating shift demand, physician availability, and expressed preferences.

4.1.1 Feasibility and Constraint Satisfaction

Figure 4.1 illustrates the resulting schedules generated by the Gurobi (left) and Z3 (right) solvers for 14 days with 10 physicians. Each cell corresponds to a binary decision variable $x_{p,s}$, indicating whether physician p is assigned to shift s. The cell is grey when scheduled and white otherwise.

Both solvers successfully produced feasible schedules that satisfy all hard constraints. Every shift is covered in accordance with the specified demand, and no physician is assigned during periods when they are marked as unavailable. Although the schedules are structurally similar and fulfill the same set of constraints, a closer inspection reveals distinct differences between the solutions. These discrepancies arise because the problem admits many valid configurations that yield similar objective

values. The Gurobi and Z3 solvers rely on different internal heuristics and decision strategies, which naturally lead to variations in the final assignment.

For example, in columns five and eleven, there are different physicians assigned to a shift that they prefer not to be scheduled for. Nevertheless, both solvers manage to satisfy all positively preferred shifts, as indicated by the light green cells, while avoiding all shifts marked as unavailable in red. In both cases, two negatively preferred (pink) shifts are assigned, although they are distributed differently across physicians.

These distinctions emphasize that the scheduling problem is under-determined, allowing for multiple solutions of comparable quality. The final outcome depends not only on the model and constraints but also on solver-specific choices such as traversal order, tie-breaking mechanisms, and the handling of soft constraints. While feasibility is a minimum requirement, this comparison shows that the qualitative aspects of a schedule, such as the distribution of undesirable shifts and individual physician satisfaction, can vary between solvers, and should be taken into account when evaluating scheduling outcomes.

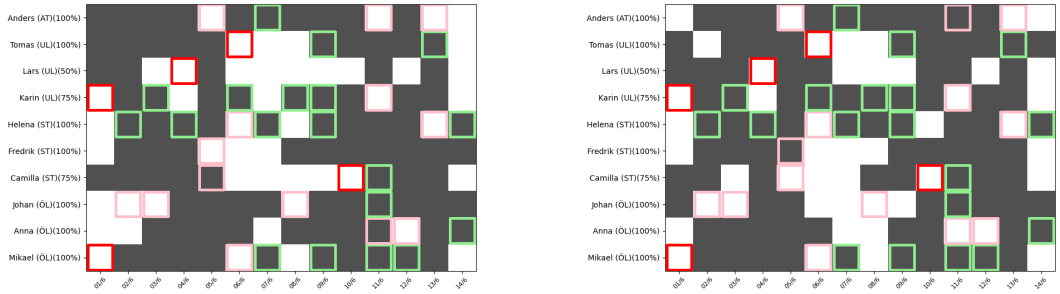


Figure 4.1: Schedules for 10 physicians over 14 days produced by Gurobi (left) and Z3 (right). Each row represents a physician and each column a shift. Gray cells indicate assigned shifts. Color-coded borders indicate preferences and constraints.

4.1.2 Preference Satisfaction

Physician shift preferences were modeled as soft constraints in both solvers, influencing the objective but not enforced strictly. Table 4.1 summarizes the percentage of positively preferred shifts that were granted, and negatively preferred shifts that were successfully avoided.

Solver	Preferred Respected (%)	Not Preferred Respected (%)
Gurobi	100.0	85.7
Z3	100.0	85.7

Table 4.1: Physician preferences fulfilled in a 10-physician, 14-day schedule.

Both solvers demonstrated excellent compliance with expressed preferences. All positively preferred shifts were fully respected by both Z3 and Gurobi. In the few instances where negatively preferred shifts were assigned, this only occurred when hard constraints made such assignments unavoidable. This indicates that both Z3 and Gurobi are capable of enforcing preference satisfaction effectively within feasible bounds.

Importantly, the few cases in which a negatively preferred shift was assigned occurred only when hard constraints such as shift demand and physician availability made it unavoidable. In these instances, the number of physicians either available or without a negative preference for a given shift was insufficient to meet the required staffing level. This indicates that both solvers respected preferences as much as possible within the feasible space defined by hard constraints.

4.1.3 Runtime Performance

Solver performance was evaluated in terms of both optimization time and total runtime, which includes data preparation and output processing. Table 4.2 shows that Gurobi significantly outperforms Z3 in terms of runtime.

Solver	Optimization Time (s)	Total Runtime (s)
Gurobi	0.0010	0.0573
Z3	2443.6573	2443.7421

Table 4.2: Solver time and total runtime for Gurobi and Z3 (10 physicians, 14 days).

As shown in Figure 4.2, the solver time required by Z3 increases sharply as the number of physicians grows. For smaller problem sizes, up to and including 7 physicians, both Z3 and Gurobi show relatively low and comparable runtimes. However, from 8 physicians onward, a clear divergence emerges. Z3's runtime begins to rise steeply. In contrast, Gurobi maintains a consistent performance across all tested problem sizes, with solver times remaining close to zero seconds throughout the entire range from 3 to 10 physicians.

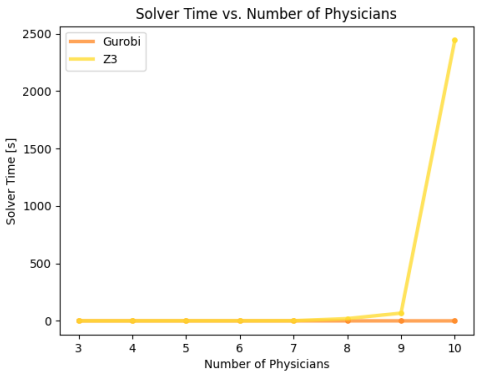


Figure 4.2: Solver time vs. number of physicians for Gurobi and Z3. Full data can be found in Table A.1 and A.2 in Appendix A

4.1.4 Memory Utilization

Memory usage was profiled by measuring the peak resident set size (RSS) during optimization. Table 4.3 and Figure 4.3 illustrate the memory footprint of each solver.

Solver	Peak Memory Usage (MB)
Gurobi	170.43
Z3	222.50

Table 4.3: Peak memory usage for Gurobi and Z3 (10 physicians, 14 days).

Figure 4.3 presents the peak memory usage of the Z3 and Gurobi solvers for problem sizes between 3-10 physicians. Across all instances, Gurobi maintains a nearly constant memory, with peak usage remaining close to 170 MB regardless of the number of physicians. Z3, however, shows a gradual but consistent increase in peak memory consumption as the number of physicians increases. Starting at approximately 181 MB, memory usage climbs to over 220 MB by the end. The difference in memory behavior between the two solvers becomes more pronounced as the problem size grows.

