THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Protein structure prediction and design on near-term quantum computers

HANNA LINN

Department of Microtechnology and Nanoscience (MC2)
Applied Quantum Physics Laboratory
Chalmers University of Technology
Göteborg, Sweden, 2025

Protein structure prediction and design on near-term quantum computers
HANNA LINN

ISBN 978-91-8103-305-2

Ny serie nr 5762
ISSN 0346-718X

Applied Quantum Physics Laboratory
Department of Microtechnology and Nanoscience (MC2)
Chalmers University of Technology
SE–412 96 Göteborg
Sweden
Telephone +46 (0)31-772 1000

Cover: Popular science doodles.

Chalmers Digitaltryck
Göteborg, Sweden, 2025

Protein structure prediction and design on near-term quantum computers
HANNA LINN
Department of Microtechnology and Nanoscience (MC2)
Applied Quantum Physics Laboratory
Chalmers University of Technology

Abstract

In the convergence of quantum computing and life science, we explore protein structure prediction and design on near-term intermediate-scale quantum devices. We investigate the algorithmic and resource constraints of today's quantum computers, aiming to assess their potential in solving biologically relevant problems. We describe key variational quantum algorithms, including the problem-informed Quantum Approximate Optimization Algorithm and the problem-agnostic Hardware-Efficient Ansatz. Additionally, quantum walks are examined. The computationally complex coarse-grained lattice models in protein structure prediction and design are discussed. Quantum algorithms are then applied to these models to address the utility and limitations of today's quantum computers. The thesis critically evaluates the limitations of quantum methods in comparison to classical approaches, highlighting the trade-offs between resource requirements in today's quantum devices and the performance of quantum algorithms. Through this interdisciplinary investigation, the work contributes to understanding how quantum algorithms may advance computational biology in today's quantum computing landscape.

Keywords: life science, protein folding, protein structure prediction, protein design, variational quantum algorithms, hardware-efficient ansatz, quantum approximate optimization algorithm, quantum walk, near-term intermediate-scale quantum devices

Acknowledgments

This journey would not have been possible without the amazing people who have supported, inspired, and challenged me along the way. It was an adventure, and I had the best company imaginable along the way.

First and foremost, I would like to express my deepest gratitude to my supervisors. Göran, thank you for your guidance and support, for fostering an environment where scientific excellence thrives, and for connecting us with brilliant collaborators. Our discussions not only taught me about science, but also how to be confident in what I do know and how to take the next step. Laura, thank you for your endless patience and encouragement that goes above and beyond. You have been both a colleague and a friend. You have a natural way of approaching science that makes it so easy to join in and love doing what we do. You have taught me a great deal about physics and how to work in science, as well as many other valuable life skills. You create a vibrant discussion space around quantum algorithms, making it easy for us in the community to grow and thrive.

I am forever grateful to the incredible people I've had the privilege to work with during my PhD. The group at Lund University, Lucas, Anders, Carsten, and Sandipan, thank you for the stimulating discussions and shared curiosity. Being able to ask you the questions that we did not know how to find the answers to has truly been amazing. Extra thank you to Anders, who also helped me proofread parts of this thesis. Ruihao, Bryan, and Dan, at the Cleveland Clinic, thank you for welcoming me into a fascinating world where quantum computing and life science meet in a hospital setting. My side quest to Cleveland was genuinely inspiring. The researchers at Sahlgrenska University, Vilma, Kristina, and Per, thank you for helping me begin my journey in this interdisciplinary field and for patiently explaining the life science aspects of it, as well as for answering all my endless questions. Ebba, your passion and engagement know no limits. You make things happen and bring people together. Further, I would like to thank the people at the Quantum Life Science Round Table for being a great community to be a part of. Thank you, Arseny and Anders at AstraZeneca, for the discussions around real-world challenges at the intersection of quantum computing and life science. My first contact with the field was during my master's thesis with Anton, thank you for welcoming me to Chalmers and for your continued support. Thank you, Isak, for joining

the life science journey and helping steer it toward our first publication. My examiner, Janine, thank you for your insightful guidance and for being a source of inspiration.

I sincerely appreciate the support from the wonderful people around me, both inside and outside the office. Thank you, Ariadna and Therese, for always being ready to listen, offer advice, and help me sort through my thoughts. My office mates, and the ever-changing environment at the Applied Quantum Physics Laboratory, thank you for all the fika times and for making me happy to come into the office every day. The people in and around Chalmers and WACQT, thank you for creating an environment where I could always ask a question, find support, or a laugh.

A heartfelt thank you to my family and friends outside academia for keeping me grounded and dancing along the way. A special thank you to my parents, Lena and Ulf, who sparked my curiosity to ask "how" and "why". Your love, support, and belief in me are boundless, and for that, I am forever grateful.

Finally, to Trygve, my lighthouse and my love. Thank you for always being there through this long and winding path. Your love, encouragement, and patience have made every step along this journey lighter.

Hanna Linn, Göteborg, October 2025

Publications

A **Resource analysis of quantum algorithms for coarse-grained protein folding models**
Hanna Linn, Isak Brundin, Laura García-Álvarez, and Göran Johansson
Phys. Rev. Research **6**, 033112 (2024)

B **Designing lattice proteins with variational quantum algorithms**
Hanna Linn, Lucas Knuthson, Anders Irbäck, Sandipan Mohanty, Laura García-Álvarez, and Göran Johansson
In review, available on arXiv

C **Efficient Quantum Protein Structure Prediction with Problem-Agnostic Ansatzes**
Hanna Linn, Rui-Hao Li, Alexander Holden, Abdullah Ash Saki, Frank DiFilippo, Tomas Radivoyevitch, Daniel Blankenberg, Laura García-Álvarez, and Göran Johansson
In review, available on arXiv

D **Quantum Algorithm for Protein Structure Prediction Using the Face-Centered Cubic Lattice**
Rui-Hao Li, Hakan Doga, Bryan Raubenolt, Sarah Mostame, Nicholas DiSanto, Fabio Cumbo, Jayadev Joshi, Hanna Linn, Maeve Gaffney, Alexander Holden, Vinooth Kulkarni, Vipin Chaudhary, Kenneth M. Merz Jr, Abdullah Ash Saki, Tomas Radivoyevitch, Frank DiFilippo, Jun Qin, Omar Shehab, and Daniel Blankenberg
In review, available on arXiv

E **Detecting quantum speedup of random walks with machine learning**
Hanna Linn, Yu Zheng, and Anton Frisk Kockum
In revirew, available on arXiv

Contents

1 Introduction

Quantum computing as a concept emerged in the late 20th century as researchers began exploring how quantum mechanics could revolutionize computation. In 1980, Paul Benioff proposed a quantum mechanical model of a Turing machine, demonstrating that computation could be described within the framework of quantum theory [1]. This foundational work established the possibility of quantum computers as physical systems governed by quantum dynamics. Building on this, Richard Feynman argued that quantum computers would be uniquely suited to simulate quantum systems, which are notoriously difficult for classical computers to model efficiently [2]. Around the same time, Yuri Manin suggested that quantum computers could outperform classical ones in certain computationally intensive tasks [3, 4]. In 1985, David Deutsch formalized the notion of a universal quantum computer, showing that such a device could simulate any physical system governed by quantum mechanics [5].

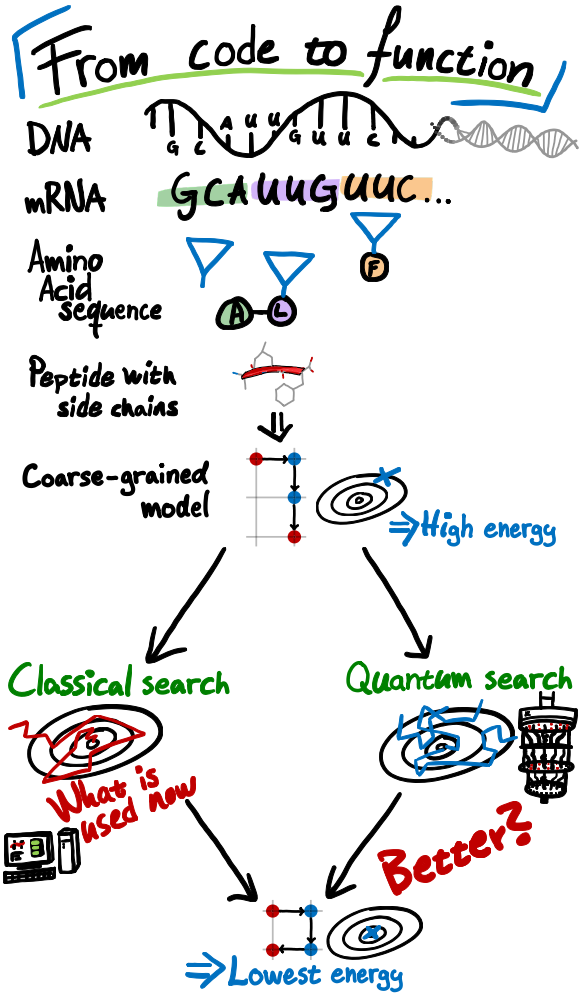

Figure 1.1: From DNA to mRNA to amino acid sequence to functioning protein. But how do we most efficiently and accurately predict what the protein will look like?

In 1994, Peter Shor introduced a groundbreaking quantum algorithm that, when run on the right quantum device, would be capable of factoring large integers exponentially faster than the best-known classical algorithms [6], demonstrating that quantum computers could, in theory, solve problems of significant practical and cryptographic importance. Shortly after, Lov Grover developed a quantum search algorithm that could theoretically provide a quadratic speedup for unstructured search problems [7], showing that even tasks with minimal structure could benefit from quantum acceleration. These results marked a turning point, establishing that sufficiently large quantum computers could theoretically outperform classical ones in solving real-world problems. Quantum computers of this scale and capability have yet to be built.

Even so, quantum computing has seen remarkable progress in recent years, with advances in hardware, algorithms, and theoretical understanding [8–11]. The field of quantum algorithms can be broadly divided into two domains today. First, those like Shor's and Grover's algorithm, which are theoretically proven to outperform their classical counterparts running on quantum devices we anticipate in the future. Second, the algorithms that are tailored for the quantum computers we have today. Current quantum devices remain relatively small and unstable compared to the demands of those early quantum algorithms. Their limited scale and susceptibility to noise restrict the sizes of problem instances, the duration of computations, and the reliability of the results. In the era of the so-called noisy intermediate-scale quantum (NISQ) device [12], hybrid quantum-classical algorithms have gained prominence. These algorithms utilize a classical computer in conjunction with a quantum circuit, enabling us to leverage quantum properties while mitigating hardware limitations. Applications span diverse domains, including machine learning [13], optimization [14], and more, see Fig. 1.2.

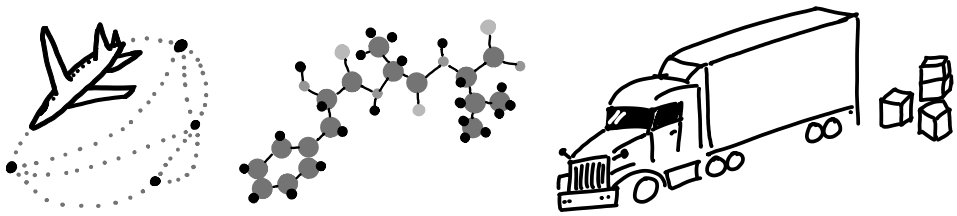

Figure 1.2: **Examples of application domains explored in quantum computing research.** These include flight scheduling in logistics [15], molecular simulation in chemistry [16], and vehicle routing in transportation [17].

Trying to find yet more applications for quantum algorithms, this thesis began at the intersection of two dynamic fields: How quantum algorithms could be used to improve any computation classically challenging in any problem within the *life sciences*. The life sciences encompass a wide array of disciplines that study living organisms, including biology, genomics, proteomics, biotechnology, and more. The potential for finding an application within these fields seems promising [18, 19].

Our early investigations focused on *DNA sequencing*, a cornerstone of ge-

nomics. However, after a year of exploration, we concluded that DNA sequencing may not be the most suitable application for current quantum computing technologies. The limitations of today's quantum devices make them better suited for problems with small input domains and large search spaces. In contrast, a single DNA sample contains millions of nucleotides but only four distinct types, resulting in a vast input space and relatively straightforward computational operations. We deemed it a poor candidate for quantum acceleration under current hardware constraints. Despite our inability to identify a compelling quantum advantage in this domain, the intersection of quantum computing and genomics remains an open and evolving frontier [20–23].

The realization of the difficulties in applications in DNA sequencing led to a pivot toward computational problems involving proteins, which not only align better with the characteristics of today's quantum computers but were, and remain, a growing and impactful research area within quantum algorithms [24–41]. The essence of the question is visualized in Fig. 1.1. Proteins are made up of sequences of amino acids, and their biological roles are determined by the way these chains fold into complex three-dimensional (3D) structures. Protein structures govern how molecules interact, how cellular pathways operate, and how biological systems behave. Understanding these structures is essential not only for decoding life at the molecular level but also for enabling structure-based drug design, which relies on accurate models of protein targets to develop effective therapeutics [42, 43]. The *protein structure prediction (PSP)* problem, inferring a protein's 3D conformation from its amino acid sequence, has long stood as a central issue in computational biology, due to the intricate link between molecular structure and biological function. The inverse problem of determining amino acid sequences that would fold into a target structure is called *protein design*. Proteins typically consist of a few hundred amino acids, but the search space for possible foldings is astronomically ample due to the many degrees of freedom [44]. This combination of a compact input space and a vast search space makes PSP and protein design ideal candidates for quantum computing.

While the theoretical promise of quantum computing is substantial, a central question remains: what can be realistically achieved with current quantum devices in the domain of PSP and protein design? As previously discussed, today's quantum hardware is currently constrained by limited scale, noise, and short run time, and most existing implementations have primarily served as proof-of-concept demonstrations rather than practical solutions.

In this thesis, we mainly examine the potential and limitations of modern quantum devices through four interconnected avenues. First, discuss what has been done in the field, give a detailed overview of the methods, i.e., quantum algorithms, and the problem at hand, i.e., PSP and protein design. Second, we investigate the scale of future quantum computers required to challenge classical methods in protein-related tasks. Third, we explore the problem of protein design, which may be more tractable than PSP and thus more amenable to current quantum capabilities. Finally, we assess how far existing devices can be pushed and try to reach outside proof-of-concept in PSP with NISQ algorithms.

These efforts naturally raise two key challenges. One concerns the encoding of protein information from atomic-level data, translated into bitstring representations suitable for input into a quantum computer. This is discussed in Chapter 2, which provides an in-depth look at proteins, a brief description of how to simulate protein dynamics, and how to predict the 3D structure or design proteins. The second key challenge involves designing hybrid quantum-classical algorithms, as explained in Chapter 3, together with a ground-up description of quantum computing, where careful decisions must be made about which parts of the computation are best handled by quantum resources and which should remain in the classical domain. Furthermore, Chapter 4 brings it all together, discussing the current state of the field and describing the paper contributions of this thesis. Concluding the thesis, Chapter 5 provides a summary and outlines potential future research directions based on our findings.

2 Computational problems involving proteins

Proteins are fundamental to biological function, and understanding their behavior is central to many scientific and medical challenges. This chapter explores computational approaches to protein-related problems, beginning with an introduction to protein structure and function.
We then examine coarse-grained models, as shown in Fig. 2.1, for simulating protein folding and predicting the protein's 3D structure. The chapter also covers protein design, engineering novel proteins with desired properties. Lastly, the chapter reviews classical algorithms used to tackle these tasks.

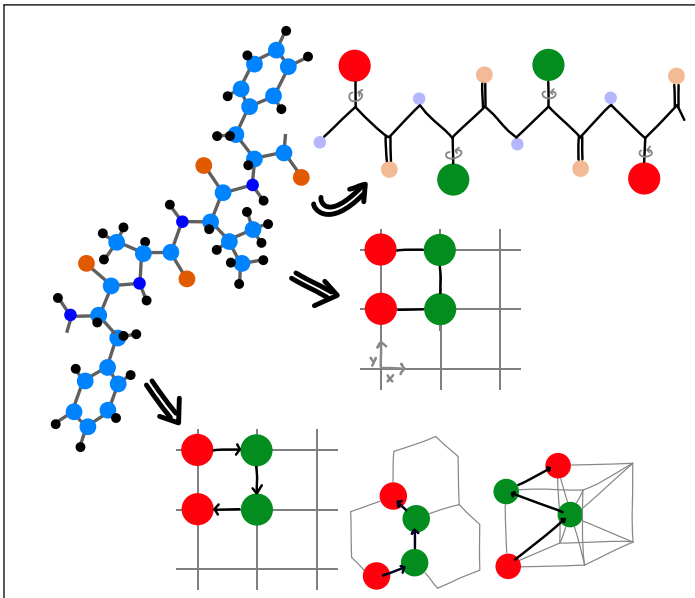

Figure 2.1: Fully atomistic depiction of a four amino acid chain and the corresponding **coarse-grain model representations** for the side-chain conformation-based model (top), coordinate-based lattice model (middle), and turn-based lattice model (bottom) with the square, tetrahedral, and body-centered cubic (BCC) lattice (from left to right).

2.1 Protein structure and function

Proteins are fundamental to all biological processes, including, among many, mobilizing the intracellular response, controlling cell dynamics, and cell shape. These macromolecules exhibit significant diversity in form and function, which

is determined by their three-dimensional structure.

Proteins are composed of one or more long chains of *amino acid residues*. Although over 500 amino acids exist in nature, only 22 α-amino acids are encoded by the genetic code and used in protein synthesis within living cells. These are referred to as canonical or natural amino acids [45, 46]. Each protein chain consists of one or more polypeptides, typically comprising at least 50-100 amino acids per chain. Human proteins have a median length of 375 amino acids [47], and therapeutic targets average around 414 amino acids [48]. The organism's DNA determines the sequence of the amino acids in a protein (see Fig. 1.1), and the amino acids are linked via peptide bonds between their amino (NH_2) and carboxyl (COOH) groups. Once incorporated into a chain, each amino acid is referred to as a residue, and the repeating backbone of two carbon and a nitrogen forms the *protein backbone*. Each amino acid possesses a unique side chain attached to one of the two backbone carbons, which contributes to the protein's chemical properties and interactions. These side chains can be classified based on their polarity and charge, and are often abbreviated by a three-letter combination or just one letter. For example, Alanine in Fig. 1.1 is a non-polar amino acid with a methyl group as its side chain and is abbreviated as Ala or A. In the same figure, we also see Leucine (L) and Phenylalanine (F).

Broadly, proteins are classified into three categories: fibrous, membrane, and globular proteins. Fibrous proteins, including those found in bone and muscle, are elongated and generally insoluble. Membrane proteins, such as receptors and ion channels, are embedded within cellular membranes and play critical roles in signal transduction and transport. Globular proteins, such as enzymes and signaling molecules, fold to bury hydrophobic residues within, forming a hydrophobic core, and expose hydrophilic residues to water, forming a spherical shape. These are discussed in Paper A and are the most studied in this thesis.

2.2 Protein folding

The functional form of a protein arises through a process known as *folding*, whereby the linear *amino acid chain* adopts a specific 3D structure [49]. Most proteins typically fold reliably into a single structure called the *native state*. This folding is a spontaneous process within the crowded cell, determined by the amino acids interacting with their surroundings, where factors such as solvent (water or lipid bilayer), temperature, pH, salt concentration, and other molecules present play a significant role. The folding in the cell is sometimes facilitated by molecular chaperones, which assist in folding and prevent aggregation (misfolding).

Levinthal's paradox highlights the complexity of this process, formulated by Cyrus Levinthal [50] in 1968. The paradox states that the number of possible conformations (ways that the protein can fold), caused by the large number of degrees of freedom, for an unfolded polypeptide chain is so astronomically large that an exhaustive search for the native state would take longer than

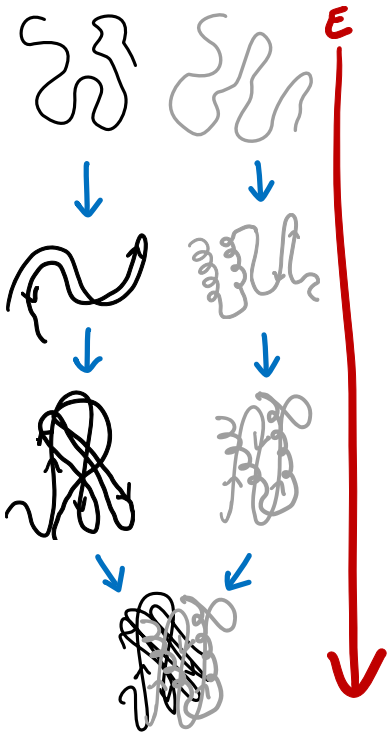

Figure 2.2: **Protein folding** begins with a high-energy, unfolded polypeptide chain. As folding progresses through the four levels, entropy decreases and the system transitions toward lower-energy, more stable configurations. Amino acid chains will fold into, often unique, native structures.

the age of the universe. Noting that proteins fold much faster than a random search would allow, Levinthal suggested that folding must proceed through a sequence of semi-stable intermediate structures rather than by exploring all possible conformations. As the protein folds, entropy decreases and the system moves toward lower-energy configurations that are more compact, as shown in Fig. 2.2. Although the energy landscape is rugged with many local minima corresponding to partially folded intermediates, the funnel model [51] assumes a bias toward the native state, enabling efficient folding despite the complexity.