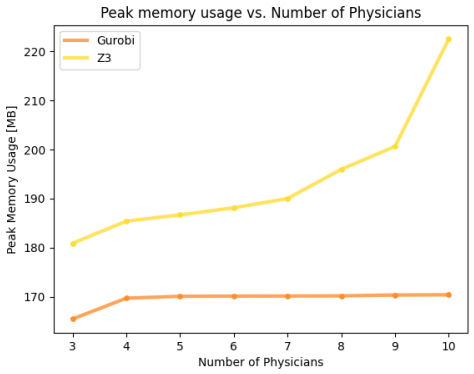


Figure 4.3: Peak memory usage vs. number of physicians for Z3 and Gurobi. Full data can be found in Table A.1 and A.2 in Appendix A

4.2 Classical and Quantum Optimization

The following section presents the results of applying three optimization approaches; the classical optimization solver Gurobi, quantum hardware provided by *IBM*, and Aer quantum simulator, to solve the PSP in two test designs. The methods are given the same input data in terms of calendar dates and the individual preferences, availability, employment extent, and titles of the physicians.

4.2.1 Month Scheduling

Figures 4.4 to 4.6 show the final schedules generated by the three methods for the month scheduling test. Each row corresponds to a physician and each column to a shift. Assigned shifts are indicated by gray tiles, while tiles with colored borders highlight constraints or preferences: red for unavailability, pink for negatively preferred shifts, and green for preferred shifts.

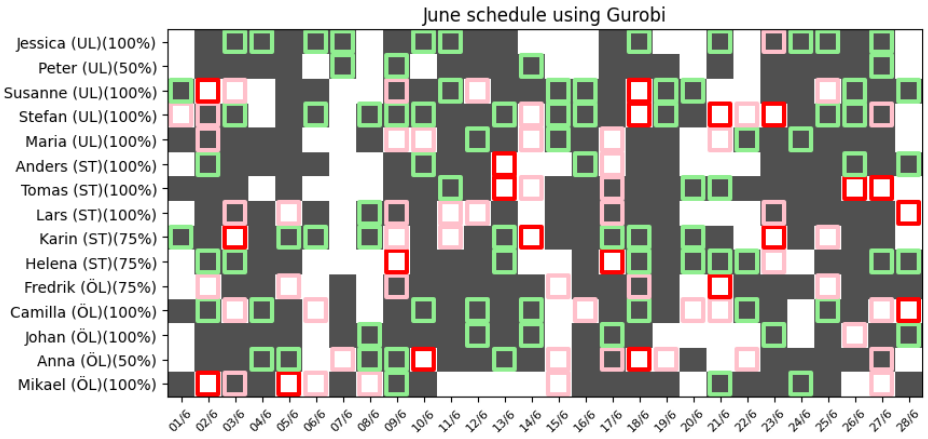


Figure 4.4: Final schedule generated by Gurobi.

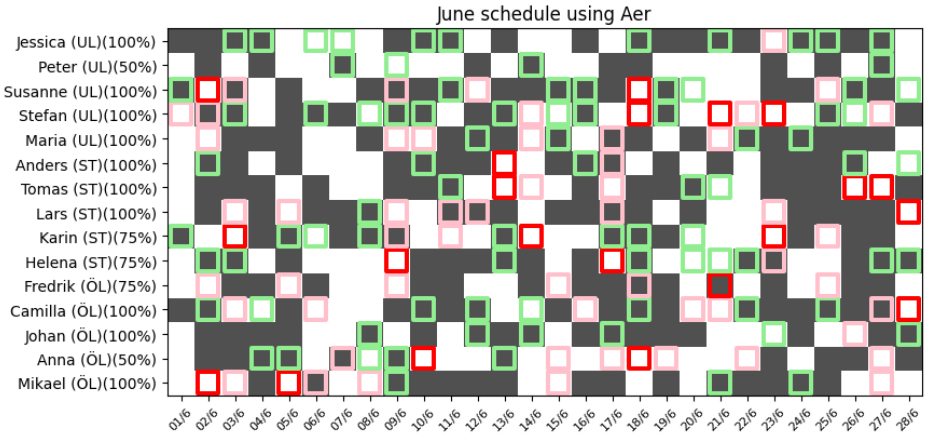


Figure 4.5: Final schedule generated by Aer quantum simulator.

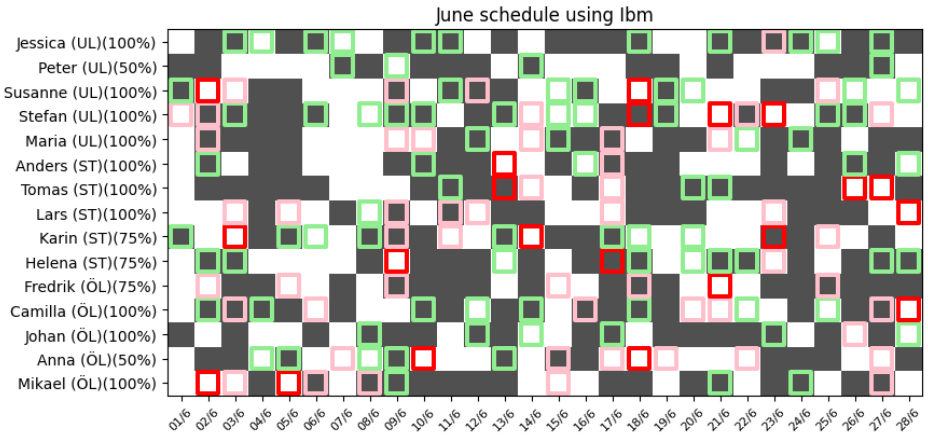


Figure 4.6: Final schedule generated by IBM quantum hardware.

Table 4.4 presents the computation times. Gurobi completed the optimization in 0.1 seconds. Aer required approximately 49 minutes, while IBM needed almost 10 hours. This highlights the computational overhead associated with quantum methods, particularly when executed on physical hardware.

Table 4.4: Computation times from the month scheduling test

Method	**Time [s]**
Aer	2946
IBM	35780
Gurobi	0.1

Only Gurobi was included in the classical versus quantum comparison for the month scheduling to ensure that the comparison reflects practically scalable and usable classical methods. When comparing the classical and quantum results, expected differences related to constraint handling became evident. Gurobi avoided infeasible solutions such as overstaffing, understaffing, or assigning shifts to unavailable physicians. The quantum methods, particularly IBM, showed weaker performance in these areas.