The protein folding process is organized hierarchically into four distinct levels, see Fig. 2.2, each contributing to the molecule's final shape and function [49].

- The *primary structure* refers to the linear sequence of amino acids, which is determined by the process of mRNA translation. This sequence dictates how the protein will fold and ultimately function.
- The *secondary structure* arises from local interactions within the backbone, forming recurring motifs such as α-helices and β-sheets through hydrogen bonding. These elements serve as the building blocks for more complex configurations.
- The *tertiary structure* describes the overall three-dimensional conformation of a single polypeptide chain, incorporating its secondary structural

elements and functional domains into a cohesive unit. The term is often used synonymously with the term fold.

- The *quaternary structure* emerges when multiple polypeptide chains assemble into a larger, functional protein complex, often enabling cooperative interactions and enhanced biological activity.

Correct folding is essential for protein function. Even a single amino acid substitution can disrupt the folding process, leading to loss of function or disease. A well-known example is the inherited disorder sickle cell anemia, which is caused by a single amino acid change out of a total of 574 amino acids in hemoglobin, leading to rigid, sickle-shaped red blood cells that can obstruct blood flow [52]. Other examples of diseases linked to improperly folded proteins are neurological diseases like Alzheimer's and Parkinson's. Usually, spontaneously misfolded proteins are typically recognized and degraded by cellular quality control mechanisms.

With this in mind, studying protein folding becomes essential for understanding disease mechanisms and advancing medical research, as it provides a valuable approach to enhancing our understanding of protein interactions, functions, and structures. However, simulating the protein folding process is a formidable computational challenge. The rugged energy landscape requires a highly detailed representation of amino acid interactions in their natural environment [53, 54].

2.2.1 In silico protein structure prediction

Studying the native state of a protein can be considered a gateway problem for understanding protein folding. One way to study proteins is through the use of experimental techniques. Two often mentioned experimental techniques are *X-ray crystallography*, which determines atomic-level structures by analyzing diffraction patterns from protein crystals, and *Nuclear magnetic resonance (NMR) spectroscopy*, which uses radiofrequency radiation to probe the magnetic properties of atomic nuclei, revealing details about their electronic surroundings, thereby offering insights into molecular bonding [55]. More recently, cryo-electron microscopy (cryo-EM) has emerged as a powerful method, enabling the visualization of biomolecules in near-native states by flash-freezing samples and imaging them with an electron beam, thus providing high-resolution structural information without the need for crystallization. The experimental data of numerous proteins is publicly accessible in the *Protein Data Bank (PDB)* [56], a repository that archives experimentally determined 3D structures of proteins, nucleic acids, and complex biomolecular assemblies. These techniques are generally applied to proteins in crystalline or ensemble states, providing only static snapshots rather than capturing the full dynamics of the protein in its native environment. As a result, they offer limited insight into how proteins interact with their surroundings. NMR spectroscopy has the advantage of operating under conditions closer to physiological temperatures, but it faces significant

challenges when applied to large proteins, typically those exceeding 270–360 amino acids. Moreover, all these experimental methods are resource-intensive and costly, both in terms of time and equipment.

One computational approach to this problem is protein structure prediction (PSP), which aims to infer a protein's secondary and tertiary structure from its primary amino acid sequence. *In silico* PSP algorithms typically consist of two tightly integrated components: the conformational search strategy and the energy function. The conformational search explores the vast space of possible protein structures, seeking the global energy minimum. Historically, methods such as simulated annealing and molecular dynamics have been widely used for this purpose [54, 57]. The energy function takes a protein conformation as input and returns its associated energy, and the conformation corresponding to the native state should be the minimum of the energy function. Energy functions can be broadly categorized into two types: *physics-based*, which rely on fundamental physical principles, and *knowledge-based*, which incorporate statistical and empirical data [58]. Hybrid approaches that combine both types of terms also exist.

In strictly physics-based approaches, atomic interactions are modeled using quantum mechanics and fundamental physical constants such as the electron charge. Each atom is characterized by its type and electronic configuration, enabling highly detailed and accurate simulations. However, performing such all-atom quantum calculations is computationally prohibitive, making it necessary to combine them with classical mechanics for practical applications. A well-known example of an all-atom physics-based model is *CHARMM*, introduced by Brooks *et al.* [59]. However, these models are computationally intensive and quickly become impractical for large or complex systems, necessitating simplifications. A practical starting point is to treat atoms as point particles interacting through a defined potential form, or to group a few atoms in a process called *coarse-graining*. An example of a coarse-grained physics-based model is *UNRES* by Liwo *et al.* [60]. Coarse-grained models can also be constructed using statistical data.

Furthermore, the knowledge-based approach relies on statistical models, evolutionary information, and experimentally determined structures, often sourced from the PDB, to infer likely conformations. Rather than simulating the folding process, they predict static structures based on patterns observed in known proteins. While experimental techniques offer high-resolution data, they are both costly and time-consuming. Computational alternatives are therefore especially valuable for studying mutations, such as single amino acid substitutions, without the need for protein synthesis or extensive laboratory procedures.

More recently, knowledge-based methods have achieved a high rate of success, with deep learning algorithms revolutionizing the field of PSP [61]. A landmark development was the release of *AlphaFold2* [62], a transformer architecture capable of predicting protein structures with near-experimental accuracy. AlphaFold2 employs a deep neural network trained on the PDB, utilizing multiple sequence alignments (aligning biological sequences to identify regions of similarity that may indicate functional, structural, or evolutionary relation-

ships) to predict torsion angles and inter-residue distances. The model generates distance maps and refines them via gradient descent, achieving remarkable performance in the Critical Assessment of Structure Prediction (CASP) competition. The CASP is a biennial global competition that evaluates the accuracy of computational methods for PSP and is widely regarded as the "world championship" of PSP, providing a rigorous, double-blind benchmark for the field [63]. Participants are tasked with predicting the 3D structures of proteins whose experimental structures have been solved but not yet made publicly available. Before AlphaFold2, the *Rosetta software suite* had long been a leading tool in CASP, consistently ranking among the top-performing methods [64–67]. Rosetta's fragment-based assembly, all-atomistic fine-tuning, and coarse-grained sampling strategies combine physics-based and knowledge-based methods to lay the groundwork for many of the innovations seen in modern structure prediction pipelines. Both the success of AlphaFold2 and the Rosetta suite were recognized with the 2024 Nobel Prize in Chemistry [68].

However, knowledge-based methods are limited by their reliance on experimentally resolved structures, which are produced much more slowly than new amino acid sequences are identified through genomic sequencing. This data imbalance restricts their scalability and generalizability. Because these models are trained on static structural snapshots, they may also struggle with tasks that fall outside the distribution of their training data. For example, AlphaFold models have been shown to perform poorly on short peptide fragments and highly flexible local domains, where limited sequence context and sparse evolutionary information lead to structural ambiguity and reduced accuracy [69, 70].

In contrast, physics-based simulations offer an alternative that potentially enables a better description of proteins with large unstructured regions. Furthermore, knowledge-based models do not learn the folding process [71], which may be better described using physics-based models. However, the study of physics-based models is computationally challenging and would benefit from advances in conformational search algorithms, an area where quantum optimization techniques may offer promising solutions.

2.3 Coarse-grained models

Before the advent of *fully atomistic* simulations, where every atom and interaction is explicitly represented, including bond angles, torsion angles, and environmental effects such as solvent interactions, *coarse-grained models* played a crucial role in protein folding research and remain essential in multiscale modeling. While fully atomistic models offer high accuracy, they are computationally demanding due to the vast number of degrees of freedom. Using models that simplify the system by grouping atoms into larger units, groups of atoms are represented as single interaction centers or *beads*, which reduces computational complexity significantly by lowering the degrees of freedom [72]. Coarse-grained solutions often serve as initial guesses for more detailed simulations, signifi-

cantly reducing the overall runtime of folding algorithms and enabling the study of large-scale protein dynamics, folding mechanisms, and structure prediction, especially when fully atomistic simulations are infeasible. However, even coarse-grained models become computationally intensive with increasing chain length.

A basic coarse-grained representation treats amino acids as beads on a string, where each bead encodes specific properties of the corresponding residue. The string of beads is then placed on a lattice in the *lattice model* [73, 74]. This model enables efficient simulations, albeit at the expense of detailed information regarding residue orientation and side-chain interactions. One of the simplest examples is the *HP model* [75, 76], which classifies amino acids as either hydrophobic (H) or polar (P), capturing essential folding behavior through hydrophobic collapse. A more refined approach is the *Miyazawa-Jernigan (MJ) model* [77], which assigns interaction energies to all 20 canonical amino acids based on statistical potentials derived from known protein structures.

Other coarse-graining strategies increase resolution by using multiple beads per amino acid. For instance, the *CABS model*, which uses a high-resolution lattice with 800 possible orientations between virtual α-carbon atoms [78].

Some models don't use a lattice, so-called *off-lattice models*. For example, the *AWSEM model* [79], which employs three beads per residue, two for the backbone and one for the side chain, and places particular emphasis on solvent effects. Other notable off-lattice models (but can also be used with a lattice) include *MARTINI* [80], which is widely used for simulating biomolecular systems with a four-to-one mapping of atoms to beads, and *OPEP* [81], which combines coarse-grained potentials with implicit solvent models for folding and aggregation studies. These models vary in resolution and physical assumptions, offering tailored solutions for different scales and types of protein simulations [72].

In this thesis, we focus on two coarse-grained models that have previously been explored in the context of quantum computing: the lattice model with various lattices [24–32, 38, 40, 41, 82–84] and the papers appended in this thesis Paper A, Paper D and Paper C. As well as the off-lattice *side-chain based model* [35–37]. Paper A concerns both the lattice model and one simple off-lattice model. While both models differ in resolution and complexity, they are relatively simple and have been previously successfully translated into Hamiltonians (the term is explained in Chapter 3), making them suitable for quantum optimization algorithms. However, as demonstrated in Paper C, such a translation is not strictly necessary, potentially opening the door to exploring more complex coarse-grained models that were previously considered too complicated for direct quantum encoding.

2.3.1 Lattice model

The lattice model discretizes physical space into a grid, where each amino acid is represented as a bead s_i, which indicates the type of the ith bead, e.g., H or P in the HP model, occupying a lattice point. The accuracy of lattice-based predictions depends on the resolution and geometry of the lattice. Despite its

simplicity, the lattice protein model is *NP-complete* [85, 86], i.e., computationally challenging.

The protein chain is often modeled as a *self-avoiding walk*, where each bead is placed sequentially based on a set of discrete turns that determine the position of bead s_{i+1} relative to bead s_i. Overlaps are forbidden, reflecting the physical constraint that no two residues can occupy the same space.

The level of structural detail captured in lattice-based protein models depends on the specific lattice employed; higher-resolution lattices that allow greater degrees of freedom, often characterized by higher coordination numbers and angles mirrored in nature, tend to produce more accurate and realistic structural predictions [87, 88]. Here follows a selection of commonly used lattices in quantum algorithms for PSP that illustrate these differences in resolution and flexibility.

Square and Cubic lattices illustrated in Fig. 2.3, are widely used as a starting point in lattice-based modeling due to their structural simplicity. Both lattices are bipartite, allowing the constituent beads to be partitioned into two alternating subsets, denoted as the A and B sublattices in Fig. 2.3. The square lattice features a coordination number of four, meaning that each bead connects to four neighbors, while the cubic lattice exhibits a coordination number of six. The square and cubic lattice is discussed in Paper A and Paper B.

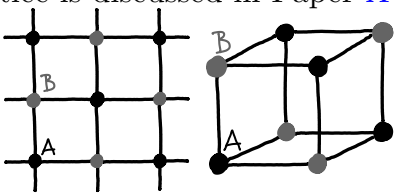

Figure 2.3: Unit cell representations of the **square lattice (left)** and **cubic lattice (right)**, in which each bead denotes an amino acid and each black edge indicates a permissible directional turn. The bipartite nature of these lattices is illustrated through the alternating A (black) and B (gray) sublattices.

Body-centered cubic lattice formed by adding a central point to each cube in a standard cubic lattice and illustrated in Fig. 2.4, is employed in Paper C. With a coordination number of eight, the BCC lattice provides greater flexibility than the simple cubic lattice. Its characteristic bond angles, $\{70.53°, 109.47°, 180°\}$, allow for a wider range of conformational possibilities.

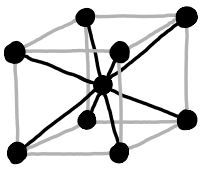

Figure 2.4: Unit cell representations of the **BCC lattice** in which each bead denotes an amino acid and each black edge indicates a permissible directional turn.

Face-centered cubic lattice introduced for PSP using quantum algorithms in Paper D and further employed in Paper C, features a coordination number of 12 and supports a diverse set of turn angles: $\{60°, 90°, 120°, 180°\}$. By incorporating additional points at the center of each face of the unit cell, the face-centered cubic (FCC) lattice achieves a higher packing density than the above-presented cubic lattices. This increased spatial resolution enables more realistic approximations of protein structures and better captures the steric constraints inherent in biological molecules. A notable advantage of the FCC lattice is its capacity to represent secondary structures with greater fidelity, particularly α-helices. In such helices, the protein backbone adopts a right-handed coil, with each residue contributing a 100° rotation along the helical axis, corresponding to approximately 3.6 residues per turn [89]. Furthermore, the FCC lattice has been shown to yield significantly lower energy configurations compared to other coarse-grained lattice models [90].

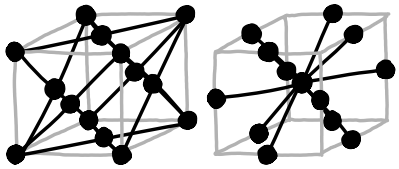

Figure 2.5: Unit cell representations of the **FCC lattice**, where each bead represents an amino acid and each black edge denotes an allowed directional turn. The left panel shows the conventional perspective, while the right panel emphasizes the coordination number more clearly.

Tetrahedral lattice also referred to as the diamond lattice and illustrated in Fig. 2.6, is a structurally simple yet chemically meaningful model with a coordination number of four.

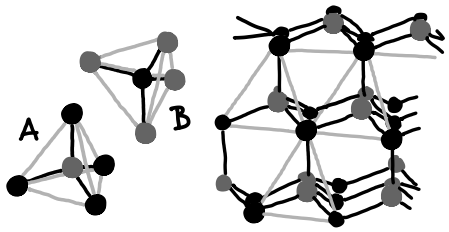

Figure 2.6: Unit cell representations of the **tetrahedral lattice**, where each bead represents an amino acid and each black edge indicates an allowed directional turn. The bipartite nature of the lattice is illustrated by alternating A (black) and B (gray) sublattices, which are shown separately on the left and combined in the full lattice view on the right.

Like the square and the cubic lattices, it is bipartite, allowing the beads to be partitioned into two alternating subsets. First introduced for PSP using quantum algorithms in Robert *et al.* [31], and further examined in Paper A and Paper C, the tetrahedral lattice is notable for its chemically plausible angular constraints. Specifically, it enforces a fixed bond angle of 109.47° between any two consecutively connected beads, reflecting the geometry of carbon atoms in the backbone. Dihedral angles are restricted to either 60° or 180°, contributing

to a more realistic depiction of protein structure. Despite these strengths, the lattice does not accommodate all naturally occurring bond geometries, limiting its ability to capture certain structural motifs fully. The tetrahedral lattice could also be used for modeling backbone and side-chain interactions.

2.3.2 Off-lattice model

Among the various off-lattice models, this thesis focuses on those with a fixed backbone and with *side-chain optimization* as these have been previously explored in the context of quantum algorithms. The positioning of side chains plays a critical role in protein folding [91], and is often referred to as the *packing problem*. This problem is typically addressed in an iterative framework, where backbone conformations are optimized in one step, followed by side-chain adjustments, and then further refinement, with the process iteratively alternating between backbone optimization and side-chain adjustments.

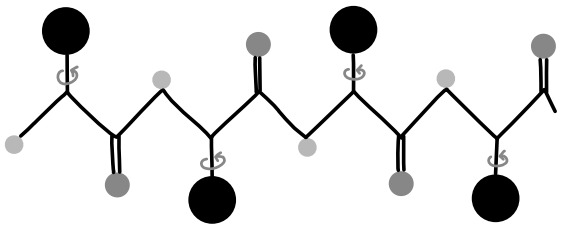

Figure 2.7: A sketch of the **rotamer model**. The backbone remains fixed, and the residues are grouped into beads (black) that can rotate.

Side chains vary in complexity depending on their length and chemical structure. Longer side chains possess more degrees of freedom, while shorter ones have fewer components and thereby fewer angles to vary. These rotation angles can be modeled either as continuous variables or discretized into sets of torsion angles. The entire side chain can also be represented by a single bead, like in Fig. 2.7. In discrete models, side-chain conformations can be represented by predefined angle sets known as *rotamers*, rotational conformations, typically derived from empirical frequency distributions such as those compiled by Dunbrack and Karplus [92]. We can also have other side-chain conformations. The combinatorial nature of the problem renders side-chain conformation selection NP-complete, and the associated energy minimization task is NP-hard [93].

Rosetta software suite uses one of the most widely used implementations of the rotamer model, as mentioned above [64–67, 94, 95]. Rosetta employs both coarse-grained and all-atom representations of proteins, transitioning iteratively between low- and high-resolution models. The coarse-grained model encompasses all heavy backbone atoms and virtual atoms representing side chains. Rosetta is a comprehensive platform for macromolecular modeling and is widely used in tasks such as PSP, protein–protein docking, protein–ligand docking, antibody modeling, refinement of experimental structures, and protein design, among others.

2.4 Protein design

A significant biophysical challenge in molecular biology is the inverse folding problem, commonly referred to as protein design [96–98]. Unlike PSP, which seeks to determine the 3D structure from a given amino acid sequence, protein design aims to identify sequences that will reliably fold into a predefined target structure. This task has gained significant momentum in recent years, driven in part by advances in machine learning and computational modeling [57, 99, 100].

The field has matured to the point where some of its most promising applications lie in biomedicine, including the design of vaccines and protein-based inhibitors [99], as well as in synthetic biology, where engineered proteins are used to create novel biological functions [101]. A landmark achievement in the field was the design, synthesis, and crystallographic validation of Top7, a 93-residue protein with no detectable sequence similarity to any known natural protein [96]. This demonstrated that it is possible to design entirely novel proteins that fold into stable, functional structures.

These achievements were achieved using coarse-grained off-lattice models, such as the rotamer model discussed in Sec. 2.3.2. In this context, rotamer sets can be extended to include mutational flexibility, allowing the exploration of alternative sequences that may better stabilize a target structure. On-lattice approaches have also been used to study simplified design dynamics [102, 103]. Despite these advances, the computational modeling of the biophysical principles underlying protein design remains a formidable challenge.

2.5 Protein structure prediction as an optimization problem

The PSP problem can be expressed as a *combinatorial optimization problem* when the conformational space is discretized. Let us define the protein conformation as a vector $\vec{x}$, representing a 3D structure of an amino acid sequence $\vec{s}$ of length N, the number of amino acids. Each amino acid s_i can adopt a set of possible conformations $\vec{R}_i$, which can be indexed and, thereby, represented as integers. This allows us to encode the entire protein structure as a sequence of integers representing discrete choices. Given a choice $r_i \in \vec{R}_i$ for each amino acid, we create $\vec{x} = (r_0, \ldots, r_{N-1})$.

In Paper A, we introduce the *cardinality vector*, $\mathbf{C} = (n(\vec{R}_0), \ldots, n(\vec{R}_{N-1}))$, a vector with the cardinalities of the sets of conformations. This abstraction simplifies the representation and manipulation of protein configurations, particularly when preparing the problem for encoding onto a quantum circuit.

For the lattice model, the conformations are the positions on the grid, e.g., first, second, third, seen from the upper left corner, and the cardinality vector represents the number of positions on the grid for each bead. If the lattice is bipartite, we can split the amino acid sequence and let half of it populate

each sublattice, as mentioned above, which allows us to divide each element in the cardinality vector by half. This can be described as the *coordinate-based* encoding. Another way to represent a structure on the lattice is through a turn sequence, a more compact encoding. This method, known as the *turn-based* encoding, assigns an integer to each possible directional turn: for example, turning up might be represented by one, left by two, down by three, and so on.