Table 4.5: Satisfaction of constraints for the month scheduling test

Metric	**Aer**	**Gurobi**	**IBM**
Average satisfaction score	5.2	8.6	1.6
Variance in satisfaction score	9.4	19.0	4.0
Too many workers	0	0	2
Too few workers	23	0	23
Employment extent error	14.7	21.9	20.2
Scheduling unavailable	1	0	4
Imbalance among titles	11	26.7	13.7

Although IBM achieved a lower variance in satisfaction score, suggesting greater fairness, it also produced the lowest average satisfaction. This was likely caused by assigning shifts to unavailable physicians, which substantially decreased the satisfaction score for the affected individuals. Aer outperformed IBM on most metrics, which is consistent with its execution in a noise-free simulated environment.

Table 4.6: Quantum circuit data from the month scheduling test

Metric	**Aer**	**IBM**
Circuit depth	48	1172
Two-qubit gates	210	899

The significant difference in circuit depth between Aer and IBM, as shown in Table 4.6, illustrates the difficulty of executing quantum algorithms on current hardware. IBM required deeper circuits and a larger number of two-qubit gates, both of which increase susceptibility to noise. These results confirm the quantum hardware incompatibility of the problem instance.

This hardware sensitivity arises because the physician scheduling problem involves dense variable interactions. Constraints such as shift demand and fairness require combining many variables, creating high entanglement and long-range interactions across qubits. These characteristics necessitate many SWAP gates and increase circuit depth, especially in the absence of a sparse or structured topology.

Furthermore, the quantum formulation does not cleanly separate objectives from constraints. While classical solvers like Gurobi treat constraints as hard boundaries, the QUBO framework integrates them as penalty terms within the objective function. This leads to a much broader theoretical solution space, though many of the solutions are strongly penalized and practically infeasible.

In addition, QUBO models cannot directly represent inequality constraints. Instead, penalties are placed symmetrically around a target value. This makes the system highly sensitive to how those targets are defined and may pull the optimization toward unrealistic values.

Another fundamental distinction is that quantum methods are stochastic and produce a distribution over solutions. In contrast, classical solvers are deterministic and return a single optimal solution.

Despite these differences and limitations, quantum methods produced favorable H_C values, which is expected since H_C defines the objective for QUBO-based methods. Gurobi, by contrast, was solving a different objective not aligned with H_C.

4.2.2 Stress Test

One of the most critical limitations of current quantum optimization methods is the challenge of scaling with respect to qubit count. Each binary decision variable in a QUBO formulation must be mapped to a physical qubit when executing QAOA on real quantum hardware. This mapping creates a direct relationship between problem size and hardware requirements, making scalability a primary bottleneck. The average H_C costs of the solutions achieved in the stress test can be seen in Figure 4.7, where the cost of the randomly generated bitstrings remains close to zero for all numbers of variables. IBM has varying performance over the different problem sizes, where a majority of the costs are slightly lower than the random sampling. Gurobi, on the other hand, has consistently lower costs than random sampling, despite the metric being based on the quadratic, quantum problem formulation. The Aer quantum simulator is only tested for 21 decision variables, since the used hardware did not allow it to handle larger problem sizes. However, at this size it outperforms all the other methods and reaches the lowest overall cost value.

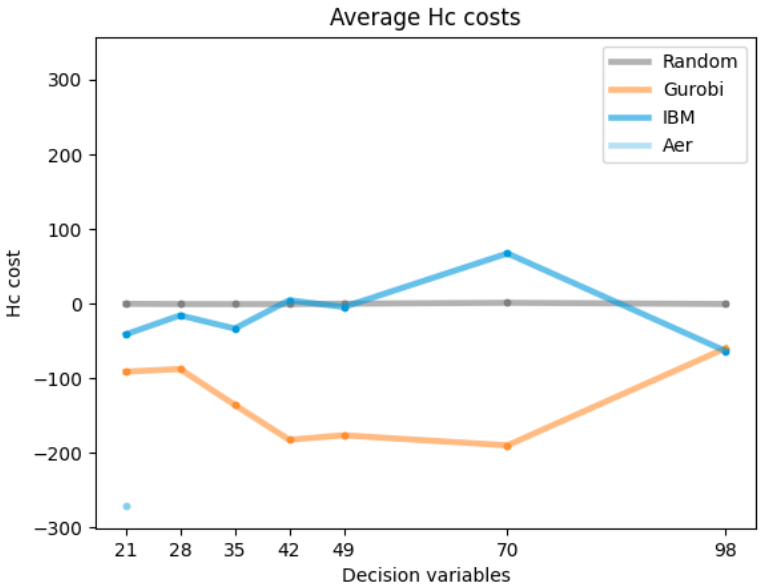


Figure 4.7: Average H_C-costs from "Stress test", with increasing number of variables.

The probability distribution of H_C-costs generated randomly and by IBM are given in Figure 4.8, for one of the problem sizes. This exemplifies how the distributions tend to compare to each other for different problem sizes and shows how IBM's performance is relatively similar to the results of random sampling. The classical Gurobi solver outperforms both in terms of average H_C-cost.

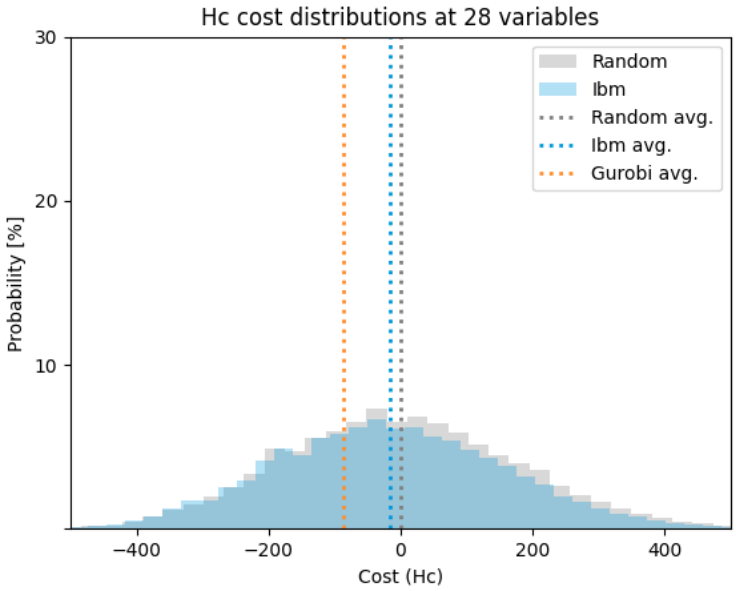


Figure 4.8: Probability distributions of the H_C-costs of the solutions using random solutions and IBM, for 28 decision variables.

Figure 4.9 illustrates that computation times grow rapidly with problem size when using IBM hardware, reaching up to over an hour for only 28 decision variables—still far from the scale required for real hospital scheduling scenarios. In contrast, Gurobi consistently takes less than a second, highlighting a stark performance gap.

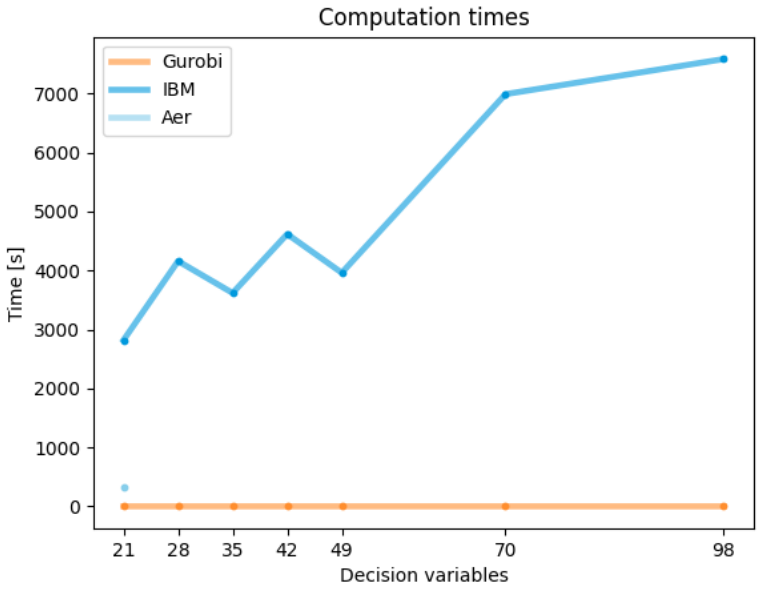


Figure 4.9: Computation times from the stress test, with increasing number of variables.

The reason for this decrease in performance lies in both the limited number of physical qubits and the increased circuit depth required to represent complex constraint interactions. As described in Section 2.3.2, the QUBO-to-Ising transformation introduces quadratic interactions between decision variables, which are implemented as entangling gates between qubits. These gates are not only time-consuming but also prone to error, especially when qubits are not directly connected on the device. The need for SWAP gates to route interactions between distant qubits further increases circuit depth and noise.

As discussed in Section 2.2.8, the level of noise in a quantum circuit is strongly influenced by the number of SWAP gates and the overall circuit depth. These factors are in turn determined by how many pairs of decision variables interact in the optimization problem. In the case of the PSP, many constraints, such as demand satisfaction and fairness, create dense interactions across variables. For instance, the demand constraint requires summing over all physicians for each shift, leading to widespread entanglement. Furthermore, the problem does not exhibit a sparse, graph-like structure where qubits interact locally, exacerbating the need for SWAP gates. These properties increase the circuit depth and vulnerability to noise, reducing quantum compatibility and negatively affecting performance on real hardware.

This challenge is also reflected in Table 4.6, where the quantum circuit executed on IBM hardware reaches a depth of 1172 and uses 899 two-qubit gates—an indication of substantial entanglement overhead. In contrast, the Aer simulator achieves comparable outcomes with much lower circuit complexity, highlighting how real hardware still lags behind idealized quantum simulations.

5

Discussion

This chapter reflects on the findings presented in the results and places them in the broader context of quantum and classical optimization for physician scheduling. The discussion is structured around key themes that emerged during the study, including quantum compatibility, solver performance, hardware constraints, and the practical implications of implementing quantum-based methods in real healthcare environments. Differences in modeling structure, constraint handling, and solution quality between classical and quantum approaches are critically analyzed. The chapter also addresses the scalability limitations of current quantum hardware and evaluates the feasibility of using such methods in realistic scheduling scenarios. Finally, practical challenges related to integration, interpretability, and operational use in clinical contexts are explored, followed by suggestions for future research directions.

5.1 Method and Performance Comparison

Chapter 4 evaluated and compared the performance of classical and quantum optimization methods for solving the Physician Scheduling Problem. The analysis included three solvers: the classical mixed integer solver Gurobi, the satisfiability modulo theories solver Z3, and two quantum approaches using IBM quantum hardware and the Aer simulator.

The results demonstrate that classical optimization remains the most effective and reliable method for generating feasible and high-quality schedules. Gurobi consistently produced schedules that fully satisfied all hard constraints, including physician availability and shift coverage, while also fulfilling a large proportion of soft preferences. Its computation time was extremely low and remained stable across different problem sizes. In contrast, Z3 was only practical for small instances. As the number of physicians increased, both its runtime and memory usage grew rapidly, which severely limited its scalability.

The quantum methods revealed important insights regarding the current state of quantum optimization for real-world problems. While the Aer simulator achieved favorable outcomes in terms of the cost function, the physical quantum hardware exhibited weaker performance. For instance, in the month scheduling experiment,

IBM produced a solution that required several hours to compute and still violated key constraints such as assigning shifts to unavailable physicians and leaving some shifts understaffed.

These limitations can largely be explained by the structure of the scheduling problem itself. The problem involves many interacting variables and overlapping constraints, which result in circuits with high depth and a large number of entangling operations. These characteristics increase the sensitivity of the quantum algorithm to noise and make execution on current hardware highly error prone. The comparison between IBM and Aer further illustrates this, since Aer simulates an ideal environment and is not affected by hardware-level noise.

Another important aspect is how the two optimization paradigms handle constraints. Classical solvers such as Gurobi enforce constraints explicitly and deterministically. In contrast, quantum methods based on the quadratic unconstrained binary optimization formulation represent constraints as penalty terms within the objective function. This approach makes constraint enforcement less precise and depends heavily on how penalty weights are chosen. Moreover, quantum solvers do not return a single optimal result but instead generate a distribution over possible solutions, which adds to the challenge of interpretation and reliability.

Despite these limitations, the results from the stress test suggest that quantum optimization could become more competitive in the future. At the largest tested problem size, IBM's performance in terms of the cost function began to resemble that of Gurobi. Although it required significantly more time, this indicates that quantum methods may scale differently than classical ones and could benefit from further improvements in hardware and parameter tuning.

5.2 Real-World Implementation Challenges

While quantum optimization methods such as QAOA demonstrate theoretical and experimental potential for solving constrained scheduling problems, several practical barriers remain before such techniques can be used in real-world hospital environments.

One major issue is integration. Scheduling in healthcare is embedded in a complex system of existing software platforms, including electronic health records, HR systems, and institutional planning tools. Quantum methods must be compatible with these systems in terms of data formats, reporting standards, and automation protocols.