For the side-chain conformation model, the cardinality vector is simply the number of conformations for each side chain in order, given by a statistical distribution of the most probable side-chain conformations. An illustrative example of a tripeptide is provided in Fig. 2.8.

An energy function $E(\vec{x})$ evaluates the stability of a given conformation by assigning it an energy value. This function can be constructed using either physics-based interaction potentials or knowledge-based scoring functions, as discussed in previous sections. The energy function will contain the interaction energies between amino acids, such as nearest-neighbor or higher-order connections, and constraints to ensure that the conformation is feasible. For example, we do not want two amino acids to occupy the same position, causing overlap, and we aim to have only one conformation per amino acid. The constraints can, e.g., be softly enforced by a penalty term, as in the example in Fig. 2.8; other ways of implementing a constraint are discussed in Chapter 4. In the coordinate-based model, we obtain another condition of chain continuity, i.e., bead s_{i+1} follows bead s_i. The conformation with the lowest energy is referred to as the *native state*, denoted $\vec{x}_{\text{native}}$, with its corresponding energy $E(\vec{x}_{\text{native}}) = E_{\text{gs}}$ with *gs* for the *ground state*. There can be more than one ground state.

A simple example of an energy function is one where we can divide the interactions into one-body energies, corresponding to the choice of a specific conformation, and two-body energies, which represent the interaction energies between the chosen conformations. This can be the formulation for the side-chain conformation-based model

$$E(\vec{x}) = \sum_{i=0}^{N-1} O(\vec{x}_i) + \sum_{i=0}^{N-1} \sum_{j<i}^{N-2} T(\vec{x}_i, \vec{x}_j), \tag{2.1}$$

where $O(\vec{x}_i)$ depends in the side-chain conformation i and contains the one-body energy for that conformation and $T(\vec{x}_i, \vec{x}_j)$ depends on the pair of side-chain conformations i, j and contains the two-body energy between the pair. The summation over $i < j$ ensures that each pair is only counted once.

The goal of *in silico* PSP is to identify $\vec{x}_{\text{native}}$ from a given amino acid sequence $\vec{s} = (s_0, \ldots, s_{N-1})$. This task can be formally stated as an optimization problem:

$$\begin{aligned} &\text{minimize} && E(\vec{x}) \\ &\text{subject to} && \vec{x} \in F_{\text{structure}}, \end{aligned} \tag{2.2}$$

where $F_{\text{structure}}$ denotes the set of all valid conformations that satisfy the constraints of the protein model, e.g., chain continuity, non-overlapping residues, and complete assignment of atomic positions.

This formulation aligns PSP with well-known combinatorial problems such as `MAX-CUT`, `SAT`, `EXACT-COVER`, and `KNAPSACK` [104], many of which are known to be NP-complete. The inherent complexity of PSP makes it a promising candidate for quantum optimization techniques, which may offer advantages in exploring large and rugged energy landscapes more efficiently than classical methods.

2.6 Protein design as an optimization problem

Protein design can be formulated as the *inverse* of the PSP problem. Instead of identifying the most stable protein conformation for a given amino acid sequence, the goal is to find one or more sequences $\vec{s} = (s_0, \ldots, s_{N-1})$ that fold into a predefined target structure $\vec{x}$.

Each amino acid s_i in the sequence is selected from a set of possible residues, which may include all 20 canonical amino acids or a restricted subset depending on the design constraints. A candidate sequence $\vec{s}$ is thus a discrete assignment of residues to each position in the structure.

Given a fixed target conformation $\vec{x}$, the energy function $E(\vec{s}, \vec{x})$ evaluates the compatibility of a sequence with the structure, typically based on interaction potentials, packing efficiency, and other biophysical criteria. The objective is to identify sequences that minimize this energy function while satisfying biochemical and structural constraints.

Formally, the protein design problem can be stated as:

$$\begin{aligned} &\text{minimize} && E(\vec{s}, \vec{x}) \\ &\text{subject to} && \vec{s} \in F_{\text{sequence}} \end{aligned} \tag{2.3}$$

Here, F_{sequence} denotes the set of all valid amino acid sequences that satisfy design constraints such as residue compatibility, structural stability, and functional requirements. It may also be necessary to restrict the amino acid composition to prevent trivial homopolymer sequences. Furthermore, achieving minimal energy in the target structure does not guarantee that an alternative structure with even lower energy does not exist. Therefore, it is essential to verify whether the optimized sequence actually folds into the intended structure. This inverse formulation is computationally challenging due to the vastness of the sequence space and the complexity of accurately evaluating sequence-structure compatibility.

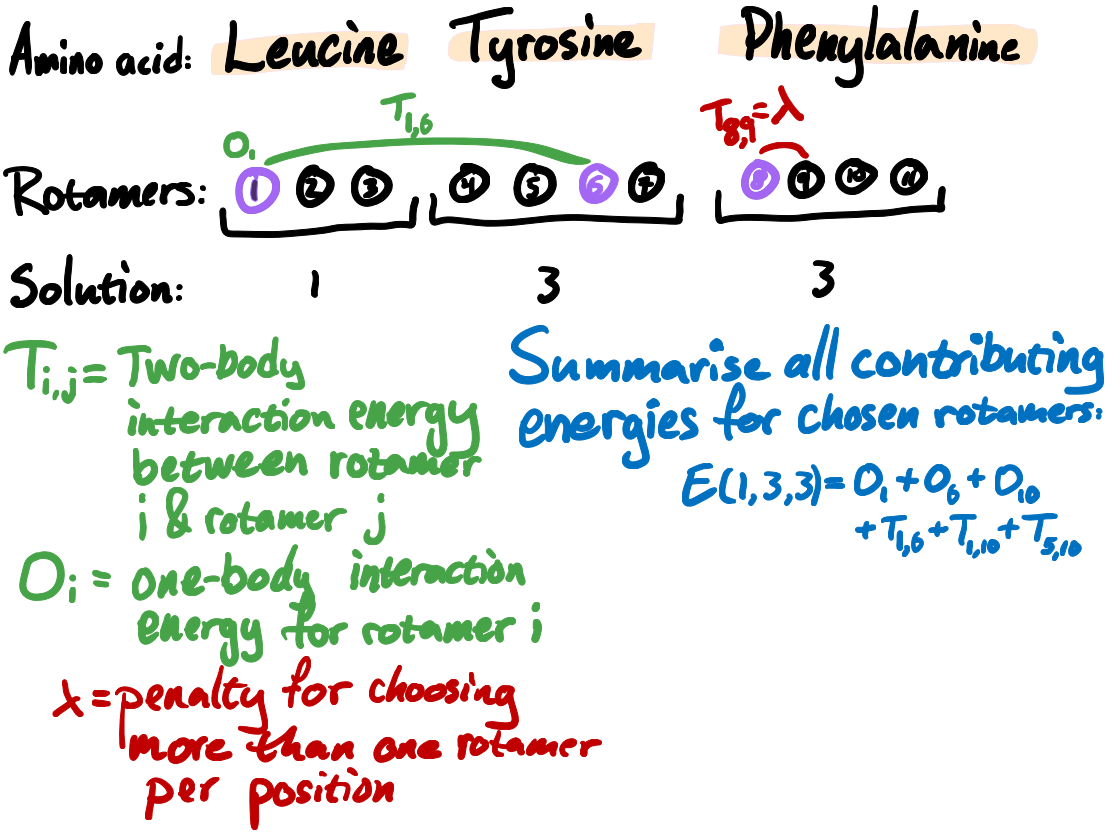

Figure 2.8: **An example of a structure in the rotamer model** with corresponding energy function. The solution (purple) is $\vec{x} = (1, 3, 3)$, i.e., the first rotamer for the first side chain which is the first rotamer, the third rotamer for the second side chain which is the sixth rotamer, and the third rotamer for the last side chain which is the eighth rotamer—which results in an energy (blue) that consists of the two-body energies of the three rotamers pairwise and the three one-body energies added up, from Eq. 2.1. If two rotamers for the same amino acid are chosen, e.g., rotamers eight and nine that both correspond to the third side chain, in the solution, then the solution energies are set to a high value in order to avoid unfeasible solutions (red).

3 Quantum computing

In this chapter, we will introduce quantum states that provide the foundations for qubits, which are manipulated using quantum gates. This is followed by quantum algorithms, with extra focus on variational quantum algorithms.

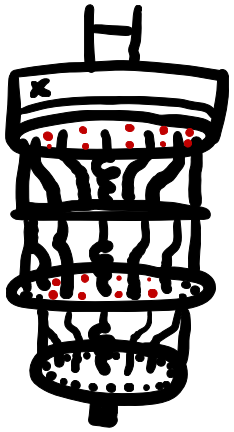

Figure 3.1: The chandelier-like **cryostat** cools quantum processors, like superconducting qubits. Although not the quantum computer itself, the cryostat is often used as a symbol for quantum hardware and will be used in this context throughout this thesis.

3.1 Quantum basics

Quantum computing is just linear algebra in disguise, albeit expressed through a formalism unique to quantum mechanics. This mathematical framework enables the representation and manipulation of quantum information, which is essential for the development and analysis of quantum algorithms.

3.1.1 Quantum state

To describe quantum states, *Dirac notation* is usually used, also known as *bra-ket notation*. This notation provides a concise and expressive way to represent vectors and inner products in complex Hilbert spaces, $\mathcal{H}$ [105, 106].

- A *ket* vector represents a quantum state, denoted as $|\psi\rangle$, which corresponds to a column vector in a complex vector space.
- The *bra* vector, written as $\langle\psi|$, is the Hermitian conjugate (complex conjugate transpose) of the ket vector and corresponds to a row vector.
- The term "bra-ket" originates from the inner product of two quantum states, expressed as $\langle\psi|\phi\rangle$, which yields a complex scalar and encodes the overlap or probability amplitude between the states $|\psi\rangle$ and $|\phi\rangle$.

3.1.2 Qubits

A quantum state of a two-level system is called a *qubit* (quantum bit), analogous to the classical bit in conventional computing [106, 107], see Fig. 3.2. A classical bit can exist in one of two discrete states: `1` or `0`, physically represented by high or low voltage in a wire. A qubit can exist in a continuous superposition of both states and is physically manifested through a quantum system. There is more than one way of implementing a qubit, including ion traps [108], superconducting circuits [109], and photonic systems [110].

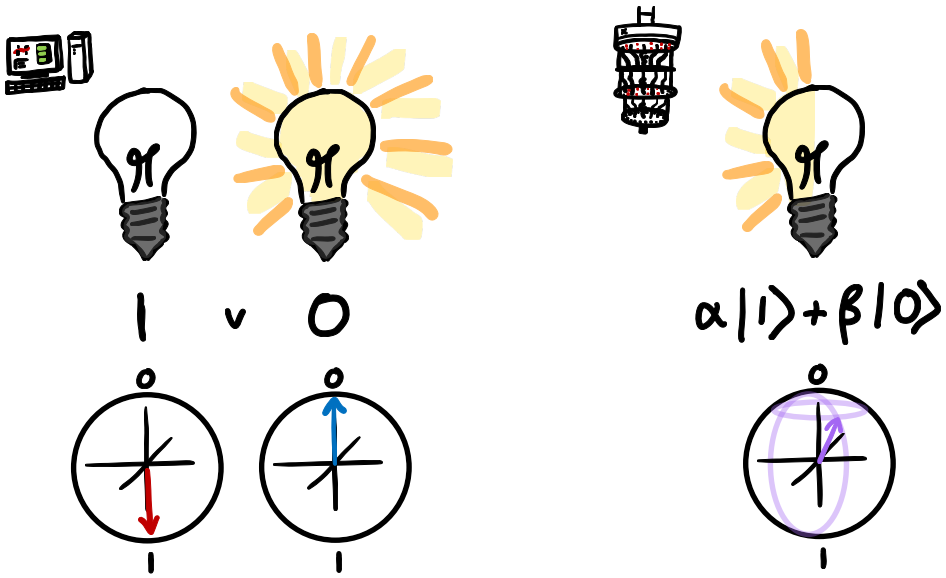

Figure 3.2: *Left:* The **classical bit** can either be on, 1, or off, 0. The classical computer is represented throughout this thesis by the screen, keyboard, and box seen in the figure.
Right: The quantum bit, **qubit**, can be in a superposition of on, 1, and off, 0.
Bottom: The **Bloch sphere**, a way to visualize the states of the bits. A point on the surface of the sphere represents a quantum state. Operations on the qubit correspond to rotations of the point on the sphere. The north and south poles correspond to the basis states $|0\rangle$ and $|1\rangle$, respectively, while all other points represent superpositions of $|1\rangle$ and $|0\rangle$.

The Hilbert space of a qubit, q, is

$$\mathcal{H}_{\text{qubit}} = \{|0\rangle, |1\rangle\} = \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\}. \tag{3.1}$$

Mathematically, the state of a qubit is represented as:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \tag{3.2}$$

where $\alpha, \beta \in \mathbb{C}$ are complex amplitudes satisfying the normalization condition

$$\langle\psi|\psi\rangle = |\alpha|^2 + |\beta|^2 = 1. \tag{3.3}$$

The squared magnitudes of these coefficients correspond to the probabilities of measuring the qubit in the respective basis states: $|\alpha|^2$ for $|0\rangle$ and $|\beta|^2$ for $|1\rangle$.

The *Bloch sphere* provides a geometric representation of a single qubit state. Any pure qubit state can be visualized as a point on the surface of a unit sphere in three-dimensional space. The north and south poles correspond to the basis states $|0\rangle$ and $|1\rangle$, respectively, while all other points represent superpositions of $|1\rangle$ and $|0\rangle$. Of particular interest are the *phase states*, located on the equator of the Bloch sphere:

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \quad \text{and} \quad |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}, \tag{3.4}$$

as these are unique to the qubit. The probability of the qubit measuring to 1 and 0 is 1/2 for both.

3.1.3 Multi-qubit systems

In systems with multiple qubits, the state space grows exponentially. An M-qubit system spans a 2^M-dimensional Hilbert space, allowing for superpositions over all possible classical bitstrings of length M. For example, a three-qubit system has the basis:

$$\mathcal{H}_{3-\text{qubit}} = \{|000\rangle, |001\rangle, \ldots, |111\rangle\}, \tag{3.5}$$

with the number of states in the Hilbert space being $|\mathcal{H}_{3-\text{qubit}}| = 2^3 = 8$. A general state of the system is written as:

$$|\psi\rangle = \sum_{i=0}^{2^M-1} c_i \, |\vec{x}_i\rangle \,, \tag{3.6}$$

where $c_i \in \mathbb{C}$ and $\sum_i |c_i|^2 = 1$, same as α and β above. The coefficients $|c_i|^2$ represent the probabilities of measuring the system in the corresponding basis state $|\vec{x}_i\rangle$.

Unlike classical registers, which can only occupy one configuration at a time, quantum registers encode information in the amplitudes of all basis states simultaneously. Storing these amplitudes classically would require 2^M complex numbers, highlighting the exponential advantage of quantum systems. Upon measurement, the superposition collapses to a single basis state. To extract all the probabilities $|c_i|^2$, repeated measurements of identically prepared systems are necessary.

3.1.4 Quantum gates

Quantum gates are used to manipulate the probability amplitudes c_i in the quantum state $|\psi\rangle$ defined in Eq. 3.6. To understand how quantum states are

manipulated, we begin with the time evolution of a quantum system, which is governed by the *time-dependent Schrödinger equation*:

$$i\hbar\frac{d}{dt}\left|\psi(t)\right\rangle = H\left|\psi(t)\right\rangle, \tag{3.7}$$

where H is the system's *Hamiltonian operator*, whose eigenvalues correspond to the energy levels of the system. In this thesis, we adopt natural units and set Planck's constant $\hbar = 1$ for simplicity. Solving the time-independent Schrödinger equation,

$$H\left|\vec{x}_i\right\rangle = E_i\left|\vec{x}_i\right\rangle, \tag{3.8}$$

yields a set of eigenstates $|\vec{x}_i\rangle$ and corresponding eigenvalues E_i, which are real since Hamiltonian operator is Hermitian. The lowest eigenvalue E_0 defines the ground state $|\vec{x}_0\rangle$ of the system. In the energy basis, the Hamiltonian is diagonal:

$$H = \sum_{i=0}^{2^M-1} E_i\left|\vec{x}_i\right\rangle\left\langle\vec{x}_i\right|. \tag{3.9}$$

If the Hamiltonian is time-independent, the solution to the Schrödinger equation can be expressed using the unitary time-evolution operator:

$$U(t) = e^{-iHt}, \quad \left|\psi(t)\right\rangle = U(t)\left|\psi(0)\right\rangle, \tag{3.10}$$

where e^A is defined via the matrix exponential and $|\psi(0)\rangle$ can be the initial state of the quantum register. This formalism describes quantum gates, which are unitary operators such as U, that preserve the norm of the quantum state vector and ensure that the total probability remains normalized when acting on quantum states, evolving them in time according to the system's dynamics.

To introduce the concept of quantum gates and aid in understanding, it is helpful to begin with their classical counterparts. In classical computing, logical operations are carried out using gates such as NOT, AND, and OR, which manipulate or operate on classical bits, as shown in Fig. 3.3. For example, the one-bit NOT-gate, $\neg$, inverts the input bit, transforming a one into a zero:

$$\neg 1 = 0. \tag{3.11}$$

The two-bit AND-gate, $\wedge$, on the other hand, outputs one only if both input bits are one:

$$1 \wedge 1 = 1, \quad \text{but} \quad 1 \wedge 0 = 0. \tag{3.12}$$

A quantum gate that acts on a qubit can be visualized by rotations on the Bloch sphere, see Fig. 3.2. Rotations around axis j by angle θ are given by:

$$R_i(\theta) = e^{-i\frac{\theta}{2}\sigma^j}, \tag{3.13}$$

where σ^j, with $j \in \{x, y, z\}$, is one of the commonly used Hamiltonians called *Pauli matrices*:

$$\sigma^x = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \sigma^y = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \text{and} \quad \sigma^z = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{3.14}$$

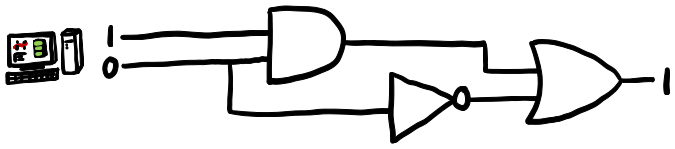

Figure 3.3: **Classical logic circuit composed of multiple gates.** The register starts in the state `10`, three gates act on the initial state in the order [AND, NOT, OR], and the final readout after manipulation from the gates is `1`.

For example, a rotation around the x-axis:

$$R_x(\theta) = \cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right) \sigma^x. \tag{3.15}$$

A rotation of $\theta = \pi$ around the x-axis yields the quantum NOT-gate (omitting the global phase $-i$):

$$R_x(\pi) = \sigma^x, \tag{3.16}$$

$$\sigma^x |1\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle . \tag{3.17}$$

The *Hadamard gate*,

$$\mathbb{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \tag{3.18}$$

transforms basis states into phase states, enabling superposition of qubits, here the probability of measuring $|0\rangle$ and $|1\rangle$ is both 50%, see Fig. 3.4. The Hadamard

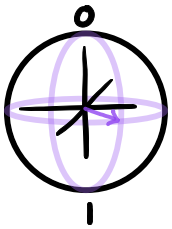

Figure 3.4: **Phase state** from Eq. 3.4 on the equator of the Bloch sphere, created with the Hadamard gate from Eq. 3.18. The probability of measuring either $|0\rangle$ or $|1\rangle$ is 50%.

gate acting on $|1\rangle$ gives the $|-\rangle$:

$$\mathbb{H} |1\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = |-\rangle , \tag{3.19}$$

and the $|+\rangle$ when acting on $|0\rangle$:

$$\mathbb{H} |0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = |+\rangle . \tag{3.20}$$

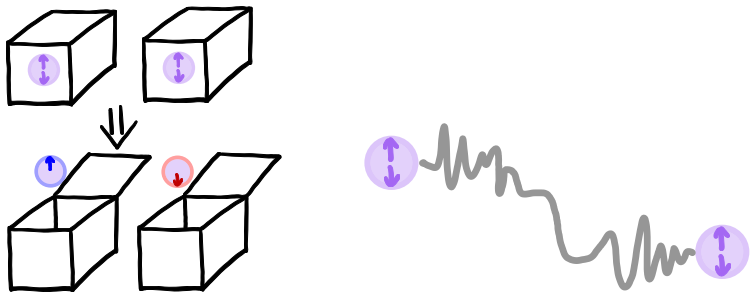

Figure 3.5: **Entangled qubits** in a superposition where measurement of one affects the state of the other (left), regardless of spatial separation (right).

3.1.5 Multi-qubit gates and entanglement

In quantum computing, entangling gates are multi-qubit operations that generate *quantum entanglement*, non-classical correlations between qubits that cannot be described by separable product states. These gates are essential for leveraging the full computational capacity of quantum systems, as operations on one qubit within an entangled register can influence the state of all entangled qubits. This non-local behavior enables certain computations to be performed more efficiently, often requiring fewer gate operations than their classical counterparts.

Formally, a gate is said to be entangling if it can transform a product state into an entangled state, one in which the individual qubits no longer possess independent descriptions. A measurement of one qubit instantaneously affects the state of the other, regardless of spatial separation, see Fig. 3.5. This non-classical dependency is not due to any signal transmission, but rather reflects the nature of quantum states.

For example, the *controlled-NOT (CNOT)* gate flips the target qubit only if the control qubit, the first qubit, is in state $|1\rangle$:

$$\mathrm{CNOT}\,|11\rangle = |10\rangle \quad \text{and} \quad \mathrm{CNOT}\,|01\rangle = |01\rangle\,. \tag{3.21}$$

Further, we consider the application of a Hadamard gate followed by a CNOT gate:

$$|00\rangle \xrightarrow{\mathbb{H}\otimes I} \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) \xrightarrow{\mathrm{CNOT}} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \tag{3.22}$$

The resulting state is one of the four Bell states, which are maximally entangled and form a complete basis for the two-qubit Hilbert space.

A *universal gate set* allows the construction of any quantum operation and must be able to approximate any unitary operation to arbitrary precision. In classical computing, the AND and NOT gates form a universal set of logic gates together. In quantum computing, a universal gate set can be combined to approximate any unitary operation on a multi-qubit quantum system. One example of a quantum universal gate set is two of the rotational gates, e.g., $R_x(\theta)$ and $R_y(\theta)$, but it can be any combination, together with the CNOT gate. The

gate set of a specific quantum device affects how an algorithm is compiled for that device, which in turn influences the depth of the quantum circuit, i.e., the number of sequential layers of gates that are executed. A deeper circuit typically requires more time to run and is more susceptible to noise and decoherence.

Other entangling gates include the controlled-Z (CZ) gate and the iSWAP gate. More complex multi-qubit gates, such as the three-qubit gate controlled-controlled-NOT (ccNOT), often referred to as the Toffoli gate, enable smaller circuit depths because the gate encapsulates multi-qubit logic in a single operation, thereby reducing the need for decomposing it into multiple two-qubit gates and improving computational efficiency in both simulation and hardware execution.

3.2 Quantum algorithms

A quantum algorithm is typically implemented as a quantum circuit: a sequence of quantum gates applied to an initial quantum state, followed by measurement. As illustrated in Fig. 3.6, the circuit begins with state preparation, proceeds through unitary operations, and concludes with measurements on each qubit. To obtain the entire probability distribution over output bitstrings, which is often necessary, we must execute the circuit multiple times. However, some methods are more efficient and can extract useful information with significantly fewer runs. This probabilistic nature is intrinsic to quantum computation and forms the basis for potential algorithmic speedups.

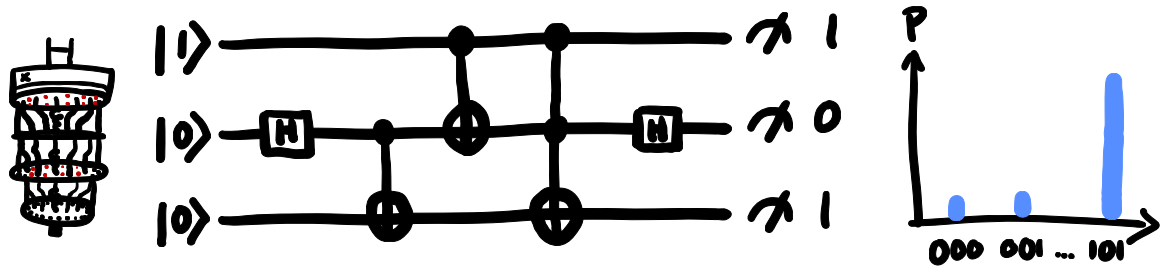

Figure 3.6: **A quantum circuit** consists of a sequence of quantum gates acting on an initial register of qubits, followed by measurement. The register starts in an initial state of $|100\rangle$, five gates operate on the initial state [$\mathbb{H}$, CNOT, CNOT, Toffoli, $\mathbb{H}$], and finally a measurement of each qubit is performed to give the state $|101\rangle$. By repeatedly executing the same circuit, one obtains a probability distribution over the possible output bitstrings.

The long-term vision for quantum computing involves a large-scale, fault tolerant quantum computer capable of executing complex algorithms with high fidelity. The two most famous examples are Shor's algorithm for prime factorization [6] and Grover's algorithm for unstructured search [7]. In theory,

these two algorithms could provide exponential and polynomial speedups, respectively, over their classical counterparts, if run on a fault tolerant quantum computer. These algorithms highlight the potential of quantum computing and have motivated global efforts to build scalable and robust quantum hardware. However, both algorithms require fully error-corrected quantum devices, with logical qubit counts that far exceed current technological capabilities, where each logical qubit is encoded using multiple physical qubits to mitigate errors. In the case of Shor's algorithm, estimates suggest that approximately 20 million noisy physical qubits, or around 6,000 logical qubits, would be required to match the performance of modern classical cryptographic standards [111].

In today's quantum systems, noise and decoherence represent two of the most significant challenges to reliable quantum computation. Noise refers to unwanted interactions between the quantum system and its environment, leading to stochastic errors in gate operations, state preparation, and measurement. Decoherence, more specifically, refers to the loss of quantum coherence resulting from these interactions. This process destroys superposition and entanglement, undermining the computational advantage of quantum algorithms. The rate of decoherence is typically characterized by relaxation times, which quantify the dissipation of energy and the randomization of phase. Mitigating noise and decoherence is a central focus of quantum error correction, fault-tolerant circuit design, and hardware engineering.

Until fault-tolerant quantum computers become available, research has focused on developing algorithms suitable for noisy intermediate-scale quantum devices, quantum processors with 50 to a few hundred qubits and limited coherence times [82, 112–114]. Despite their limitations, NISQ devices have shown potential in outperforming classical systems for specific tasks, but have yet to surpass the capabilities of classical computers.

Among the most prominent NISQ-era algorithms are the variational quantum algorithms (VQAs) and two examples are the quantum approximate optimization algorithm (QAOA) [14], inspired by quantum annealing [115, 116], and the hardware-efficient ansatz (HEA) [16], which tailors circuit structure to the native gate set of the quantum processor. These approaches exemplify the ongoing effort to extract meaningful computation from imperfect quantum hardware by leveraging the combined strengths of quantum and classical resources, a paradigm known as *hybrid quantum computing*. Hybrid quantum-classical algorithms have attracted considerable attention lately and have the potential to provide a quantum advantage for specific computational problems [117]. Hybrid algorithms are highly adaptable and thereby find applications in various domains, including chemistry [16], machine learning [13], and optimization [14].

Other relevant modern algorithms for quantum PSP and design are the quantum adiabatic algorithm [118] and the concept of quantum walk. To simulate a quantum algorithm or run real quantum hardware, we need a quantum software framework. Among these many, *Qiskit* [119] stands out as the most widely adopted open-source platform, offering tools for circuit design, simulation, and execution on IBM Quantum hardware. However, the ecosystem is diverse: alternatives like PennyLane [120] emphasize hybrid quantum-classical workflows

and machine learning integration, while Cirq [121], developed by Google, provides specialized tools for near-term quantum devices. Other platforms, such as Amazon Braket [122], further expand the landscape, providing researchers with the flexibility to choose the framework best suited to their algorithmic and hardware requirements.

3.2.1 Quantum approximate optimization algorithm

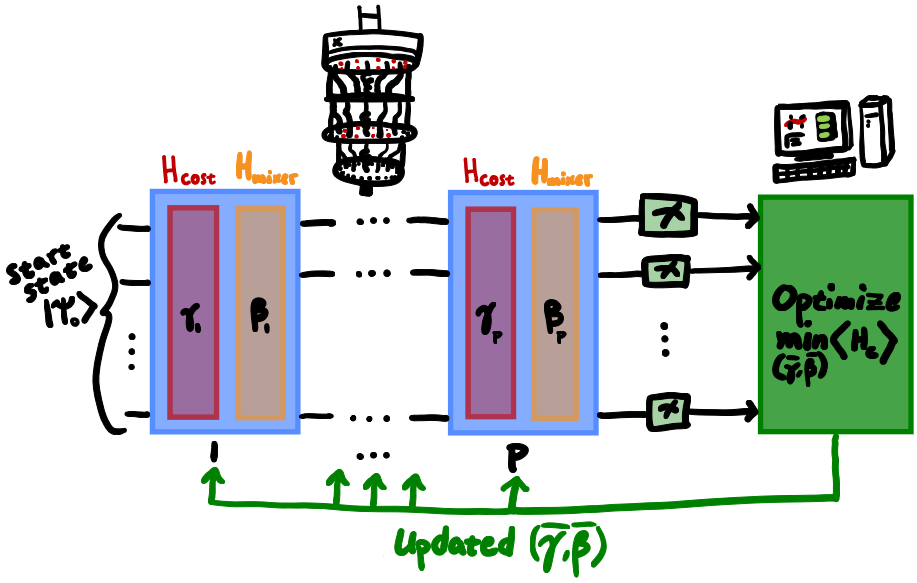

Figure 3.7: **The Quantum Approximate Optimization Algorithm** constructs a layered quantum circuit, where each layer alternates between problem-specific gates in H_{cost} and mixing operations in H_{mixer}. The circuit is parameterized by a set of angles $\vec{\gamma}$ and $\vec{\beta}$, which are iteratively optimized by a classical optimizer. At each step in the parameter optimization, the classical optimizer queries the quantum circuit several times to evaluate the cost function, usually the expectation value of H_{cost}, and updates the parameters accordingly.

The quantum approximate optimization algorithm [14] is to solve combinatorial optimization problems on NISQ devices, and has been studied extensively for its potential advantages over classical methods both in terms of solution quality and computational speed [123, 124]. Inspired by quantum annealing, see sec. 3.2.4, it may be favorable for tackling NP-hard and NP-complete problems, where classical algorithms often struggle with scalability and efficiency. QAOA is a hybrid quantum algorithm in which a parameterized quantum circuit is executed on a quantum processor, while a classical optimizer iteratively updates the circuit parameters to maximize or minimize a problem-derived cost function, see Fig. 3.7.

The standard QAOA circuit consists of an alternating sequence of two Hamiltonians: the *cost Hamiltonian* H_{cost}, which encodes the energy function of the optimization problem E, and the *mixer Hamiltonian* H_{mixer}, which introduces quantum superpositions to explore the solution space. Given an initial state of

the circuit $|\psi_0\rangle$, the quantum state after p layers of the algorithm is given by:

$$\left|\psi_p(\vec{\gamma},\vec{\beta})\right\rangle = \prod_{j=1}^{p} e^{-i\beta_j H_{\text{cost}}} e^{-i\gamma_j H_{\text{mixer}}} |\psi_0\rangle \,, \tag{3.23}$$

where $\vec{\gamma} = (\gamma_1, \ldots, \gamma_p)$ and $\vec{\beta} = (\beta_1, \ldots, \beta_p)$ are variational parameters optimized by the classical part of the algorithm, and i is the imaginary unit in this case. The objective function is often set to maximize or minimize the expectation value:

$$C(\vec{\gamma},\vec{\beta}) = \left\langle \psi_p(\vec{\gamma},\vec{\beta}) \right| H_{\text{cost}} \left| \psi_p(\vec{\gamma},\vec{\beta}) \right\rangle . \tag{3.24}$$

The optimal parameters should yield a quantum state with a high probability of measuring the ground state solution to the problem being implemented. See Sec. 3.2.3 for more details on parameter optimization.

Cost Hamiltonian

To map a classical bitstring solution $\vec{x} = (b_0 \ldots b_{M-1})$, with $b_i \in \{0, 1\}$, each classical binary variable b_i maps to a quantum variable σ_i^z that can take values $\{1, -1\}$ as $b_i \mapsto (1 - \sigma_z^i)/2$. The cost Hamiltonian can then be expressed as a k-local spin Hamiltonian:

$$H_{\text{cost}} = \sum_i h_i \sigma_i^z + \sum_{ij} J_{ij} \sigma_i^z \sigma_j^z + \sum_{ijk} J_{ijk} \sigma_i^z \sigma_j^z \sigma_k^z + \ldots , \tag{3.25}$$

where h_i, J_{ij}, and J_{ijk} are coefficients representing single-qubit, two-qubit, and higher-order interactions, respectively, and σ_i^z is the Pauli Z-matrix acting on qubit i. The cost Hamiltonian guides the evolution of the quantum state toward configurations that minimize the encoded energy function $E(\vec{x})$.

If the problem at hand can be formulated as a quadratic unconstrained binary optimization (QUBO) problem:

$$\min_{\vec{x}} \quad \sum_{i \leq j = 0}^{M-1} Q_{ij}\, b_i b_j, \tag{3.26}$$

where $\vec{x} = (b_0 \ldots b_{M-1})$ is a binary vector with $b_i \in \{0, 1\}$ and $Q \in \mathbb{R}^{M \times M}$ is a symmetric matrix of coefficients. The translation into an Ising Hamiltonian for use as a cost Hamiltonian is easily accomplished, as shown in App. A.1. Many NP-complete and NP-hard problems can be mapped to the Ising model [125].

The periodicity of the cost function landscape plays a critical role in the optimization process. When the eigenvalues E_i of the cost Hamiltonian are rationally independent, the energy landscape becomes non-periodic, which can complicate the search for optimal circuit parameters. If the eigenvalues are integers, they are guaranteed to be rationally dependent, leading to a periodic energy landscape that is easier to navigate. However, when the eigenvalues are non-integer decimals, we can apply smoothing techniques or discretization, such as truncation or rounding, to induce rational dependence and thereby facilitate the identification of approximately optimal parameters.

Mixer Hamiltonian

The mixer Hamiltonian introduces transitions between basis states. The simplest choice is the X-mixer:

$$H_{X-\text{mixer}} = \sum_{i=0}^{M-1} \sigma_i^x, \tag{3.27}$$

which allows transitions between all computational basis states. More sophisticated mixers, such as the XY-mixer [126], can preserve the Hamming weight of the solution (the number of ones in the bitstring), thereby fulfilling a constraint that can be expressed as a Hamming weight. The construction of the XY-mixer is then:

$$H_{XY-\text{mixer}} = \frac{1}{2} \sum_{i,j \in e} (\sigma_i^x \sigma_j^x + \sigma_i^y \sigma_j^y), \tag{3.28}$$

where e denotes the set of edges of a graph defining allowed transitions. Two examples are explored in Paper B. A fully connected graph, which allows interactions between all qubit pairs, and a ring topology, which restricts interactions to nearest neighbors in a cyclic arrangement. Using an XY-mixer can reduce or eliminate the need for penalty terms for punishing unfeasible solutions in the cost Hamiltonian. Initial states can be chosen either in a feasible solution or a uniform superposition over all states in the feasible solution set. Such uniform supersposition corresponds to a so-called *Dicke state* when the Hamming weight is fixed.

3.2.2 Hardware-efficient ansatz

The hardware-efficient ansatz approach [16] is designed to reduce circuit depth and gate complexity compared to other VQAs, making it particularly suitable for implementation on NISQ devices. In this framework, quantum circuits are constructed from alternating layers of parameterized single-qubit rotations and entangling gates, e.g., CNOTs, arranged to reflect the native connectivity of the quantum hardware. The parameters of the rotations are denoted $\vec{\theta}$, see Fig. 3.8 for an illustration of the circuit and information flow. This structure enhances hardware compatibility, while offering sufficient expressibility for many variational tasks [117, 127].

Unlike problem-specific ansatzes such as QAOAs, which incorporate knowledge of the problem into the circuit design, HEAs can be *problem-agnostic*. This means they can be applied to a broad class of optimization problems without requiring structural tailoring, making them highly versatile for general-purpose variational tasks and potentially reducing circuit depth, thereby lessening the demand for a long coherence time.

A problem-informed HEA can use the same objective function as QAOA, described in Eq. 3.24, the expectation value given parameters $\vec{\theta}$:

$$C(\vec{\theta}) = \left\langle \psi(\vec{\theta}) \right| H_{\text{cost}} \left| \psi(\vec{\theta}) \right\rangle. \tag{3.29}$$

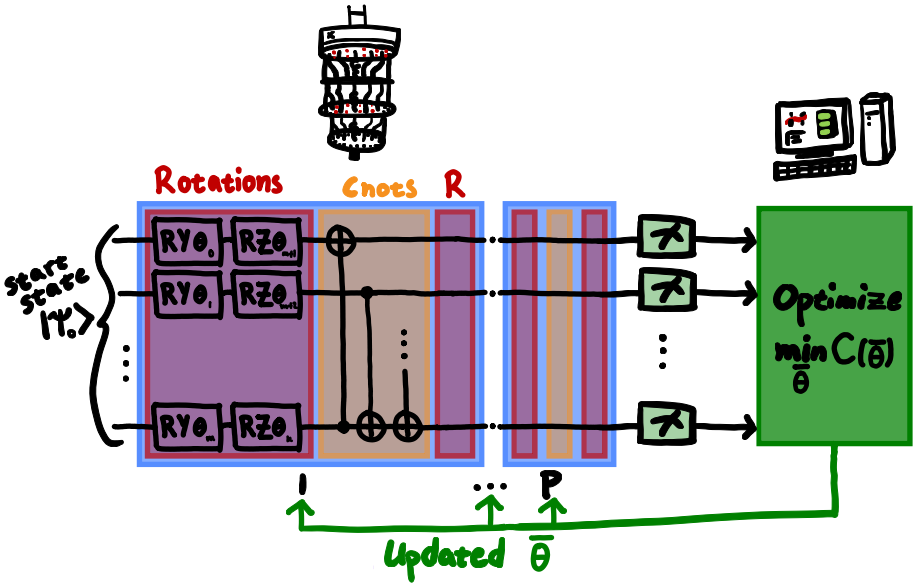

Figure 3.8: The **Hardware-Efficient Ansatz** consists of alternating layers of parameterized single-qubit rotations and entangling gates arranged to match the native connectivity of the quantum hardware. The parameters of the rotation gates $\vec{\theta}$ are optimized by a classical optimizer to minimize a cost function, such as the expectation value of a problem Hamiltonian or the energy function of the problem.

However, the problem-agnostic HEA may utilize an objective function where the output from the quantum computer running the parameterized circuit is processed fully classically, thereby avoiding the use of ancilla qubits for slack variables. This is used in Paper C.

A variety of problem-agnostic HEAs exist. The simplest version, called `RealAmplitudes` [128] in the Qiskit package, consists of a single-layered circuit with a sequence of parameterized single-qubit rotations, typically R_y gates, followed by a layer of entangling gates, such as CNOTs, and then another layer of single-qubit rotations before measurement. The entangling gates are often arranged in a reverse linear pattern, where the last qubit is entangled with the second-to-last, and so on. In this configuration, both the circuit depth and the number of trainable parameters scale linearly with the number of qubits. We could also add another block of rotations after the CNOTs in each layer, called `EfficientSU2` in the Qiskit package, in the pursuit of building an ansatz with sufficient expressivity. Both of these ansatzes are used in the papers of this thesis.

Designing a problem-agnostic ansatz presents a fundamental challenge in quantum algorithm development. Much like selecting an architecture for a neural network so that the untrained network can learn the symmetries and constraints of the problem, the ansatz begins as a blank canvas before the parameters are optimized, and must have a structure that is trainable to express the nuances inherent to the problem at hand. The choice of ansatz is critical: it must be expressive enough to represent the solution space effectively, yet not so complex that it introduces an overwhelming number of parameters. Excessive

parameterization can lead to optimization difficulties.