Another important concern is data privacy. Physician schedules involve sensitive personal information related to work assignments, availability, and preferences. Healthcare institutions are subject to strict privacy laws, such as the General Data Protection Regulation (GDPR), which impose rigorous standards on how such data is handled. Since current quantum hardware is usually accessed remotely via cloud

services, secure data transmission and access control become critical. Without clear mechanisms to guarantee confidentiality and regulatory compliance, it would not be acceptable to handle real hospital data in this way.

Interpretability also poses a significant barrier to practical use. Scheduling decisions in hospitals must be explainable to staff, unions, and regulators. Classical solvers typically offer transparent outputs with clear constraint diagnostics, enabling administrators to understand and justify each assignment. In contrast, quantum algorithms produce results through non-deterministic sampling from complex cost functions. These solutions lack interpretability and cannot easily be audited or traced back to specific constraints, which undermines trust and limits practical value.

Even if technical integration were solved, quantum optimization would still require significant adaptation of hospital workflows. Most scheduling tools in use today are designed to support human users, often non-specialists, who must interact with the system through visual interfaces, adjust schedules manually, and understand the logic of suggestions or warnings. Current quantum optimization pipelines are not designed with this type of user in mind. Making them practically usable would require the development of new interfaces, robust fallback mechanisms, and the ability to incorporate user feedback.

Finally, today's quantum hardware has limitations that make it unsuitable for use in time-sensitive settings like hospitals. Issues such as unstable qubits, low accuracy in multi-qubit operations, and long waiting times to run programs cause uncertainty in the results. Even though the technology is improving, these problems still make it difficult to produce reliable and consistent schedules—especially when the number of staff or shifts is even moderately large.

5.3 Future Work

There are many possible directions in which this work can be extended, both in terms of improving the underlying optimization model and adapting it to more realistic and dynamic healthcare environments. As the results have shown, the current approach is capable of generating feasible and fairly high-quality schedules under constrained conditions. However, practical deployment in real hospitals would require additional functionality, greater flexibility, and deeper integration with actual decision-making workflows.

The following subsections outline several promising areas for future work. These include hybrid solution strategies that combine classical and quantum methods, adaptive learning of optimization priorities from historical data, localized rescheduling in response to disruptions, support for rolling long-term planning, and deeper exploration of quantum techniques and hardware integration. Each of these directions aims to increase the robustness, efficiency, and real-world applicability of the scheduling system.

5.3.1 Feasible-Then-Optimal Scheduling with Hybrid Solvers

One promising extension is to split the optimization process into two stages. In the first stage, a classical solver such as Z3 or Gurobi is then used to quickly produce a feasible schedule that satisfies all hard constraints, and in the second stage, a quantum solver can be used to improve the schedule by minimizing soft constraint violations. This two-phase method takes advantage of the speed and feasibility of classical solvers and the global search capabilities of quantum optimization. How to best link these stages should be explored further, as well as looking at whether quantum methods consistently improve upon classically feasible solutions.

5.3.2 Adaptive and Data-Driven Objective Weights

In the current system, weights are manually set to control the importance of different goals in the optimization, such as fairness, preference satisfaction, and memory. This manual tuning may not reflect what real hospitals actually care about, and it may not adapt well to different scheduling situations.

Future work could investigate learning these weights automatically from historical scheduling data using supervised machine learning. For example, if historical schedules show a tendency to prioritize preferences over fairness, the learned weights could reflect that trade-off. Another idea is to use reinforcement learning, where the system learns from previous optimization attempts to tune the weights dynamically based on outcomes (e.g., satisfaction scores or fairness metrics).

In addition, researchers could design the system to adjust its weights during the optimization process. For instance, if a solution starts to drift too far from fairness, the fairness penalty could automatically increase to correct the imbalance. This kind of adaptive optimization would allow the model to respond to the current state of the solution rather than relying on fixed priorities.

5.3.3 Local Rescheduling and Minimal Impact Optimization

In practice, hospitals often need to update only a small part of the schedule due to sudden changes, such as staff illness, emergency shifts, or changes in patient volume. Re-running a full optimization from scratch in such cases would be inefficient and potentially disruptive.

Future research should focus on local rescheduling methods that can make small changes to the schedule without altering unaffected parts. This could involve identifying a minimal subset of shifts and physicians affected by a change, and re-optimizing only that part of the schedule. One approach is to design algorithms that support incremental constraint solving, where constraints and variables can be added or removed without restarting the entire solver.

Another direction is to study the impact radius of changes; how far a change spreads through the schedule and which metrics (e.g., fairness or preference violations) are

most affected. Tools that can measure and control this propagation would make the system more robust and easier to use in real-world settings.

5.3.4 Longer Time Horizons and Rolling Optimization

The system currently focuses on short-term scheduling, often for a week or two at a time. However, hospitals plan weeks or even months in advance and continuously update these plans.

A future goal is to support rolling optimization, where schedules are updated over time as new data becomes available. This would require tracking staff fatigue, cumulative satisfaction and fairness over multiple weeks. Researchers should explore algorithms that can reuse previous optimization results to speed up the planning of future periods, reducing the computational burden.

It is also worth studying how long-term planning interacts with short-term rescheduling, for instance, whether local changes made today create larger imbalances later in the month.

5.3.5 Quantum Methods and Hardware Adaptation

While this project focuses on QAOA, other quantum optimization techniques could be valuable. For example, Variational Quantum Eigensolvers or Quantum Annealing could be explored to see if they offer better convergence for certain objective functions or constraint structures.

Researchers should also explore how to tailor the QUBO formulations to suit the capabilities of different quantum hardware backends, including noise-resilience, connectivity constraints, and gate depth limits. This will become increasingly important as quantum devices continue to improve but remain limited in qubit count and coherence time.

6

Conclusion

The goal of this thesis was to explore how Quantum Computing (QC), and in particular Quantum Approximate Optimization Algorithm (QAOA), can be used to solve the Physician Scheduling Problem (PSP). This is a complex and important task in healthcare, where many constraints need to be balanced such as staffing requirements, individual preferences, and fairness. To investigate this, a framework was developed that supports both classical and quantum optimization methods and allows for realistic scheduling scenarios.

The results show that classical solvers like Gurobi and Z3 perform well in generating feasible and fair schedules, as can be seen in section 4.1. Gurobi, in particular, was very fast and efficient, even as the size of the problem increased. Z3 offered greater flexibility in how the constraints were modeled but became too slow and memory intensive as the problem grew. The quantum approach using QAOA showed potential, especially in simulation. Schedules generated with the quantum simulator met several of the key constraints and performed reasonably well in terms of fairness and preference selection. However, when run on real quantum hardware, the quality of the solutions decreased due to noise and hardware limitations. In particular, the long runtime and large number of two-qubit gates made it difficult to scale the method to more realistic problem sizes.

Based on the findings, we conclude that quantum optimization is not yet ready to replace classical methods in real healthcare scheduling. However, the method has promise, and future improvements in quantum hardware and hybrid solver strategies could make it a useful tool. One interesting idea for future work is to first find a feasible solution using a classical solver and then use a quantum algorithm to improve the fairness or preference satisfaction. We also identified some important practical challenges. These include integrating quantum methods with hospital software systems, ensuring data privacy, and making the results understandable to hospital staff. Addressing these issues will be important for any future use of quantum computing in real healthcare settings.

In conclusion, this thesis shows that while classical methods are still the best choice for physician scheduling today, quantum optimization could become useful in the future. Our framework provides a first step toward that goal and can be used as a basis for further research as the technology continues to develop.

Bibliography

[1] R. Mansini and R. Zanotti, "Optimizing the physician scheduling problem in a large hospital ward," *Journal of Scheduling*, vol. 23, no. 3, pp. 337–361, Jun. 2020, ISSN: 1094-6136, 1099-1425. DOI: 10.1007/s10951-019-00614-w. [Online]. Available: http://link.springer.com/10.1007/s10951-019-00614-w (visited on 01/27/2025).

[2] H.-Z. Xu, J.-H. Chen, X.-C. Zhang, *et al.*, "High-speed train timetable optimization based on space–time network model and quantum simulator," *Quantum Information Processing*, vol. 22, no. 11, p. 418, Nov. 2023, ISSN: 1573-1332. DOI: 10.1007/s11128-023-04170-3. [Online]. Available: https://link.springer.com/10.1007/s11128-023-04170-3 (visited on 01/28/2025).

[3] A Gunawan and H. C. Lau, "Master physician scheduling problem," *Journal of the Operational Research Society*, vol. 64, no. 3, pp. 410–425, Mar. 2013, ISSN: 0160-5682, 1476-9360. DOI: 10.1057/jors.2012.48. [Online]. Available: https://www.tandfonline.com/doi/full/10.1057/jors.2012.48 (visited on 02/05/2025).

[4] M. Erhard, J. Schoenfelder, A. Fügener, and J. O. Brunner, "State of the art in physician scheduling," *European Journal of Operational Research*, vol. 265, no. 1, pp. 1–18, Feb. 2018, ISSN: 03772217. DOI: 10.1016/j.ejor.2017.06.037. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0377221717305787 (visited on 02/05/2025).

[5] N. Abdullah, M. Ayob, M. Chun Lam, N. R. Sabar, G. Kendall, and M. Khairulamirin Md Razali, "Optimization Techniques for Physician Scheduling Problem: A Systematic Review of Recent Advancements and Future Directions," *IEEE Access*, vol. 13, pp. 5203–5218, 2025, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3524599. [Online]. Available: https://ieeexplore.ieee.org/document/10819388/ (visited on 02/05/2025).

[6] O. G. Olanrewaju and M. Erkoc, "Physician scheduling for emergency telemedicine across multiple facilities," *IISE Transactions on Healthcare Systems Engineering*, vol. 13, no. 3, pp. 182–197, Jul. 2023, ISSN: 2472-5579, 2472-5587. DOI: 10.1080/24725579.2023.2201481. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/24725579.2023.2201481 (visited on 02/05/2025).

[7] H. A. Bakr, A. M. Salama, A. Fares, and A. B. Zaky, "Physician Scheduling using Constraint Satisfaction Programming," in *2023 11th International Japan-Africa Conference on Electronics, Communications, and Computations (JAC-*

ECC), Alexandria, Egypt: IEEE, Dec. 2023, pp. 218–223, ISBN: 9798350369328. DOI: 10.1109/JAC-ECC61002.2023.10479618. [Online]. Available: https://ieeexplore.ieee.org/document/10479618/ (visited on 02/05/2025).

[8] E. Farhi, J. Goldstone, and S. Gutmann, *A Quantum Approximate Optimization Algorithm*, 2014. DOI: 10.48550/ARXIV.1411.4028. [Online]. Available: https://arxiv.org/abs/1411.4028 (visited on 01/28/2025).

[9] K. Ikeda, Y. Nakamura, and T. S. Humble, "Application of Quantum Annealing to Nurse Scheduling Problem," *Scientific Reports*, vol. 9, no. 1, p. 12 837, Sep. 2019, ISSN: 2045-2322. DOI: 10.1038/s41598-019-49172-3. [Online]. Available: https://www.nature.com/articles/s41598-019-49172-3 (visited on 01/27/2025).

[10] Arindam Sadhu, Sahil Zaman, Kunal Das, Asmita Banerjee, and F. S. Khan, "Quantum annealing for solving a nurse-physician scheduling problem in Covid-19 clinics," 2020. DOI: 10.13140/RG.2.2.22952.80648. [Online]. Available: http://rgdoi.net/10.13140/RG.2.2.22952.80648 (visited on 01/27/2025).

[11] P. Vikstål, M. Grönkvist, M. Svensson, M. Andersson, G. Johansson, and G. Ferrini, "Applying the Quantum Approximate Optimization Algorithm to the Tail-Assignment Problem," *Physical Review Applied*, vol. 14, no. 3, p. 034 009, Sep. 2020, ISSN: 2331-7019. DOI: 10.1103/PhysRevApplied.14.034009. (visited on 01/27/2025).

[12] R. Shaydulin and Y. Alexeev, "Evaluating Quantum Approximate Optimization Algorithm: A Case Study," in *2019 Tenth International Green and Sustainable Computing Conference (IGSC)*, Oct. 2019, pp. 1–6. DOI: 10.1109/IGSC48788.2019.8957201. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8957201 (visited on 02/05/2025).

[13] J. Marshall, F. Wudarski, S. Hadfield, and T. Hogg, "Characterizing local noise in QAOA circuits," *IOP SciNotes*, vol. 1, no. 2, p. 025 208, Sep. 2020, ISSN: 2633-1357. DOI: 10.1088/2633-1357/abb0d7. [Online]. Available: https://iopscience.iop.org/article/10.1088/2633-1357/abb0d7 (visited on 02/05/2025).

[14] M. Streif and M. Leib, *Comparison of QAOA with Quantum and Simulated Annealing*, 2019. DOI: 10.48550/ARXIV.1901.01903. [Online]. Available: https://arxiv.org/abs/1901.01903 (visited on 02/05/2025).