3.2.3 Optimization of quantum circuit parameters

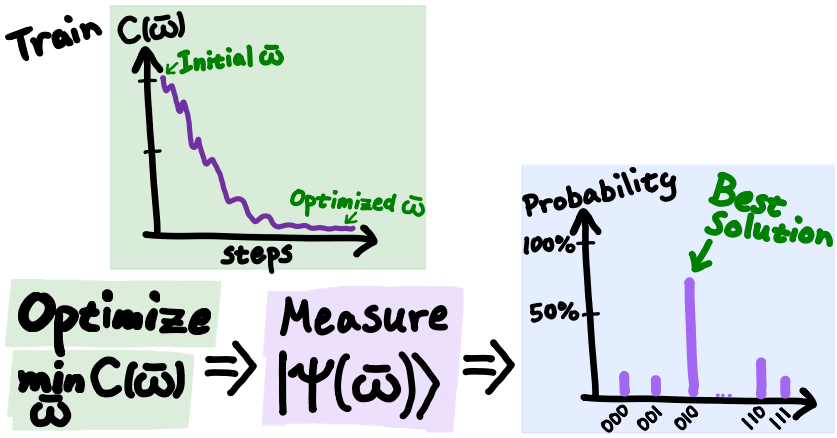

Figure 3.9: **Iterative training of quantum circuit parameters.** The quantum circuit parameters $\vec{\omega}$, such as $\vec{\gamma}, \vec{\beta}$ in QAOA or $\vec{\theta}$ in HEA, are optimized to minimize a cost function $C(\vec{\omega})$, e.g., the expectation value in Eq. 3.24. Successful optimization results in a decreasing cost across iterations, yielding a probability distribution where high-quality solutions are more likely to be observed.

The classical optimization of quantum circuit parameters in VQAs, visualized in Fig. 3.9, is a central yet challenging task, known to be NP-hard [129]. This difficulty is compounded by the presence of noise and barren plateaus [130, 131], regions in the parameter landscape where gradients vanish and optimization stagnates, see Fig. 3.10. These issues necessitate a careful balance between circuit expressibility and depth, especially for ansatzes like HEA, which are known to exhibit barren plateaus as system size increases due to circuits having large expressibility [132] or generating large quantities of entanglement [133].

Even in scenarios where barren plateaus are absent, training variational quantum circuits can still be hindered by other optimization challenges. The parameter landscape may be highly irregular, with numerous suboptimal local minima that make it challenging to identify the global optimum [134]. This issue is intensified when the model lacks symmetry or when a suitable initialization strategy is unavailable (we do not know a good starting point), both of which can significantly reduce the likelihood of successful convergence. In such cases, although the energy landscape may not be flat, its complexity can still render the model effectively untrainable.

To mitigate these challenges, various optimization strategies have been developed. Gradient-based methods, such as gradient descent, were commonly used initially but have been shown to be often inefficient. In contrast, gradient-free approaches, like evolutionary algorithms, show more promise. In our work, we found that the `COBYLA` algorithm [135] consistently performed very well, and

we successfully applied it in both Paper B and Paper C.

One effective strategy is to *optimize circuit parameters iteratively, layer by layer* [136]. Starting with a one-layer circuit, and initializing parameters to an arbitrary value, such as π or a random value, then optimizing the parameters or performing a grid search to obtain the optimal parameters. Performing a grid search on only the first layer's parameters is more feasible as it includes only a handful of parameters, e.g, only two for QAOA. For each increment in depth, the parameters from the previous layer are used to generate an initial guess for the next via linear interpolation. This method is used for QAOA in Paper B.

Another approach involves *initializing parameters based on quantum annealing schedules* [137], which provides the optimizer with a good starting point for initiating the optimization at a higher p. Yet another useful technique for higher p is *parameter donation* [138–141], where optimized parameters from smaller problem instances are reused as initial values for larger ones. This method leverages structural similarities between instances to accelerate convergence. This method is used for HEA in Paper B.

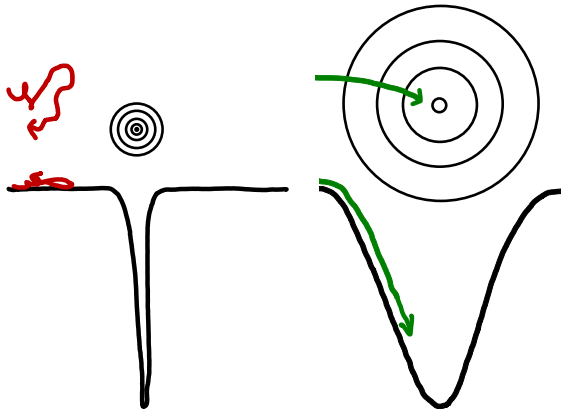

Figure 3.10: Illustration of a **barren plateau** in the cost landscape, where gradients vanish and hinder optimization, versus a smooth sloped landscape that is easy to optimize.

Metrics of performance

To evaluate the performance of our quantum algorithms, we employ several metrics, each offering distinct insights into the quality of the solutions produced. The output of the quantum circuit is typically assessed relative to the energy of the lowest-energy configuration, known as the ground state energy, E_{gs}. Even if the exact ground state is not reached, proximity to it can be beneficial, as post-processing techniques may help refine near-optimal solutions into exact ones. For PSP, such post-processing could be using simulated annealing [142] or the Rosetta suite [94].

The parameters optimized in a quantum circuit can be denoted generally by $\vec{\omega}$, which may correspond to $\vec{\gamma}, \vec{\beta}$ in QAOA or $\vec{\theta}$ in HEA, see Fig. 3.9. Given an objective or cost function dependent on these parameters, $C(\vec{\omega})$, e.g., the expectation value in Eq. 3.24 for QAOA and Eq. 3.29 for HEA, we can then

evaluate and update $\vec{\omega}$. An alternative to using the expectation value is the Conditional Value at Risk [143] (CVaR) function, a risk-aware objective function that improves solution quality by focusing on the best-performing subset of measurement outcomes. Instead of optimizing the full expectation value, CVaR minimizes the average cost over the lowest-energy fraction of sampled states, thereby reducing the influence of high-energy outliers and enhancing the probability of sampling near-optimal solutions. Since the ground state energy can be computed classically, the best measurement outcome directly reflects the optimal solution. Therefore, it is more practical to focus on improving the quality of the best sampled solution rather than the average over all samples.

To quantify how close the sampled solutions are to the ground state energy, we use the *ratio of approximation (RA)*, defined as:

$$\mathrm{RA}(\vec{\omega}) = \frac{C(\vec{\omega})}{E_{\mathrm{gs}}}, \tag{3.30}$$

where a value close to one indicates high-quality solutions [136].

For minimization problems, especially those where the optimal energy is negative, the *relative error (RE)* is often more informative than RA. It is defined as:

$$\mathrm{RE}(\vec{\omega}) = \left| \frac{C(\vec{\omega}) - E_{\mathrm{gs}}}{E_{\mathrm{gs}}} \right|. \tag{3.31}$$

A relative error near zero implies that the quantum circuit produced a low-energy state close to the ground state.

Another important metric is the *success probability*, *overlap with the optimal solution* or *hit rate*, which measures the probability of sampling the correct solution:

$$P_{\mathrm{sp}}(\vec{\omega}) = \left| \langle \vec{x}_{\mathrm{native}} | \psi(\vec{\omega}) \rangle \right|^2, \tag{3.32}$$

where $|\vec{x}_{\mathrm{native}}\rangle$ is the basis state corresponding to the optimal solution, the ground state.

In addition to scalar metrics, visualizing the *energy distribution* of sampled states can provide valuable insights. It helps identify gaps in the optimization process, such as insufficient exploration of certain energy regions or concentration around suboptimal local minima.

3.2.4 Quantum annealing

Quantum annealing [118] is a metaheuristic optimization technique that leverages quantum superpositions to explore the solution space of combinatorial problems. Unlike gate-based quantum algorithms described above, quantum annealing employs a different hardware architecture with a unique type of qubit, often referred to as an "annealing qubit". Experimental implementations of quantum annealers have been developed using various qubit architectures, including ion traps [144], optical lattices [145], and, the most well-known from D-Wave, using a type of superconducting qubit called a flux qubit [146, 147].

Quantum annealing operates by slowly evolving a quantum system from an initial Hamiltonian H_{init} with an easy-to-prepare ground state, often a uniform superposition over all basis states, to a problem-specific cost Hamiltonian H_{cost} whose ground state encodes the optimal solution. The cost Hamiltonian is most often formulated like an Ising Hamiltonian from a QUBO, namely the cost Hamiltonian for the QAOA in Eq. 3.25 with only the one- and two-body terms.

The process of a quantum annealer relies on the adiabatic theorem, which ensures that if the evolution is slow enough, the system remains in its ground state throughout the process:

$$H(t) = [1 - s(t)]H_{\text{init}} + s(t)H_{\text{cost}}, \tag{3.33}$$

where $s(t)$ is the annealing scheme, an often linear function following $s(t_0 = 0) = 0$ and $s(t_{\text{end}} = T)$, with T being the total running time. The evolution is slow enough if T is in the order of $\mathcal{O}(1/\Delta E_{\text{min}}^2)$, where ΔE_{min} is the minimum energy gap between the evolution's two lowest eigenstate energies. Selecting an appropriate annealing scheme is a research field within itself. The QAOA can be seen as a discretized version of quantum annealing, with the initial Hamiltonian corresponding to the mixer Hamiltonian. However, in comparison, QAOA can learn non-adiabatic mechanisms once optimized.

Quantum annealing is particularly well-suited for problems with rugged energy landscapes. While it does not guarantee finding the global optimum, it often provides high-quality approximate solutions. Examples in PSP and protein design with the quantum annealer have seen success [24–27, 35, 84]

3.2.5 Quantum walk

Quantum versions of random walks [148, 149] were introduced as quantum analogues of classical random walks [150, 151], which have found widespread use in modeling stochastic processes across disciplines [152–158]. This foundational role inspired the development of quantum walks as algorithmic tools [159, 160], with demonstrated advantages in specific graph traversal problems [161]. These speed-ups have been linked to quantum transport phenomena, such as charge and energy transfer in biological systems [162, 163].

Quantum walks have been proposed as a foundational mechanism for designing quantum search algorithms, offering an alternative to amplitude amplification techniques such as Grover's algorithm. In particular, discrete- and continuous-time quantum walks have been shown to enable search on structured graphs with provable speed-ups over classical counterparts [159]. These approaches exploit quantum interference to concentrate probability amplitudes on marked vertices, potentially reducing the number of steps required to locate a target. Moreover, quantum walks have been demonstrated to support universal quantum computation [164–166], further reinforcing their theoretical significance.

In Paper E, we model both classical and quantum walks on an undirected graph $G(e, v)$, where v is the set of nodes and e the set of edges. In the classical

case, a *continuous-time random walk (CTRW)* begins at the start node and aims to reach a target node. The walk is governed by a modified adjacency matrix A^c, which prevents transitions out of the target node to simulate absorption. The time evolution of the node occupation probabilities $p(t)$ follows the equation

$$\frac{dp(t)}{dt} = (T - I)p(t), \tag{3.34}$$

where $T_{ij} = A^c_{ij}/\deg(j)$, and $\deg(j)$ is the degree of node j. The solution,

$$p(t) = e^{(T-I)t}p(0), \tag{3.35}$$

with $p(0) = (1, 0, \ldots, 0)^T$, allows estimation of the hitting time, the time it takes for the particle to reach the target node with sufficient probability. For random graphs, the characteristic path length is $\propto \ln n$, such that $p_{\text{th}} \propto \frac{1}{\ln n}$ [167, 168].

In the quantum case, a *continuous-time quantum walk (CTQW)* is defined on the same graph with Hilbert space $\mathcal{H}_{G(e,v)} = \{|0\rangle, \ldots, |n-1\rangle\}$ and Hamiltonian $H = A$. To avoid collapsing the quantum state during measurement, a sink node is added, connected only to the target node. The Lindblad equation describes the system's dynamics so that we can model decay from the target to the sink [169, 170]. The hitting time is then extracted from the population in the sink node, starting from the initial state, in only the initial node.

The quantum walk framework described here shares conceptual similarities with the *Quantum Metropolis algorithm* [171], which is commonly used in quantum PSP [33, 172], particularly in its application of graph-based dynamics to explore complex state spaces. In both cases, transitions between states (nodes) are governed by adjacency relations, and the goal is to efficiently sample or reach specific configurations. The Quantum Metropolis algorithm adapts the classical Metropolis sampling algorithm to quantum systems by employing controlled quantum walks and projective measurements to enforce detailed balance and ensure convergence to a target distribution. While our quantum walk model focuses on hitting times and absorption via a sink node, the underlying mechanism (quantum evolution constrained by graph topology) aligns with the Metropolis approach to navigating energy landscapes. However, unlike the Metropolis algorithm, which is designed for sampling from thermal distributions, our model is tailored to directed search and does not inherently guarantee equilibrium sampling. This distinction highlights both the versatility and the limitations of quantum walks as algorithmic primitives in quantum simulation and optimization.

4 Structure and sequence on NISQ devices

In this chapter, we present an integrated view of how NISQ devices can be used for PSP and protein design. We then review prior work in this domain, highlighting key contributions that have shaped the field, and also discuss the specific contributions of the papers included in this thesis.

Figure 4.1 illustrates a typical workflow often proposed in the literature where a quantum computer and a classical computer work in tandem.

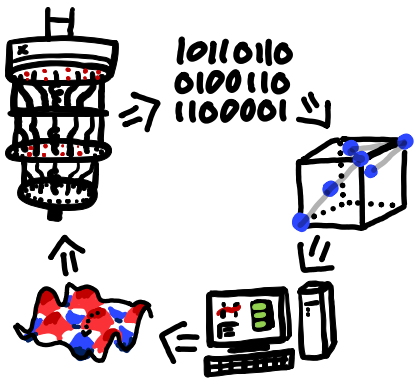

Figure 4.1: **Hybrid quantum-classical workflow for PSP.** Bitstrings representing protein conformations, generated from the quantum computer, are processed by the classical computer. The classical computer evaluates the output and, with an optimizer, updates the quantum circuit parameters, penalizing invalid configurations and favoring feasible low-energy conformations. Towards the goal of finding the native structure of the protein.

4.1 Overview of the procedure

Applying quantum computing to PSP involves a sequence of interdependent steps, each requiring trade-offs between biological accuracy and the limitations of today's quantum devices. These limitations include the number of available qubits, the connectivity and allowed interactions between the qubits, and the achievable circuit depth given current gate fidelities.

The process begins with formulating the biological problem as a mathematical optimization problem. This step includes selecting a suitable coarse-grained model and defining an energy function that captures the relevant interactions and constraints. Once the problem is formulated, it must be encoded into a representation suitable for quantum compu-

tation. This involves mapping the protein conformation into a bitstring and constructing a quantum circuit that implements the energy function, often, but not necessarily, expressed as a Hamiltonian.

Next is the choice of quantum algorithm and hardware platform, which is closely tied to these earlier decisions. For instance, if the hardware supports only two-body interactions, a model with at most two-local interactions is allowed; otherwise, the circuit must be decomposed into smaller gates, which increases circuit depth and coherence time requirements. Similarly, the choice of lattice model affects both the accuracy of the predicted structure and the number of qubits and gates required, which in turn limits the size of the protein instance that can be processed if the hardware is limited. Paper A explores these trade-offs in detail, analyzing the long-term potential of quantum computing for protein folding and evaluating resource requirements for simplified yet computationally challenging models.

Finally, once the quantum computation is complete, and we have found a good candidate structure or even the ground state structure of the model, the coarse-grained structure can be converted into three-dimensional coordinates and optionally refined. The predicted structure is then compared to a reference from the PDB using metrics such as the Root Mean Square Deviation (RMSD):

$$\text{RMSD} = \sqrt{\frac{1}{\#\text{atoms}} \sum_i \|\mathbf{x}_i - \mathbf{y}_i\|^2}, \tag{4.1}$$

where $\mathbf{x}_i$ and $\mathbf{y}_i$ are the coordinates of corresponding atoms in the predicted and reference structures. Lower RMSD values indicate higher accuracy, with values below 2 Å generally considered good.

4.1.1 Problem Formulation

The first step in the workflow is to express PSP or protein design as an optimization problem, as discussed in Sec. 2.5. This requires defining an energy function that captures the relevant interactions, such as HP or MJ potentials, and imposing constraints to ensure physical feasibility, such as avoiding overlaps and maintaining chain connectivity. The choice of model, whether lattice-based or off-lattice, has a significant impact on both the accuracy of the representation and the computational resources required. Paper A analyzes the effect of lattice choice on resource requirements, discussing the number of qubits, interaction, and two-qubit gates needed for various lattice and off-lattice models. Meanwhile, Paper C introduces a new lattice and compares three different lattice types, demonstrating their influence on qubit counts.

4.1.2 Encoding

Problem formulation is closely intertwined with the encoding process, which bridges the gap between the abstract optimization problem and its quantum

implementation. We need a representation of the protein structure as a bitstring. The string of integers representing the protein conformation, discussed in Sec. 2.5, can be mapped to a bitstring $\vec{x} = (b_0, \dots b_{M-1})$, with $b_i \in \{0, 1\}$, if we translate the integers into a binary number and concatenate them.

There are several ways of translating from an integer into a bitstring, either by a Unary, sometimes called One-Hot, Binary, BUBinary, Domain wall, or something else [173]. An example of how integers are translated into a bitstring is given in the continued version of Fig. 2.8, which is shown in Fig. 4.2.

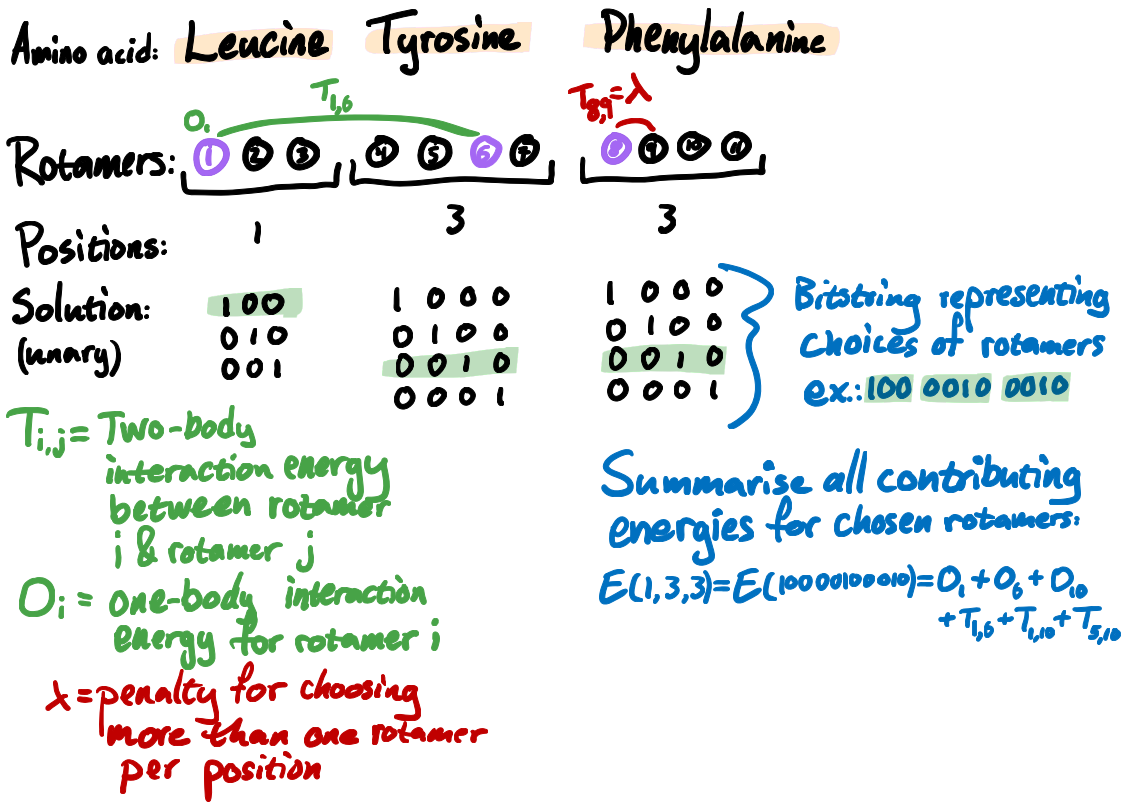

Figure 4.2: **Revisiting the example of a structure in the rotamer model** with the bitstring encoding. The solution (purple) is $\vec{x} = (1, 3, 3)$, which results in the bitstring `10000100010` in the Unary encoding. If two rotamers for the same amino acid are chosen, e.g., rotamers eight and nine that both correspond to the third side chain resulting in the last subbitstring `1100`, in the solution. A constraint we need to add is to keep the Hamming weight of each subbitstring to one, so that only one rotamer is chosen per amino acid. The interaction energies between the rotamers for the same amino acid are set to a high value to avoid unfeasible solutions (red).

The cardinality vector, defined in Paper A and written in Sec. 2.5, helps us allot the correct number of bits to encode the integer. For example: the Unary encoding will need $M = n(\vec{R}_i)$ qubits and the Binary encoding will need M bits so that $2^M \leq n(\vec{R}_i)$. Here, a choice arises regarding how to encode the integers, presenting a trade-off between the number of needed qubits and the complexity

of the resulting energy function. In Fig. 4.3, we give an example of how the choice of the encoding affects the number of qubits. Comparing turn-based and coordinate-based encodings can also be seen in Fig. 4.3. The choice of encoding is the main topic of Paper A, analyzing the effect of choosing Unary, Binary, or BUBinary.