[15] S. Brandhofer, D. Braun, V. Dehn, *et al.*, "Benchmarking the performance of portfolio optimization with QAOA," *Quantum Information Processing*, vol. 22, no. 1, p. 25, Dec. 2022, ISSN: 1573-1332. DOI: 10.1007/s11128-022-03766-5. [Online]. Available: https://doi.org/10.1007/s11128-022-03766-5 (visited on 02/05/2025).

[16] J. Van den Bergh, J. Beliën, P. De Bruecker, E. Demeulemeester, and L. De Boeck, "Personnel scheduling: A literature review," *European Journal of Operational Research*, vol. 226, no. 3, pp. 367–385, May 2013, ISSN: 0377-2217. DOI: 10.1016/j.ejor.2012.11.029. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221712008776 (visited on 04/02/2025).

[17] A. T Ernst, H Jiang, M Krishnamoorthy, and D Sier, "Staff scheduling and rostering: A review of applications, methods and models," *European Journal*

of Operational Research, Timetabling and Rostering, vol. 153, no. 1, pp. 3–27, Feb. 2004, ISSN: 0377-2217. DOI: 10.1016/S0377-2217(03)00095-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037722170300095X (visited on 04/02/2025).

[18] P. Brucker, R. Qu, and E. Burke, "Personnel scheduling: Models and complexity," *European Journal of Operational Research*, vol. 210, no. 3, pp. 467–473, May 2011, ISSN: 0377-2217. DOI: 10.1016/j.ejor.2010.11.017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221710007897 (visited on 04/02/2025).

[19] A. Caprara, M. Monaci, and P. Toth, "Models and algorithms for a staff scheduling problem," *Mathematical Programming*, vol. 98, no. 1, pp. 445–476, Sep. 2003, ISSN: 1436-4646. DOI: 10.1007/s10107-003-0413-7. [Online]. Available: https://doi.org/10.1007/s10107-003-0413-7 (visited on 04/02/2025).

[20] E. K. Burke, T. Curtois, G. Post, R. Qu, and B. Veltman, "A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem," *European Journal of Operational Research*, vol. 188, no. 2, pp. 330–341, Jul. 2008, ISSN: 0377-2217. DOI: 10.1016/j.ejor.2007.04.030. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221707004390 (visited on 04/02/2025).

[21] K. Socha, J. Knowles, and M. Sampels, "A MAX-MIN Ant System for the University Course Timetabling Problem," in *Ant Algorithms*, M. Dorigo, G. Di Caro, and M. Sampels, Eds., Berlin, Heidelberg: Springer, 2002, pp. 1–13, ISBN: 9783540457244. DOI: 10.1007/3-540-45724-0_1.

[22] G. Costa, "Shift work and occupational medicine: An overview," *Occupational Medicine (Oxford, England)*, vol. 53, no. 2, pp. 83–88, Mar. 2003, ISSN: 0962-7480. DOI: 10.1093/occmed/kqg045.

[23] M. Di Muzio, F. Reda, G. Diella, *et al.*, "Not only a Problem of Fatigue and Sleepiness: Changes in Psychomotor Performance in Italian Nurses across 8-h Rapidly Rotating Shifts," *Journal of Clinical Medicine*, vol. 8, no. 1, p. 47, Jan. 2019, ISSN: 2077-0383. DOI: 10.3390/jcm8010047.

[24] P. Ferri, M. Guadi, L. Marcheselli, S. Balduzzi, D. Magnani, and R. Di Lorenzo, "The impact of shift work on the psychological and physical health of nurses in a general hospital: A comparison between rotating night shifts and day shifts," *Risk Management and Healthcare Policy*, vol. 9, pp. 203–211, 2016, ISSN: 1179-1594. DOI: 10.2147/RMHP.S115326.

[25] A. S. Wagstaff and J.-A. Sigstad Lie, "Shift and night work and long working hours–a systematic review of safety implications," *Scandinavian Journal of Work, Environment & Health*, vol. 37, no. 3, pp. 173–185, May 2011, ISSN: 1795-990X. DOI: 10.5271/sjweh.3146.

[26] K. Karhula, M. Härmä, M. Sallinen, *et al.*, "Association of job strain with working hours, shift-dependent perceived workload, sleepiness and recovery," *Ergonomics*, vol. 56, no. 11, pp. 1640–1651, 2013, ISSN: 1366-5847. DOI: 10.1080/00140139.2013.837514.

[27] M. Goetschalckx and C. Jacobs-Blecha, "The vehicle routing problem with backhauls," *European Journal of Operational Research*, vol. 42, no. 1, pp. 39–

51, Sep. 1989, ISSN: 0377-2217. DOI: 10.1016/0377-2217(89)90057-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/037722178990057X (visited on 04/02/2025).

[28] M. Erhard, J. Schoenfelder, A. Fügener, and J. O. Brunner, "State of the art in physician scheduling," *European Journal of Operational Research*, vol. 265, no. 1, pp. 1–18, Feb. 2018, ISSN: 0377-2217. DOI: 10.1016/j.ejor.2017.06.037. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221717305787 (visited on 04/02/2025).

[29] S. Kraul, M. Erhard, and J. O. Brunner, "Optimizing physician schedules with resilient break assignments," *Omega*, vol. 129, p. 103 154, Dec. 2024, ISSN: 0305-0483. DOI: 10.1016/j.omega.2024.103154. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305048324001191 (visited on 04/02/2025).

[30] Sveriges läkarförbund, *Tid till vård ger vård i tid*, https://slf.se/app/uploads/2019/07/lakarrapport-rgb-digital.pdf, Accessed: 2025-04-02, Sveriges läkarförbund, 2019.

[31] K. Blekos, D. Brand, A. Ceschini, *et al.*, *A Review on Quantum Approximate Optimization Algorithm and its Variants*, arXiv:2306.09198, Jun. 2023. DOI: 10.48550/arXiv.2306.09198. [Online]. Available: http://arxiv.org/abs/2306.09198 (visited on 04/02/2025).

[32] S. Sieniutycz, "Systems design: Modeling, analysis, synthesis, and optimization," in *Complexity and Complex Thermo-Economic Systems*, Elsevier, 2020, pp. 103–104, ISBN: 9780128185940. DOI: 10.1016/B978-0-12-818594-0.00005-2. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B9780128185940000052 (visited on 02/06/2025).

[33] E. K. P. Chong, W.-S. Lu, and S. H. Żak, *An introduction to optimization: with applications to machine learning*, Fifth edition. Hoboken, New Jersey: Wiley, 2024, pp. 69,249, ISBN: 9781119877639.