4.1.3 Algorithm selection

The next step is to choose a quantum algorithm, possibly one from Sec. 3.2, and construct a circuit that encodes the energy function. This can be done using a problem-informed ansatz, where the circuit structure is derived from the problem Hamiltonian, or a problem-agnostic ansatz, which is tailored to the hardware and learns the energy function through optimization. Alternative approaches may also be considered depending on the problem and hardware constraints.

Constraints can be incorporated as penalty terms within the Hamiltonian. While this approach is straightforward, it introduces an additional hyperparameter: the penalty coefficient. This parameter must be carefully tuned; if set too high, it can overshadow the objective function, whereas if too low, the algorithm may favor violating constraints to minimize overall energy. In the case of the QAOA, we can employ alternative strategies, such as relocating certain constraints from the cost Hamiltonian to the mixer Hamiltonian, e.g., using an XY-mixer as described in Sec. 3.2.1, which can reduce the total circuit depth.

Once a parameterized circuit is defined, the next challenge is to optimize its parameters, a process discussed in Sec. 3.2.3. Consequently, the choice of how the parameters are trained and which classical optimizer is used can have a substantial impact on solution quality and convergence.

4.1.4 Quantum hardware

The final step in the workflow involves selecting an appropriate compiler and quantum hardware. This choice is critical because the connectivity of the problem sets the needed connectivity of the gates, which directly influences the suitability of the hardware. The compiler, analogous to its classical counterpart, translates high-level quantum programs into low-level instructions executable on the target device. Compilation itself is an active research area as it must address several challenges, including transpiling, i.e., adapting to platform-specific native gate sets. For example, if a required gate, such as the Toffoli, is not natively supported, it must be decomposed into an equivalent subcircuit using the available gate set. Additional compilation tasks include *qubit allocation* (mapping logical qubits to physical qubits), *circuit routing* (enabling interactions between non-adjacent qubits), and *gate compression* (reducing overall gate count and circuit depth). These steps are essential for minimizing errors and ensuring that the circuit can be executed within the device's coherence time.

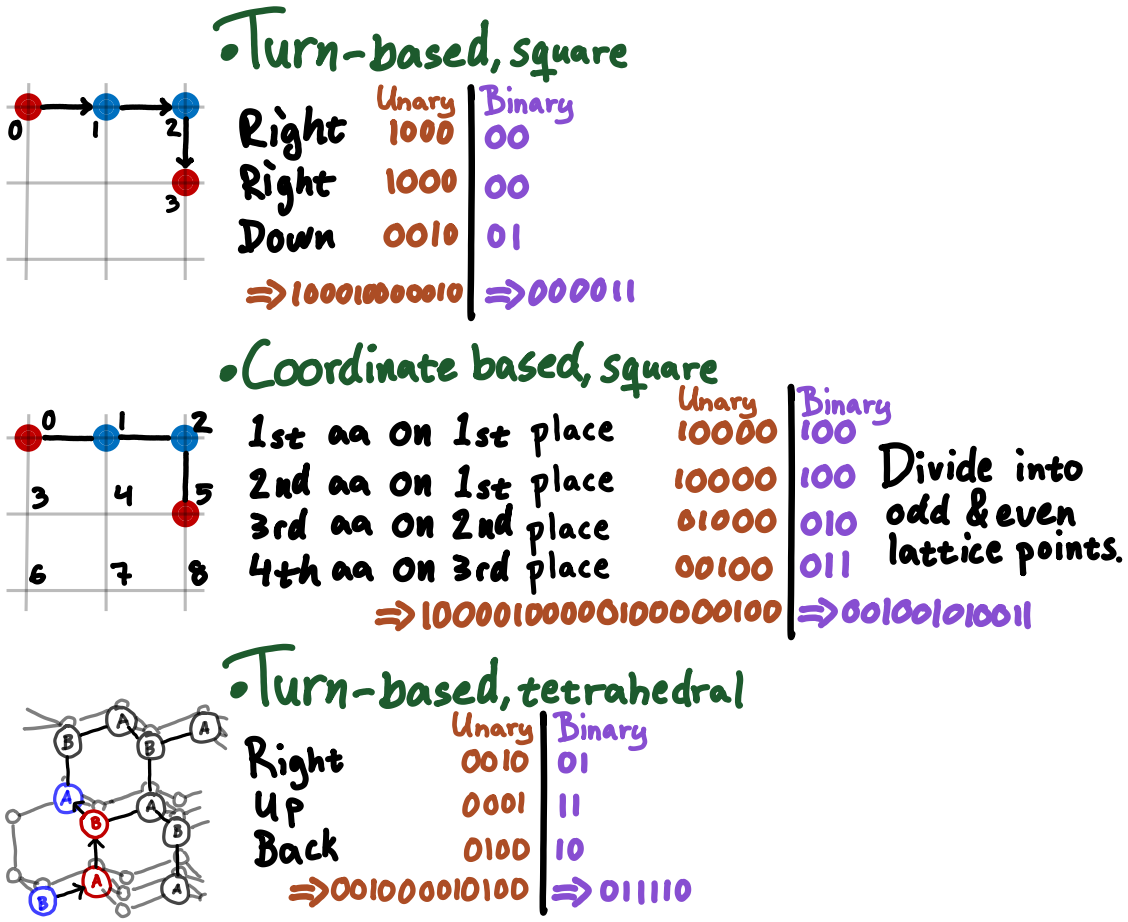

Figure 4.3: **Encoding comparison of examples of lattices with coordination number four.** Turn-based encoding on a square lattice [25], the coordinate-based encoding also on the square [27] and the turn-based encoding on the tetrahedral lattice (3D) [31]. Both the Unary (orange) and Binary (purple) encoding are shown for example protein conformations. It is clear to see that the Unary encoding always results in more qubits than the Binary encoding. The turn-based encoding uses fewer qubits than the coordinate-based encoding. It should be noted that the bitstrings shown in this example are only for the conformation qubits.

Different quantum platforms offer distinct trade-offs. Superconducting qubit systems, for instance, provide fast gate speeds and a solid-state platform but typically feature limited connectivity and are restricted to two-qubit interactions [109]. In contrast, trapped-ion systems offer superior qubit quality and reconfigurable connectivity, enabling more flexible circuit layouts, but at the cost of slower gate operations [108, 174]. Consequently, while some architectures can support multi-qubit interactions with fewer qubits, others prioritize scalability at the expense of connectivity. Selecting the optimal hardware, therefore, requires balancing these factors against the resource requirements of the chosen algorithm and encoding strategy [175].

Paper A presents an estimate of the minimum quantum resources required, namely, the number of qubits, interactions, and two-qubit gates, to implement a heuristic quantum algorithm tailored to a specific PSP instance. Our analysis focuses on the resources required to construct quantum operations derived from the Hamiltonian representation of PSP models for a given amino acid length, and plots these up to a hundred amino acids. In particular, we examine the lattice-based models as well as the fixed-backbone side-chain conformation-based model, evaluating their compatibility with current hardware constraints under different bit-encoding schemes, including Unary, Binary, and BUBinary. We compare the turn-based and the coordinate-based encoding. We conclude that the large number of ancilla qubits for the turn-based encoding and the large number of conformation qubits are the main obstacles for the models. Our findings indicate that the qubit requirements are within the reach of existing quantum technologies. However, the primary bottleneck lies in the large number of interactions present in the Hamiltonian, which translates into a gate count far beyond the capabilities of today's quantum devices.

4.2 Protein structure prediction on NISQ devices

PSP using quantum computing is a relatively young but rapidly growing research area, with a notable increase in publications in recent years. A comprehensive overview of the field in table format is provided in the appendix of Paper D. Below, we present the field of quantum PSP and its evolution over time, summarizing key contributions.

Before delving into the literature, it is worthwhile to clarify the terminology. Early work in this field often used the term *protein folding*, whereas more recent studies tend to adopt the term *PSP*. While these approaches do optimize over bitstrings whose energies correspond to conformational energies, thus favoring compact, folded structures, they do not solve the protein folding problem in its strict sense. Proper folding is a path-dependent process governed by thermodynamics and kinetics, whereas quantum approaches to PSP are largely path-agnostic.

The first proposal to address PSP on a quantum device was introduced by Perdomo *et al.* [28] in 2008, who suggested a square HP-lattice model using the coordinate-based encoding for the quantum annealer. Furthermore, Perdomo-Ortiz *et al.* [29] employed the turn-based model in 2012 to fold small peptides of up to six residues on a square MJ-lattice, running experiments on D-Wave hardware with 81 superconducting qubits.

Building on this idea, in 2014, Babbush *et al.* [25] reduced the number of quantum operations required for the turn-based encoding on the square lattice. Followed by Babej *et al.* [26] in 2018, who optimized the arithmetic complexity of the previous approach and demonstrated PSP of a ten-residue Chignolin protein and an eight-residue Trp-Cage peptide using 2D and cubic lattice models, respectively, on D-Wave's 2000Q quantum annealer with 2048 qubits. Later in the same year, Fingerhuth *et al.* [30] extended this work into the gate-based realm by adopting a unary turn-based encoding on a cubic lattice. This encoding simplified the Hamiltonian at the cost of requiring more qubits and a non-trivial mixer. They successfully implemented the algorithm on a real quantum computer, obtaining non-trivial results for a tripeptide.

Furthermore, the use of a BCC lattice in quantum PSP was first proposed by Wong and Chang [40] in 2021, who combined Grover's search algorithm with the HP model to identify low-energy conformations. The same group later ran Grover's algorithm on the square lattice in Wong and Chang [41].

At the same time, a meta-study of the field was conducted by Outeiral *et al.* [24] in 2021, where they investigated the potential for quantum speedup by analyzing the scaling of the spectral gap in dense encodings as the chain length increases. They observed exponentially closing gaps in worst-case scenarios, but only polynomial scaling on average. They compared simulated annealing with ideal quantum annealing through direct Schrödinger equation simulations for short peptides.

The above-presented models and encodings rely on ancilla qubits to implement interaction terms, both for residue-residue interactions and for preventing overlap. These extra qubits stem from turn-sequence-based conformational encodings, which require storing residue distance data. The core challenge lies in computing the distances between residues. To combat this, Irbäck *et al.* [27] introduced a new coordinate-based encoding in 2022 with the square HP-lattice model without ancilla qubits.

A significant development was introduced by Robert *et al.* [31] in 2021, who proposed the tetrahedral MJ-lattice, to better approximate realistic bond angles while maintaining a coarse-grained representation. Their method was more resource-efficient, and they successfully demonstrated the folding of a seven-residue peptide on IBM's quantum hardware and simulated a ten-residue sequence using a VQE variant with CVaR optimization. However, the method scales poorly, as the number of qubits grows exponentially with the interaction range, limiting practical implementations to nearest-neighbor interactions. The problem of ancilla qubits remained. Despite these limitations, the tetrahedral model gained traction, maybe due to its availability in Qiskit, prompting further analyses and algorithmic adaptations. Subsequent studies highlighted

weaknesses in overlap penalization, which can lead to infeasible structures for sequences longer than ten residues.

Following the success of Robert *et al.* [31], recent works from 2023 and until today have explored both algorithmic and practical aspects using the same tetrahedral MJ-lattice model. Boulebnane *et al.* [83] evaluated QAOA for PSP while using a more complex Lennard-Jones cost function, reporting limited success compared to classical methods, while encoding overlap constraint into the problem rather than the Hamiltonian for better scaling. Doga *et al.* [32] proposed a framework to identify proteins that could benefit from quantum approaches, particularly those with rugged energy landscapes and few homologues, and demonstrated that for a proof-of-principle protein, a hybrid quantum-classical approach achieved lower RMSD than AlphaFold2 after post-processing.

Scheiber *et al.* [176] introduced an adaptation of the coordinate-based model on the tetrahedral grid for quantum annealers. Their analysis revealed that turn-based models on cubic grids performed poorly across most metrics due to the overhead of reducing high-order terms to two-local interactions; in contrast, the tetrahedral turn-based model performed significantly better. Scaling studies further indicated that coordinate-based models are more favorable for quantum annealers, while turn-based approaches remain limited by locality.

Pamidimukkala *et al.* [177] enhanced the 3D cubic HP-lattice model by incorporating diagonal movements, expanding each bead's degrees of freedom to 26. They employ the CVaR-VQE algorithm on both simulators and hardware.

Extensive research has continued on the tetrahedral lattice using various quantum algorithms. Mustafa *et al.* [178] reported promising results using a HEA with average cost and HEA with CVaR, while emphasizing the need for robust encodings to prevent chain overlaps. More recently, Zhang *et al.* [39] performed a large-scale comparison between tetrahedral lattice-based quantum models and AlphaFold2, noting that extensive post-processing was required because quantum measurements often yielded states far from the ground state. Chandarana *et al.* [82] ran the digitized counterdiabatic quantum algorithm, which tackled a nine-amino-acid chain using two superconducting circuit gate-based quantum devices and an ion-trap platform. Followed by Romero *et al.* [38], who presents a bias-field digitized counterdiabatic quantum optimization algorithm, on a fully connected trapped-ion quantum processor, tackling the tetrahedral lattice for up to 12 amino acids. Finally, the most recent Li *et al.* [179] explores the use of circuit cutting as a strategy to improve hardware efficiency by decomposing large quantum circuits into smaller subcircuits, thereby reducing overall routing and mapping overhead [180]. Their work applies this technique to a counter-adiabatic circuit implemented on a tetrahedral lattice, combined with Initial-State-Dependent Optimization. This framework adapts quantum gate parameters based on the specific states encountered during execution to minimize circuit depth. While the study demonstrates promising reductions in circuit complexity, it does not include results on generating actual protein conformations, focusing instead on depth optimization. Nevertheless, the approach appears to be a promising direction for future research.

While the tetrahedral lattice is resource-efficient and thereby well-suited for

NISQ devices, its main drawback is geometric: the protein chain can only extend in four directions, and only one angle can be formed between two consecutive turns. To assess its suitability for predicting secondary structures, a denser lattice with a higher coordination number is needed.

Paper D introduces the first quantum encoding of the FCC lattice for PSP, alongside a detailed comparison with the tetrahedral lattice. The FCC lattice consistently yields lower RMSDs, demonstrating superior protein modeling capabilities. The paper also proposes two novel approaches for constructing the problem Hamiltonian without ancilla qubits: one based on polynomial fitting, which increases circuit depth due to exponential growth in Hamiltonian terms, and another using Lagrangian duality, which avoids this growth and enhances scalability. However, the latter requires careful hyperparameter tuning and a deeper understanding of the cost landscape to ensure convergence. Paper D takes one step towards a more resource-efficient algorithm and a more resource-demanding but more accurate lattice model, and finally runs a six-amino acid sequence on quantum hardware. *Note:* I had the pleasure of contributing to discussions surrounding this excellent paper. However, my involvement in writing and coding was minimal; I'm grateful to have participated in the exchange of ideas.

Paper C continues the use of the FCC lattice, but uses a method to altogether avoid ancilla qubits, hyperparameter tuning, and deep quantum circuits. The paper employs a Hamiltonian-free approach to the quantum PSP problem using a HEA. The ansatz is trained to minimize an entirely classic energy-based cost function, enabling a more scalable solution compared to problem-informed ansatzes. This design not only simplifies implementation but also facilitates the inclusion of higher-order interactions, which are typically challenging to incorporate into quantum models and, to date, have not been run on real hardware. The study evaluates this method across three lattice geometries, tetrahedral, BCC, and FCC, demonstrating the approach's flexibility. Benchmarking is performed on a diverse set of proteins with up to 26 amino acids, incorporating interactions up to the second-nearest neighbor, with potential for even higher-order terms. Experiments are conducted on both noise-free simulators and real gate-based quantum hardware, pushing the limits of current quantum methods by targeting sequences significantly longer than those addressed in previous studies. Overall, the results highlight the scalability and versatility of this Hamiltonian-free framework while identifying key challenges that inform future algorithmic improvements and hardware development.

4.3 Protein design on NISQ devices

4.3.1 Lattice model

Irbäck *et al.* [84] formulated the design problem as a QUBO for the square HP-lattice model in 2024, enabling its solution on quantum annealers.

Paper B builds on Irbäck *et al.* [84] and extends the approach to gate-based quantum devices, exploring both QAOA and HEA. The study focuses on the sequence optimization step, which is less resource-intensive than full folding computations, making it a suitable candidate for current NISQ hardware. Both the QAOA and HEA were used. While QAOA performs well in noiseless simulations, its performance degrades significantly under simulated noise conditions. Conversely, HEA shows improved robustness in both noisy and noiseless simulations. However, when executed on real quantum hardware, performance deteriorates further, likely due to noise characteristics not captured by simulation models, such as temporal noise correlations.

Panizza *et al.* [181], also in 2024, proposes a general protein design framework that integrates machine learning with quantum-inspired optimization. Their method iteratively learns an optimal physics-based scoring function using structure prediction algorithms, rather than relying on predefined interaction parameters. The sequence selection step is then mapped to a combinatorial QUBO formulation, allowing the use of both advanced classical solvers and emerging quantum technologies. The study focuses on compact structures modeled on a two-dimensional square lattice, demonstrating the potential of hybrid approaches that combine predictive modeling with quantum optimization.

Around the same time, Khatami *et al.* [182] investigated the use of Grover's algorithm for square lattice-based protein design. While theoretically appealing, the analysis concludes that the quantum resource requirements are prohibitively large for current NISQ devices, limiting their near-term applicability.

4.3.2 Off-lattice model

The first application of quantum computing to off-lattice protein design was introduced by Mulligan *et al.* [35] in 2020, who integrated the Rosetta modeling suite with D-Wave's quantum annealer. Their QPacker algorithm reformulates the side-chain packing problem as a QUBO, solved via quantum annealing and supplemented by classical post-processing. Subsequent work by Maguire *et al.* [36] introduced preprocessing techniques to reduce the number of rotamers, thereby mitigating resource constraints.

More recently in 2025, Agathangelou *et al.* [37] explored the use of QAOA with an XY-mixer for side-chain packing. Their analysis suggests a potential quantum advantage at a crossover point between 115 and 150 qubits, beyond which quantum optimization could outperform classical methods. This

finding underscores the importance of continued algorithmic and hardware improvements to make quantum approaches competitive for realistic protein design tasks.

4.4 Quantum walk approaches to the protein structure prediction problem

One classical strategy for identifying the native conformation of a protein is to perform a smarter version of a random walk in conformation space. Starting from an initial configuration, the algorithm proposes local changes and accepts or rejects them based on a criterion, typically whether the energy decreases, as in the Metropolis algorithm. Quantum algorithms can enhance this process by introducing quantum versions of either the sampling step, the acceptance step, or both, potentially combining them into a fully quantum procedure.

One approach is to integrate quantum walks with classical acceptance rules. For example, in 2023 Varsamis and Karafyllidis [172] proposed a quantum-walk-assisted algorithm in which dihedral angles are encoded as phase factors governing the evolution of the quantum walk. This method uses a variational quantum circuit to explore a Hilbert space corresponding to different spatial configurations of a given sequence. Their implementation adopts a simplified representation, modeling one bead per side chain and approximating the backbone as a one-dimensional lattice.

A more ambitious strategy is to implement the entire process on a quantum computer. In 2022, Casares *et al.* [33] introduced *QFold*, an algorithm that employs a quantum Metropolis-Hastings approach to determine torsion angles for a tetrapeptide. Unlike lattice-based models, QFold treats torsion angles as continuous variables, allowing for finer structural resolution but significantly enlarging the search space. The algorithm uses machine learning predictions as initial guesses and relies on a precomputed oracle for energy evaluations. While this avoids compiling the scoring function into a quantum circuit, it introduces a conceptual limitation: the oracle effectively encodes the solution in advance by calculating all conformational energies before the quantum run. Constructing the underlying graph for the quantum walk also requires assigning transition probabilities based on these energies, further reducing the algorithm's practical advantage.

Despite these challenges, integrating quantum walks with protein folding remains an intriguing research direction, particularly when combined with machine learning techniques. However, the reliance on oracles and the overhead of graph construction highlight the need for more scalable approaches.

Paper E explores how a classical neural network can detect the potential advantage of quantum walks over classical random walks. We employ a fully dense network, a convolutional network, and a network with problem-informed layers

together with dense layers. Our study demonstrates that enhancing the quality of the training dataset can improve neural network performance; however, all tested architectures encounter difficulties when classifying large random graphs and transferring knowledge across different graph sizes. Achieving higher accuracy could unlock valuable insights into quantum advantage, not only for random walks but also for broader applications in quantum computing and quantum transport.

We can hypothesize that quantum walks may outperform classical methods when the initial conformation is far from the native state, based on Paper E, as the quantum algorithm can exploit interference to accelerate the exploration process. Conversely, when the starting point is close to the target, classical random walks may remain more efficient. The relative performance also depends on the density of the conformation space and the allowed moves within it. These factors suggest that the benefits of quantum walks are highly problem-dependent and warrant further investigation.

5 Summary & outlook

This thesis has explored the current state of quantum approaches to PSP and protein design on near-term quantum computers, with a particular focus on algorithmic strategies, encoding schemes, and the trade-offs between accuracy and resource requirements. We began by introducing the biological and computational foundations of PSP and protein design, followed by an overview of quantum computing principles and algorithms relevant to these problems. Building on this foundation, we examined how protein information can be translated from atomic-level detail into bitstring representations suitable for quantum computation, and how these representations interact with the constraints of NISQ hardware.

5.1 Future

The field is transitioning from proof-of-principle demonstrations to tackling larger and, hopefully soon, biologically relevant proteins. However, this progress depends on four key factors:

Exploring alternative coarse-grained models: Future work could investigate more sophisticated coarse-grained models and interaction potentials, including those derived from learning methods [183], which can capture complex patterns from large structural datasets. Beyond the MJ model, alternative energy matrices could improve biological realism. Comparing different interaction models against experimentally determined structures could maybe help refine modeling strategies and guide the integration of quantum methods. With problem-agnostic ansatz, previously unattainable models, e.g., *CABS* [78], *AWSEM* [79], *MARTINI* [80], or *OPEP* [81], may become a new path to explore.

Advancing quantum algorithms: Beyond the HEA, other problem-agnostic approaches, such as the Instantaneous Quantum Polynomial ansatz [184, 185], may offer advantages in scalability and trainability. Circuit cutting [180], where large quantum circuits are broken down into smaller subcircuits to minimize

routing complexity and reduce mapping overhead, is another promising technique for reducing circuit depth. Empirical studies suggest that even when large quantum machines are available, reducing circuit size can improve fidelity [186, 187].

The challenge of trainability, discussed in Sec. 3.2.3, highlights the need for approaches that strike a balance between hardware efficiency and problem-specific structure. While fully problem-agnostic and fully problem-informed ansatz represent two extremes, hybrid strategies may offer the best compromise. For example, a semi-agnostic ansatz could combine a problem-inspired layer with a problem-agnostic layer [188], analogous to pre-trained and dense layers in neural networks. Variable-structure ansatz, such as ADAPT-VQE [189] and its extensions [190], provide another avenue by iteratively growing or pruning circuits based on empirical performance, potentially improving scalability and tackling barren plateaus.

Identifying quantum utility: Rather than seeking full quantum advantage in the near term, it is pragmatic to identify niches where quantum resources can complement classical methods. These include scenarios where quantum subroutines reduce wall-clock time or energy consumption [191], even if they do not outright outperform classical algorithms. Such hybrid workflows may deliver tangible benefits in life sciences before fault-tolerant quantum computing becomes available.

Integration across the quantum stack: The performance of quantum algorithms depends on the entire computational stack, encompassing hardware as well as compilers and error mitigation. Advances in software frameworks, such as Qiskit Runtime [119], and improvements in error correction will be critical for realizing the potential of quantum algorithms in PSP and protein design. Progress will require close collaboration between quantum computing specialists, computational biologists, and algorithm developers to ensure that innovations at each layer of the stack translate into practical gains.

5.2 Final remarks

The intersection of quantum computing and life sciences is a rapidly evolving research frontier. While significant challenges remain, the progress documented in this thesis demonstrates that quantum computing approaches to PSP and protein design are slowly moving beyond theoretical curiosity toward practical relevance. Continued interdisciplinary collaboration and open exchange of ideas will be essential to unlock the full potential of quantum technologies in computational biology.

Bibliography

[1] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines", Journal of Statistical Physics **22**, 563–591 (1980).

[2] R. P. Feynman, "Simulating physics with computers", International Journal of Theoretical Physics **21**, 467-488 (1982).

[3] Y. I. Manin, "Computable and uncomputable (in russian)", Sovetskoye Radio (1980).

[4] Y. I. Manin, "Classical computing, quantum computing, and shor's factoring algorithm", arXiv:quant-ph/9903008 [quant-ph] (1999).

[5] D. Deutsch, "Quantum theory, the church-turing principle and the universal quantum computer", Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences **400**, 97–117 (1985).

[6] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", SIAM Journal on Computing **26**, 1484–1509 (1997).

[7] L. K. Grover, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96 (Association for Computing Machinery, New York, NY, USA, 1996) pp. 212–219.

[8] K. Wang, Z. Lu, C. Zhang, G. Liu, J. Chen, *et al.*, "Demonstration of low-overhead quantum error correction codes", arXiv:2505.09684 [quant-ph] (2025).

[9] S. Krinner, N. Lacroix, A. Remm, A. Di Paolo, E. Genois, *et al.*, "Realizing repeated quantum error correction in a distance-three surface code", Nature **605**, 669–674 (2022).

[10] R. Acharya, I. Aleiner, R. Allen, T. I. Andersen, M. Ansmann, *et al.*, "Suppressing quantum errors by scaling a surface code logical qubit", Nature **614**, 676–681 (2023).

[11] R. Acharya, D. A. Abanin, L. Aghababaie-Beni, I. Aleiner, T. I. Andersen, *et al.*, "Quantum error correction below the surface code threshold", Nature **638**, 920–926 (2024).

[12] J. Preskill, "Quantum Computing in the NISQ era and beyond", Quantum **2**, 79 (2018).

[13] A. Caunhye, X. Nie, and S. Pokharel, "Optimization models in emergency logistics: A literature review", Socio-Economic Planning Sciences **46**, 4–13 (2012).

[14] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm", arXiv:1411.4028 (2014).

[15] 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 **14**, 1–11 (2020).

[16] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, "Hardware-efficient Variational Quantum Eigensolver for Small Molecules and Quantum Magnets", Nature **549**, 242–246 (2017).

[17] D. Fitzek, T. Ghandriz, L. Laine, M. Granath, and A. F. Kockum, "Applying quantum approximate optimization to the heterogeneous vehicle routing problem", Scientific Reports **14**, 25415 (2024).

[18] P. S. Emani, J. Warrell, A. Anticevic, S. Bekiranov, M. Gandal, *et al.*, "Quantum computing at the frontiers of biological sciences", Nature Methods 10.1038/s41592-020-01004-3 (2021).

[19] K. Rhrissorrakrai, K. E. Hamilton, P. B. Parthsarathy, A. Guzman-Saenz, T. Alban, F. Utro, and L. Parida, "Quantum ensembling methods for healthcare and life science", arXiv:2506.02213 [cs.LG] (2025).

[20] K. Nałecz-Charkiewicz and R. M. Nowak, "Algorithm for dna sequence assembly by quantum annealing", BMC Bioinformatics **23**, 10.1186/s12859-022-04661-7 (2022).

[21] A. Sarkar, Z. Al-Ars, and K. Bertels, "QuASeR: Quantum Accelerated de novo DNA sequence reconstruction", PLoS ONE **16**, 1–24 (2021).

[22] A. Sarkar, Z. Al-Ars, C. G. Almudever, and K. Bertels, "An algorithm for dna read alignment on quantum accelerators", arXiv:1909.05563 [quant-ph] (2019).

[23] R. Chen, Q.-X. Mei, W.-D. Zhao, L. Yao, H.-X. Yang, *et al.*, "hqqubo: A hybrid-querying quantum optimization model validated with 16-qubits on an ion trap quantum computer for life science applications", arXiv:2506.01559 [quant-ph] (2025).

[24] C. Outeiral, G. M. Morris, J. Shi, M. Strahm, S. C. Benjamin, and C. M. Deane, "Investigating the potential for a limited quantum speedup on protein lattice problems", New J. Phys. **23**, 103030 (2021).

[25] R. Babbush, A. Perdomo-Ortiz, B. O'Gorman, W. Macready, and A. Aspuru-Guzik, "Construction of energy functions for lattice heteropolymer models: Efficient encodings for constraint satisfaction programming and quantum annealing", Advances in Chemical Physics: Volume 155 , 201–244 (2014).

[26] T. Babej, C. Ing, and M. Fingerhuth, "Coarse-grained lattice protein folding on a quantum annealer", arXiv:1811.00713 (2018).

[27] A. Irbäck, L. Knuthson, S. Mohanty, and C. Peterson, "Folding lattice proteins with quantum annealing", Phys. Rev. Res. **4**, 043013 (2022).

[28] A. Perdomo, C. Truncik, I. Tubert-Brohman, G. Rose, and A. Aspuru-Guzik, "On the construction of model Hamiltonians for adiabatic quantum computation and its application to finding low energy conformations of lattice protein models", Physical Review A **78**, 012320 (2008).

[29] A. Perdomo-Ortiz, N. Dickson, M. Drew-Brook, G. Rose, and A. Aspuru-Guzik, "Finding low-energy conformations of lattice protein models by quantum annealing", Sci. Rep. **2**, 248 (2012).

[30] M. Fingerhuth, T. Babej, and C. Ing, "A quantum alternating operator ansatz with hard and soft constraints for lattice protein folding", arXiv.1810.13411 (2018), arXiv:1810.13411 [quant-ph] .

[31] A. Robert, P. K. Barkoutsos, S. Woerner, and I. Tavernelli, "Resource-efficient quantum algorithm for protein folding", Npj Quantum Inf. **7**, 38 (2021).

[32] H. Doga, B. Raubenolt, F. Cumbo, J. Joshi, F. P. DiFilippo, *et al.*, "A perspective on protein structure prediction using quantum computers", J. Chem. Theory Comput. **20**, 3359-3378 (2024).

[33] P. A. M. Casares, R. Campos, and M. A. Martin-Delgado, "QFold: Quantum Walks and Deep Learning to Solve Protein Folding", Quantum Science and Technology **7**, 025013 (2022).

[34] R. Saito, K. Okuwaki, Y. Mochizuki, R. Nagai, T. Kato, K. Sugisaki, and Y. Minato, "Protein Folding Model Using Quantum Computation", Journal of Computer Chemistry, Japan **21**, 39–42 (2022).

[35] V. K. Mulligan, H. Melo, H. I. Merritt, S. Slocum, B. D. Weitzner, A. M. Watkins, P. D. Renfrew, C. Pelissier, P. S. Arora, and R. Bonneau, "Designing Peptides on a Quantum Computer", bioRxiv:10.1101/752485 (2020).

[36] J. B. Maguire, D. Grattarola, V. K. Mulligan, E. Klyshko, and H. Melo, "XENet: Using a new graph convolution to accelerate the timeline for protein design on quantum computers", PLOS Computational Biology **17**, e1009037 (2021).

[37] A. Agathangelou, D. Manawadu, and I. Tavernelli, "Quantum algorithm for protein side-chain optimisation: Comparing quantum to classical methods", arXiv:2507.19383 [quant-ph] (2025).

[38] S. V. Romero, A. G. Cadavid, P. Nikačević, E. Solano, N. N. Hegade, *et al.*, "Protein folding with an all-to-all trapped-ion quantum computer", arXiv:2506.07866 (2025).

[39] Y. Zhang, Y. Yang, W. Martin, K. Lin, Z. Wang, *et al.*, "Prediction of protein three-dimensional structures via a hardware-executable quantum computing framework", arXiv:2506.22677 [cs.ET] (2025).

[40] R. Wong and W.-L. Chang, "Quantum speedup for protein structure prediction", IEEE Transactions on NanoBioscience **20**, 323–330 (2021).

[41] R. Wong and W.-L. Chang, "Fast quantum algorithm for protein structure prediction in hydrophobic-hydrophilic model", Journal of Parallel and Distributed Computing **164**, 178–190 (2022).

[42] D. S. Goodsell and A. J. Olson, "Structural symmetry and protein function", Annual review of biophysics and biomolecular structure **29**, 105–153 (2000).

[43] C. N. Cavasotto and A. J. W Orry, "Ligand docking and structure-based virtual screening in drug discovery", Current topics in medicinal chemistry **7**, 1006–1014 (2007).

[44] R. Zwanzig, A. Szabo, and B. Bagchi, "Levinthal's paradox", Proceedings of the National Academy of Sciences of the United States of America **89**, 20–22 (1992).

[45] A. Flissi, E. Ricart, C. Campart, M. Chevalier, Y. Dufresne, J. Michalik, *et al.*, "Norine: update of the nonribosomal peptide resource", Nucleic Acids Research **48**, D465–D469 (2020).

[46] M. Rother and J. A. Krzycki, "Selenocysteine, pyrrolysine, and the unique energy metabolism of methanogenic archaea", Archaea **2010**, 1–14 (2010).

[47] L. Brocchieri and S. Karlin, "Protein length in eukaryotic and prokaryotic proteomes", Nucleic Acids Research **33**, 3390–3400 (2005).

[48] Y. Wang, S. Zhang, F. Li, Y. Zhou, Y. Zhang, *et al.*, "Therapeutic target database 2020: enriched resource for facilitating research and early development of targeted therapeutics", Nucleic Acids Research **48**, D1031–D1041 (2020).

[49] R. K. Murray, D. K. Granner, P. A. Mayes, and V. W. Rodwell, *Harper's Illustrated Biochemistry* (McGraw-Hill Medical, New York, 2006).

[50] C. Levinthal, "Are there pathways for protein folding?", Journal de Chimie Physique **65**, 44–45 (1968).

[51] J. N. Onuchic, P. G. Wolynes, Z. Luthey-Schulten, and N. S. Socci, "Toward an outline of the topography of a realistic protein-folding funnel", Proceedings of the National Academy of Sciences of the United States of America **92**, 3626–3630 (1995).

[52] National Heart, Lung, and Blood Institute, "What is sickle cell disease?", accessed: 2025-09-07 (2024).

[53] J. N. Onuchic and P. G. Wolynes, "Theory of protein folding", Current Opinion in Structural Biology **14**, 70–75 (2004).

[54] S. Piana, J. Klepeis, and D. Shaw, "Assessing the accuracy of physical models used in protein-folding simulations: Quantitative evidence from long molecular dynamics simulations", Current opinion in structural biology **24C**, 98–105 (2014).

[55] A. Gupta, A. Singh, N. Ahmad, T. P. Singh, S. Sharma, and P. Sharma, in *Advances in Protein Molecular and Structural Biology Methods*, edited by T. Tripathi and V. K. Dubey (Academic Press, 2022) pp. 181–197.

[56] H. M. Berman, "The protein data bank", Nucleic Acids Research **28**, 235–242 (2000).

[57] B. Kuhlman and P. Bradley, "Advances in protein structure prediction and design", Nature Reviews Molecular Cell Biology **20**, 681–697 (2019).

[58] M. Dorn, M. B. e Silva, L. S. Buriol, and L. C. Lamb, "Three-dimensional protein structure prediction: Methods and computational strategies", Computational Biology and Chemistry **53**, 251–276 (2014).

[59] B. R. Brooks, C. L. Brooks, A. D. Mackerell, L. Nilsson, R. J. Petrella, *et al.*, "CHARMM: The biomolecular simulation program", Journal of Computational Chemistry **30**, 1545–1614 (2009).

[60] A. Liwo, M. Baranowski, C. Czaplewski, E. Gołaś, Y. He, *et al.*, "A unified coarse-grained model of biological macromolecules based on mean-field multipole–multipole interactions", Journal of Molecular Modeling **20**, 2306 (2014).

[61] R. Pearce and Y. Zhang, "Toward the solution of the protein structure prediction problem", Journal of Biological Chemistry **297**, 100870 (2021).

[62] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, *et al.*, "Highly accurate protein structure prediction with AlphaFold", Nature **596**, 583–589 (2021).

[63] J. Moult, "A decade of CASP: Progress, bottlenecks and prognosis in protein structure prediction", Current Opinion in Structural Biology **15**, 285–289 (2005).

[64] M. Baek, F. DiMaio, I. Anishchenko, J. Dauparas, S. Ovchinnikov, *et al.*, "Accurate prediction of protein structures and interactions using a three-track neural network", Science **373**, 871-876 (2021).

[65] S. Ovchinnikov, H. Park, D. E. Kim, F. DiMaio, and D. Baker, "Protein structure prediction using Rosetta in CASP12", Proteins: Structure, Function and Bioinformatics **86**, 113–121 (2018).

[66] R. Das, B. Qian, S. Raman, R. Vernon, J. Thompson, *et al.*, "Structure prediction for CASP7 targets using extensive all-atom refinement with Rosetta@home", Proteins: Structure, Function and Genetics **69**, 118–128 (2007).

[67] S. Raman, R. Vernon, J. Thompson, M. Tyka, R. Sadreyev, J. Pei, and ohers, "Structure prediction for CASP8 with all-atom refinement using Rosetta", Proteins **77 Suppl 9**, 89–99 (2009).

[68] "The Nobel Prize in Chemistry 2024", `https://www.nobelprize.org/prizes/chemistry/2024/summary/` (2024).

[69] V. Scardino, J. I. D. Filippo, and C. N. Cavasotto, "How good are alphafold models for docking-based virtual screening?", iScience **26**, 105920 (2023).

[70] D. Chakravarty, M. Lee, and L. L. Porter, "Proteins with alternative folds reveal blind spots in alphafold-based protein structure prediction", Current Opinion in Structural Biology **90**, 102973 (2025).

[71] C. Outeiral, D. A. Nissley, and C. M. Deane, "Current structure predictors are not learning the physics of protein folding", Bioinformatics **38**, 1881-1887 (2022), https://academic.oup.com/bioinformatics/article-pdf/38/7/1881/49009618/btab881.pdf .

[72] S. Kmiecik, D. Gront, M. Kolinski, L. Wieteska, A. E. Dawid, and A. Kolinski, "Coarse-Grained Protein Models and Their Applications", Chemical Reviews **116**, 7898–7936 (2016).

[73] M. Levitt and A. Warshel, "Computer simulation of protein folding", Nature **253**, 694–698 (1975).

[74] D. A. Hinds and M. Levitt, "A lattice model for protein structure prediction at low resolution.", Proceedings of the National Academy of Sciences of the United States of America **89**, 2536–2540 (1992).

[75] K. F. Lau and K. A. Dill, "A lattice statistical mechanics model of the conformational and sequence spaces of proteins", Macromolecules **22**, 3986–3997 (1989).

[76] S. Moreno-Hernández and M. Levitt, "Comparative modeling and protein-like features of hydrophobic-polar models on a two-dimensional lattice", Proteins 10.1002/prot.24067 (2012).

[77] S. Miyazawa and R. L. Jernigan, "Residue – Residue Potentials with a Favorable Contact Pair Term and an Unfavorable High Packing Density Term, for Simulation and Threading", Journal of Molecular Biology **256**, 623–644 (1996).

[78] A. Kolinski, "Protein modeling and structure prediction with a reduced representation", Acta Biochimica Polonica **51**, 349–371 (2004).

[79] S. Jin, M. Chen, X. Chen, C. Bueno, W. Lu, *et al.*, "Protein Structure Prediction in CASP13 Using AWSEM-Suite", Journal of Chemical Theory and Computation **16**, 3977–3988 (2020).

[80] S. J. Marrink, H. J. Risselada, S. Yefimov, D. P. Tieleman, and A. H. de Vries, "The martini force field: Coarse grained model for biomolecular simulations", The Journal of Physical Chemistry B **111**, 7812–7824 (2007).

[81] F. Sterpone, S. Melchionna, P. Tuffery, S. Pasquali, N. Mousseau, *et al.*, "The opep coarse-grained protein model: from single molecules, amyloid formation, role of macromolecular crowding and hydrodynamics to rna/dna complexes", Chemical Society Reviews **43**, 7378–7398 (2014).

[82] P. Chandarana, N. N. Hegade, I. Montalban, E. Solano, and X. Chen, "Digitized Counterdiabatic Quantum Algorithm for Protein Folding", Phys. Rev. Appl. **20**, 014024 (2023).

[83] S. Boulebnane, X. Lucas, A. Meyder, S. Adaszewski, and A. Montanaro, "Peptide conformational sampling using the Quantum Approximate Optimization Algorithm", Npj Quantum Inf. **9**, 70 (2023).

[84] A. Irbäck, L. Knuthson, S. Mohanty, and C. Peterson, "Using quantum annealing to design lattice proteins", Phys. Rev. Res. **6** (2024).

[85] W. E. Hart and S. Istrail, "Robust Proofs of NP-Hardness for Protein Folding: General Lattices and Energy Potentials", Journal of Computational Biology **4**, 1–22 (1997).

[86] B. Berger and T. Leighton, "Protein folding in the hydrophobic-hydrophilic (hp) model is np-complete", J. Comput. Biol. **5**, 27–40 (1998).

[87] B. H. Park and M. Levitt, "The complexity and accuracy of discrete state models of protein structure", Journal of Molecular Biology **249**, 493–507 (1995).

[88] A. Godzik, A. Kolinski, and J. Skolnick, "Lattice representations of globular proteins: How good are they?", Journal of Computational Chemistry **14**, 1194–1202 (1993).

[89] L. Pauling, R. B. Corey, and H. R. Branson, "The structure of proteins: Two hydrogen-bonded helical configurations of the polypeptide chain", Proc. Natl. Acad. Sci. **37**, 205–211 (1951).

[90] P. Clote, M. Cebrian, I. Dotu, and P. Hentenryck, in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence* (2008) pp. 241–246.

[91] J. Heringa and P. Argos, "Side-chain clusters in protein structures and their role in protein folding", Journal of Molecular Biology **220**, 151-171 (1991).

[92] R. L. J. Dunbrack and M. Karplus, "Backbone-dependent rotamer library for proteins. application to side-chain prediction", Journal of Molecular Biology **230**, 543–574 (1993).

[93] N. A. Pierce and E. Winfree, "Protein design is np-hard", Protein Engineering, Design and Selection **15**, 779–782 (2002).

[94] C. A. Rohl, C. E. Strauss, K. M. Misura, and D. Baker, "Protein Structure Prediction Using Rosetta", Methods in Enzymology **383**, 66–93 (2004).

[95] R. F. Alford, A. Leaver-Fay, J. R. Jeliazkov, M. J. O'Meara, *et al.*, "The Rosetta All-Atom Energy Function for Macromolecular Modeling and Design", Journal of Chemical Theory and Computation **13**, 3031–3048 (2017).

[96] B. Kuhlman, G. Dantas, G. C. Ireton, G. Varani, B. L. Stoddard, and D. Baker, "Design of a novel globular protein fold with atomic-level accuracy", Science **302**, 1364-1368 (2003).

[97] G. Bhardwaj, V. K. Mulligan, C. D. Bahl, J. M. Gilmore, P. J. Harvey, *et al.*, "Accurate *de novo* design of hyperstable constrained peptides", Nature **538**, 329–335 (2016).

[98] K. K. Yang, Z. Wu, and F. H. Arnold, "Machine-learning-guided directed evolution for protein engineering", Nat. Methods **16**, 687–694 (2019).

[99] L. Cao, I. Goreshnik, B. Coventry, J. B. Case, L. Miller, *et al.*, "De novo design of picomolar SARS-CoV-2 miniprotein inhibitors", Science **370**, 426–431 (2020).

[100] M. A. Jendrusch, A. L. J. Yang, E. Cacace, J. Bobonis, C. G. P. Voogdt, *et al.*, "Alphadesign: a de novo protein design framework based on alphafold", Molecular Systems Biology **21**, 1166-1189 (2025).

[101] G. L. Butterfield, M. J. Lajoie, H. H. Gustafson, D. L. Sellers, U. Nattermann, *et al.*, "Evolution of a designed protein assembly encapsulating its own rna genome", Nature **552**, 415–420 (2017).

[102] A. Irbäck, C. Peterson, F. Potthast, and E. Sandelin, "Design of sequences with good folding properties in coarse-grained protein models", Structure **7**, 347–360 (1999).

[103] A. Aina and S. Wallin, "Multisequence algorithm for coarse-grained biomolecular simulations: Exploring the sequence-structure relationship of proteins", J. Chem. Phys. **147**, 095102 (2017).

[104] S. A. Cook, "The complexity of theorem-proving procedures", Proceedings of the Third Annual ACM Symposium on Theory of Computing (STOC) , 151–158 (1971).

[105] L. E. Ballentine, *Quantum Mechanics: A Modern Development*, 2nd ed. (World Scientific Publishing Company, 2014).

[106] D. J. Griffiths and D. F. Schroeter, *Introduction to Quantum Mechanics*, 3rd ed. (Cambridge University Press, 2018).

[107] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information* (Cambridge University Press, 2019).

[108] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, "Trapped-ion quantum computing: Progress and challenges", Applied Physics Reviews **6**, 021314 (2019).

[109] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, "Superconducting Qubits: Current State of Play", Annual Review of Condensed Matter Physics **11**, 369–395 (2020).

[110] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, *et al.*, "Quantum computational advantage using photons", Science **370**, 1460–1463 (2020).

[111] C. Gidney and M. Ekerå, "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits", Quantum **5**, 433 (2021).

[112] L. DiCarlo, J. M. Chow, J. M. Gambetta, L. S. Bishop, B. R. Johnson, *et al.*, "Demonstration of two-qubit algorithms with a superconducting quantum processor", Nature **460**, 240–244 (2009).

[113] S. Debnath, N. M. Linke, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, "Demonstration of a small programmable quantum computer with atomic qubits", Nature **536**, 63–66 (2016).

[114] M. Reagor, C. B. Osborn, N. Tezak, A. Staley, G. Prawiroatmodjo, *et al.*, "Demonstration of universal parametric entangling gates on a multi-qubit lattice", Science Advances **4**, eaao3603 (2018).

[115] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda., "A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem", Science **292**, 472-475 (2001).

[116] T. Kadowaki and H. Nishimori, "Quantum annealing in the transverse Ising model", Phys. Rev. E **58**, 5355–5363 (1998).

[117] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, *et al.*, "Variational quantum algorithms", Nature Reviews Physics **3**, 625–644 (2021).

[118] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Quantum Computation by Adiabatic Evolution", arXiv:quant-ph/0001106 (2000).

[119] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, *et al.*, "Quantum computing with Qiskit", arXiv:2405.08810 [quant-ph] (2024).

[120] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, *et al.*, "Pennylane: Automatic differentiation of hybrid quantum-classical computations", arXiv preprint arXiv:1811.04968 (2018).

[121] G. Q. AI, "Cirq: A python framework for creating, editing, and invoking noisy intermediate scale quantum (nisq) circuits", urlhttps://github.com/quantumlib/Cirq, accessed: 2025-10-01 (2023).

[122] A. W. Services, "Amazon braket examples", urlhttps://github.com/amazon-braket/amazon-braket-examples, accessed: 2025-10-01 (2025).

[123] K. Blekos, D. Brand, A. Ceschini, C.-H. Chou, R.-H. Li, *et al.*, "A review on Quantum Approximate Optimization Algorithm and its variants", Phys. Rep. **1068**, 1–66 (2024).

[124] A. Abbas *et al.*, "Challenges and opportunities in quantum optimization", Nat. Rev. Phys. **6**, 718–735 (2024).

[125] A. Lucas, "Ising formulations of many NP problems", Frontiers in Physics **2** (2014).

[126] S. Hadfield, Z. Wang, B. O'Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, "From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz", Algorithms **12**, 34 (2019).

[127] S. Sim, P. D. Johnson, and A. Aspuru-Guzik, "Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms", Advanced Quantum Technologies **2**, 1900070 (2019).

[128] "RealAmplitudes (latest version)", `https://quantum.cloud.ibm.com/docs/en/api/qiskit/quantum.cloud.ibm.com/docs/en/api/qiskit/qiskit.circuit.library.realamplitudes` (2025).

[129] L. Bittel and M. Kliesch, "Training variational quantum algorithms is np-hard", Phys. Rev. Lett. **127**, 120502 (2021).

[130] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes", Nat. Commun. **9**, 4812 (2018).

[131] M. Cerezo, M. Larocca, D. García-Martín, N. L. Diaz, P. Braccia, *et al.*, "Does provable absence of barren plateaus imply classical simulability? or, why we need to rethink variational quantum computing", arXiv:2312.09121 [quant-ph] (2024).

[132] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, "Connecting ansatz expressibility to gradient magnitudes and barren plateaus", PRX Quantum **3**, 010313 (2022).

[133] C. Ortiz Marrero, M. Kieferová, and N. Wiebe, "Entanglement-induced barren plateaus", PRX Quantum **2**, 040316 (2021).

[134] E. R. Anschuetz and B. T. Kiani, "Quantum variational algorithms are swamped with traps", Nature Communications **13**, 7760 (2022).

[135] 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*, edited by S. Gomez and J.-P. Hennart (Springer Netherlands, Dordrecht, 1994) pp. 51–67.

[136] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices", Phys. Rev. X **10**, 021067 (2020).

[137] S. H. Sack and M. Serbyn, "Quantum annealing initialization of the quantum approximate optimization algorithm", Quantum **5**, 491 (2021).

[138] Q. Langfitt, J. Falla, I. Safro, and Y. Alexeev, in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, Vol. 02 (2023) pp. 300–301.

[139] I. Lyngfelt and L. García-Álvarez, "Symmetry-informed transferability of optimal parameters in the quantum approximate optimization algorithm", Phys. Rev. A **111**, 022418 (2025).

[140] J. A. Montañez-Barrera, D. Willsch, and K. Michielsen, "Transfer learning of optimal qaoa parameters in combinatorial optimization", Quantum Inf. Process. **24**, 129 (2025).

[141] A. Galda, X. Liu, D. Lykov, Y. Alexeev, and I. Safro, "Transferability of optimal qaoa parameters between random graphs", arXiv.2106.07531 (2021), arXiv:2106.07531 [quant-ph] .

[142] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing", science **220**, 671–680 (1983).

[143] P. K. Barkoutsos, G. Nannicini, A. Robert, I. Tavernelli, and S. Woerner, "Improving Variational Quantum Optimization using CVaR", Quantum **4**, 256 (2020).

[144] J. Chiaverini, R. B. Blakestad, J. Britton, J. D. Jost, C. Langer, D. Leibfried, R. Ozeri, and D. J. Wineland, "Surface-electrode architecture for ion-trap quantum information processing", Quantum Information and Computation **5**, 419–439 (2005).

[145] I. Bloch, J. Dalibard, and W. Zwerger, "Many-body physics with ultracold gases", Reviews of Modern Physics **80**, 885–964 (2008).

[146] M. W. Johnson, M. H. S. Amin, S. Gildert, T. Lanting, F. Hamze, *et al.*, "Quantum annealing with manufactured spins", Nature **473**, 194–198 (2011).

[147] P. I. Bunyk, E. M. Hoskinson, M. W. Johnson, E. Tolkacheva, F. Altomare, *et al.*, "Architectural considerations in the design of a superconducting quantum annealing processor", IEEE Transactions on Applied Superconductivity **24**, 1–10 (2014), arXiv:1401.5504 [quant-ph] .

[148] Y. Aharonov, L. Davidovich, and N. Zagury, "Quantum random walks", Physical Review A **48**, 1687 (1993).

[149] J. Kempe, "Quantum random walks: An introductory overview", Contemporary Physics **44**, 307 (2003).

[150] K. Pearson, "The Problem of the Random Walk", Nature **72**, 294 (1905).

[151] F. Spitzer, *Principles of Random Walk*, Graduate Texts in Mathematics, Vol. 34 (Springer New York, New York, NY, 1964).

[152] M. Kac, "Random Walk and the Theory of Brownian Motion", The American Mathematical Monthly **54**, 369 (1947).

[153] R. M. Nosofsky and T. J. Palmeri, "An exemplar-based random walk model of speeded classification", Psychological Review **104**, 266 (1997).

[154] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine", Computer Networks and ISDN Systems **30**, 107 (1998).

[155] L. Grady, "Random Walks for Image Segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence **28**, 1768 (2006).

[156] C. Gkantsidis, M. Mihail, and A. Saberi, "Random walks in peer-to-peer networks: Algorithms and evaluation", Performance Evaluation **63**, 241 (2006).

[157] E. A. Codling, M. J. Plank, and S. Benhamou, "Random walk models in biology", Journal of The Royal Society Interface **5**, 813 (2008).

[158] F. Xia, J. Liu, H. Nie, Y. Fu, L. Wan, and X. Kong, "Random Walks: A Review of Algorithms and Applications", IEEE Transactions on Emerging Topics in Computational Intelligence **4**, 95 (2020).

[159] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman, in *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing* (ACM, New York, NY, USA, 2003) pp. 59–68.

[160] S. Chakraborty, L. Novo, A. Ambainis, and Y. Omar, "Spatial Search by Quantum Walk is Optimal for Almost all Graphs", Physical Review Letters **116**, 100501 (2016).

[161] S. Chakraborty, L. Novo, and J. Roland, "Optimality of spatial search via continuous-time quantum walks", Physical Review A **102**, 032214 (2020).

[162] B. Giese, J. Amaudrut, A.-K. Köhler, M. Spormann, and S. Wessely, "Direct observation of hole transfer through DNA by hopping between adenine bases and by tunnelling", Nature **412**, 318–320 (2001).

[163] G. S. Engel, T. R. Calhoun, E. L. Read, T.-K. Ahn, T. Mančal, *et al.*, "Evidence for wavelike energy transfer through quantum coherence in photosynthetic systems", Nature **446**, 782–786 (2007).

[164] A. M. Childs, "Universal Computation by Quantum Walk", Physical Review Letters **102**, 180501 (2009).

[165] N. B. Lovett, S. Cooper, M. Everitt, M. Trevers, and V. Kendon, "Universal quantum computation using the discrete-time quantum walk", Physical Review A **81**, 042330 (2010).

[166] A. M. Childs, D. Gosset, and Z. Webb, "Universal Computation by Multiparticle Quantum Walk", Science **339**, 791 (2013).

[167] N. Masuda, M. A. Porter, and R. Lambiotte, "Random walks and diffusion on networks", Physics Reports **716-717**, 1–58 (2017).

[168] D. Aldous and J. A. Fill, *Reversible Markov Chains and Random Walks on Graphs* (1999).

[169] A. A. Melnikov, A. P. Alodjants, and L. E. Fedichkin, *Hitting time for quantum walks of identical particles*, March 2019 (2019) p. 55.

[170] A. A. Melnikov, L. E. Fedichkin, and A. Alodjants, "Predicting quantum advantage by quantum walk with convolutional neural networks", New Journal of Physics **21**, 1V (2019).

[171] K. Temme, T. J. Osborne, K. G. Vollbrecht, D. Poulin, and F. Verstraete, "Quantum Metropolis Sampling", Nature **471**, 87–90 (2009).

[172] G. D. Varsamis and I. G. Karafyllidis, "A quantum walks assisted algorithm for peptide and protein folding prediction", Biosystems **223**, 104822 (2023).

[173] N. P. Sawaya, A. T. Schmitz, and S. Hadfield, "Encoding trade-offs and design toolkits in quantum algorithms for discrete optimization: coloring, routing, scheduling, and other problems", Quantum **7**, 1111 (2023).

[174] M. Müller, K. Hammerer, Y. L. Zhou, C. F. Roos, and P. Zoller, "Simulating open quantum systems: from many-body interactions to stabilizer pumping", New Journal of Physics **13**, 085007 (2011).

[175] N. M. Linke, D. Maslov, M. Roetteler, S. Debnath, C. Figgatt, *et al.*, "Experimental comparison of two quantum computing architectures", Proceedings of the National Academy of Sciences **114**, 3305-3310 (2017).

[176] T. Scheiber, M. Heller, and A. Giebel, "Exploring quantum annealing for coarse-grained protein folding", arXiv:2508.10660 [quant-ph] (2025).

[177] J. V. Pamidimukkala, S. Bopardikar, A. Dakshinamoorthy, A. Kannan, K. Dasgupta, and S. Senapati, "Protein Structure Prediction with High Degrees of Freedom in a Gate-Based Quantum Computer", Journal of Chemical Theory and Computation **20**, 10223–10234 (2024), publisher: American Chemical Society.

[178] H. Mustafa, S. N. Morapakula, P. Jain, and S. Ganguly, in *2022 International Conference on Trends in Quantum Computing and Emerging Business Technologies (TQCEBT)* (2022) pp. 1–8.

[179] X. Li, V. R. Kulkarni, J. Nana, S. Pu, N. Xie, Q. Guan, R. Li, S. Zhang, S. Xu, D. Blankenberg, and V. Chaudhary, in *2025 55th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)* (2025) pp. 220–223.

[180] T. Peng, A. W. Harrow, M. Ozols, and X. Wu, "Simulating large quantum circuits on a small quantum computer", Phys. Rev. Lett. **125**, 150504 (2020).

[181] V. Panizza, P. Hauke, C. Micheletti, and P. Faccioli, "Protein design by integrating machine learning and quantum-encoded optimization", PRX Life **2**, 043012 (2024).

[182] M. H. Khatami, U. C. Mendes, N. Wiebe, and P. M. Kim, "Gate-based quantum computing for protein design", PLOS Comput. Biol. **19**, 1-20 (2023).

[183] M. Majewski, A. Pérez, P. Thölke, S. Doerr, N. E. Charron, *et al.*, "Machine learning coarse-grained potentials of protein thermodynamics", Nat. Commun. **14**, 5739 (2023).

[184] D. Shepherd and M. J. Bremner, "Temporally unstructured quantum computation", Proc. R. Soc. Math. Phys. Eng. Sci. **465**, 1413–1439 (2009).

[185] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, "Supervised learning with quantum-enhanced feature spaces", Nature **567**, 209–212 (2019).

[186] T. Ayral, F. L. Régent, Z. Saleem, Y. Alexeev, and M. Suchara, "Quantum divide and compute: Exploring the effect of different noise sources", SN Computer Science **2**, 1–14 (2021).

[187] M. A. Perlin, Z. H. Saleem, M. Suchara, and J. C. Osborn, "Quantum circuit cutting with maximum-likelihood tomography", npj Quantum Information **7**, 1–8 (2021).

[188] S. Wang, P. Wang, G. Li, S. Zhao, D. Zhao, *et al.*, "Variational quantum eigensolver with linear depth problem-inspired ansatz for solving portfolio optimization in finance", Science China Information Sciences **68**, 180504:1–180504:11 (2025).

[189] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, "An adaptive variational algorithm for exact molecular simulations on a quantum computer", Nature Communications **10**, 3007 (2019), arXiv:1812.11173 [quant-ph] .

[190] M. Bilkis, M. Cerezo, G. Verdon, P. J. Coles, and L. Cincio, "A semi-agnostic ansatz with variable structure for variational quantum algorithms", Quantum Machine Intelligence **5**, 43 (2023).

[191] F. Meier and H. Yamasaki, "Energy-consumption advantage of quantum computation", PRX Energy **4**, 023008 (2025).

A Appendix

A.1 Quadratic unconstrained binary optimization (QUBO) into the Ising model

Starting from the QUBO objective in Eq. 3.26 with $\vec{x} = b_0 \dots b_{M-1}$, a binary vector with $b_i \in \{0, 1\}$, and $Q \in \mathbb{R}^{M \times M}$, a symmetric matrix of coefficients,

$$\min_{\vec{x}} \sum_{0 \leq i \leq j \leq M-1} Q_{ij}\, b_i b_j, \tag{A.1}$$

we map binary variables to spin variables $\sigma_i^z = -1 \mapsto b_i = 1$ and $\sigma_i^z = +1 \mapsto b_i = 0$ using

$$b_i \;=\; \frac{1 - \sigma_i^z}{2}. \tag{A.2}$$

This implies, for any pair (i, j),

$$b_i b_j \;=\; \left(\frac{1 - \sigma_i^z}{2}\right)\left(\frac{1 - \sigma_j^z}{2}\right) \;=\; \frac{1}{4}\left(1 - \sigma_i^z - \sigma_j^z + \sigma_i^z \sigma_j^z\right). \tag{A.3}$$

Substituting Eq. A.3 into the QUBO sum yields an Ising-form Hamiltonian:

$$H_{\text{Ising}} \;=\; \sum_{0 \leq i \leq j \leq M-1} Q_{ij}\, \frac{1}{4}\left(1 - \sigma_i^z - \sigma_j^z + \sigma_i^z \sigma_j^z\right). \tag{A.4}$$

Collecting constants, single-spin (linear) terms, and two-spin (quadratic) couplings, we can write

$$H_{\text{Ising}} \;=\; c_0 \;+\; \sum_{i=0}^{M-1} h_i\, \sigma_i^z \;+\; \sum_{0 \leq i < j \leq M-1} J_{ij}\, \sigma_i^z \sigma_j^z, \tag{A.5}$$

with coefficients

$$c_0 \;=\; \frac{1}{4} \sum_{0 \leq i < j \leq M-1} Q_{ij} \;+\; \frac{1}{2} \sum_{i=0}^{M-1} Q_{ii}, \tag{A.6}$$

$$h_i \;=\; -\frac{1}{2}\,Q_{ii} \;-\; \frac{1}{4}\sum_{\substack{j=0\\ j\neq i}}^{M-1} Q_{ij}, \tag{A.7}$$

$$J_{ij} \;=\; \frac{1}{4}\,Q_{ij}, \qquad 0 \le i < j \le M-1. \tag{A.8}$$

Notes:

- The constant shift c_0 does not affect the minimizer, but it *does* matter for absolute energies and for comparing spectra across models.
- The linear terms h_i correspond to *one-body* interactions, and the quadratic terms J_{ij} correspond to *two-body* interactions, as illustrated in Fig. 4.2.