[34] J. Nocedal and S. J. Wright, *Numerical optimization* (Springer series in operations research), 2nd. New York: Springer, 1999, ISBN: 9780387987934.

[35] *Mixed-Integer Programming (MIP) – A Primer on the Basics*. [Online]. Available: https://www.gurobi.com/resources/mixed-integer-programming-mip-a-primer-on-the-basics/ (visited on 04/08/2025).

[36] E. Castillo, A. J. Gonejo, P. Pedregal, R. Garciá, and N. Alguacil, *Building and Solving Mathematical Programming Models in Engineering and Science*, 1st ed. Wiley, Oct. 2001, ISBN: 9780471150435 9780471225294. DOI: 10.1002/9780471225294.

[37] K. R. Apt, *Principles of constraint programming*, Digitally print. version (with minor corr.) Cambridge: Cambridge Univ. Pr, 2009, ISBN: 978-0521825832.

[38] C. Barrett and C. Tinelli, "Satisfiability Modulo Theories," in *Handbook of Model Checking*, E. M. Clarke, T. A. Henzinger, H. Veith, and R. Bloem, Eds., Cham: Springer International Publishing, 2018, pp. 305–343, ISBN: 9783319105758. DOI: 10.1007/978-3-319-10575-8_11. [Online]. Available: https://doi.org/10.1007/978-3-319-10575-8_11 (visited on 04/08/2025).

[39] A. Combrink, S. Do, K. Bengtsson, S. F. Roselli, and M. Fabian, *A comparative study of SMT and MILP for the nurse rostering problem*, 2025. arXiv: 2505.10328 [cs.AI]. [Online]. Available: https://arxiv.org/abs/2505.10328.

[40] R. Martí, P. M. Pardalos, and M. G. C. Resende, Eds., *Handbook of Heuristics*. Cham: Springer International Publishing, 2018, ISBN: 9783319071237 9783319071244. DOI: 10.1007/978-3-319-07124-4. [Online]. Available: http://link.springer.com/10.1007/978-3-319-07124-4 (visited on 04/08/2025).

[41] S. Rashid, R. Shakeel, H. Bashir, K. Malik, and K. Wajib, "Moore's Law Effect on Transistors Evolution," *International Journal of Computer Applications Technology and Research*, pp. 495–499, Apr. 2023, ISSN: 23198656. DOI: 10.7753/IJCATR0507.1014. [Online]. Available: https://ijcat.com/archives/volume5/issue7/ijcatr05071014.pdf (visited on 03/03/2025).

[42] E. Rieffel and W. Polak, *Quantum computing: a gentle introduction* (Scientific and engineering computation), Online-Ausg. Cambridge, Mass: MIT Press, 2011, ISBN: 9780262015066 9780262295390.

[43] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, 10th anniversary edition. Cambridge: Cambridge university press, 2010, ISBN: 9781107002173.

[44] R. P. Feynman, R. B. Leighton, and M. L. Sands, *The Feynman lectures on physics*, New millennium ed. New York: Basic Books, 2011, ISBN: 978-0465023820.

[45] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, Aug. 2018, ISSN: 2521-327X. DOI: 10.22331/q-2018-08-06-79. [Online]. Available: https://doi.org/10.22331/q-2018-08-06-79.

[46] P. Murali, J. Baker, A. Javadi-Abhari, F. Chong, and M. Martonosi, "Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers," Apr. 2019. DOI: 10.1145/3297858.3304075.

[47] F. Glover, G. Kochenberger, and Y. Du, "Quantum Bridge Analytics I: a tutorial on formulating and using QUBO models," *4OR*, vol. 17, no. 4, 2019. DOI: 10.1007/s10288-019-00424-y.

[48] A. Lucas, "Ising formulations of many NP problems," *Frontiers in Physics*, vol. Volume 2 - 2014, 2014, ISSN: 2296-424X. DOI: 10.3389/fphy.2014.00005. [Online]. Available: https://www.frontiersin.org/journals/physics/articles/10.3389/fphy.2014.00005.

[49] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, *et al.*, *Qiskit: An open-source framework for quantum computing*, version 0.7.2, Jan. 2019. DOI: 10.5281/zenodo.2562111. [Online]. Available: https://doi.org/10.5281/zenodo.2562111.

[50] M. J. D. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Advances in Optimization and Numerical Analysis*, S. Gomez and J.-P. Hennart, Eds. Dordrecht: Springer Netherlands, 1994, pp. 51–67, ISBN: 978-94-015-8330-5. DOI: 10.1007/978-94-015-8330-5_4. [Online]. Available: https://doi.org/10.1007/978-94-015-8330-5_4.

[51] dr prodigy and contributors, *Python holidays library*, https://github.com/dr-prodigy/python-holidays, Accessed: 2025-06-08, 2024.

[52] IBM Quantum, *Ibm quantum experience*, https://quantum-computing.ibm.com/, 2023.

[53] Q. D. Team, *Qiskit terra: Preset passmanagers module*, https://qiskit.org/documentation/stubs/qiskit.transpiler.preset_passmanagers.html, Accessed: 2025-06-09, 2024.

[54] P. Virtanen, R. Gommers, T. E. Oliphant, *et al.*, "Scipy 1.0: Fundamental algorithms for scientific computing in python," *Nature Methods*, vol. 17, no. 3, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

A

Appendix

Table A.1: Performance of Gurobi solver with increasing number of physicians.

Physicians	Solver Time (s)	Total Time (s)	Memory (MB)
3	0.0030	0.0549	165.50
4	0.0010	0.0542	169.74
5	0.0020	0.0540	170.13
6	0.0024	0.0531	170.16
7	0.0020	0.0534	170.18
8	0.0010	0.0542	170.21
9	0.0020	0.0541	170.39
10	0.0010	0.0537	170.43

Table A.2: Performance of Z3 solver with increasing number of physicians.

Physicians	Solver Time (s)	Total Time (s)	Memory (MB)
3	0.0239	0.0689	180.91
4	0.1905	0.2214	185.42
5	0.4279	0.4798	186.69
6	0.4478	0.4852	188.14
7	1.1550	1.1991	190.02
8	19.6563	19.7161	195.97
9	67.1877	67.2457	200.63
10	2443.6573	2443.7421	222.50

B

Appendix

Table B.1: Descriptions of the Swedish physician titles that were used

Abbreviation	**Full Title**	**Description**
AT	Allmäntjänstgöring	Psysician after basic medical education who is under training to become fully licenced
ST	Specialisttjänstgöring	Licenced psysician under training to become a certified specialist
UL	Underläkare	Assistant psysician
ÖL	Överläkare	Head psysician

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY