Master's thesis
Applied Mathematics & Applied Physics

Solving Correlation Clustering with the Quantum Approximate Optimisation Algorithm

J.R.Weggemans
2020

QuSoft

Supervisors
prof. dr. C.J.M. Schoutens
dr. F. Speelman

UNIVERSITY OF TWENTE.

Supervisors
prof. dr. A. Brinkman
prof. dr. R.J. Boucherie

Members
prof. dr. M.J. Uetz
dr. J.J. Renema

Abstract

The quantum approximation optimisation algorithm (QAOA) is one of the leading candidates for testing the applicability of gate-model quantum resources at solving optimisation problems on small-sized quantum hardware. A combinatorial problem that has been understudied with QAOA is correlation clustering: given a weighted (+1,-1) graph, where the edge weights indicate whether two nodes are similar (positive edge weight) or different (negative edge weight), the task in correlation clustering is to find a clustering that either maximises agreements, minimises disagreements or a combination of both. In this thesis, we design Hamiltonian formulations that encode correlation clustering problems such that they can be solved with QAOA. For all Hamiltonian formulations we propose circuit implementations and study their complexities. To benchmark the performances of a basic QAOA algorithm using these formulations, we use numerical simulations on complete graph data sets. For one of the formulations, which uses a multi-level approach naturally suitable to qudit systems, we investigate the performances of several optimisers and introduce heuristic strategies to further improve its performance. On all instances in our data-sets, which include complete as well as Erdős–Rényi graphs, the improved algorithm shows competitive performances for QAOA depth $p \geq 2$. We also show that for this algorithm at $p = 1$ parameters exists such that it has a performance guarantee of 0.670 on 3-regular graphs.

Acknowledgements

In the search for a suitable thesis topic that would combine both of my master specialisations, Alexander Brinkman suggested taking a look at QuSoft. After some not-so-promising initial mailing conversations, I was happily surprised when Kareljan Schoutens invited me over to discuss a possible graduation project. I just happened to be back in the Netherlands somewhat early due to unfortunate family circumstances, and hence was by chance able to accept his invitation and visit in person (some other times indeed).

I am grateful for the opportunity that was given to me by Kareljan Schoutens. I also owe a lot of gratitude to Richard Boucherie and Alexander Brinkman, my supervisors for Applied Mathematics and Applied Physics, who gave me the opportunity to do something off the beaten track. Next, I would like to thank Florian Speelman for his guidance in the form of our weekly discussions, as well as providing the foundations which this thesis was build upon. Marc Uetz and Jelmer Renema, I thank you for being part of the graduation committee. I would also like to acknowledge Jiri Minar, with whom (as well as some of his experimentalist colleagues) we will develop a full stack quantum computing story based on the algorithmic results from this work.

My gratitude also goes to Bosch Research, and in particular Alexander Rausch with whom I met on a nearly biweekly basis, for supporting this project. It was very inspiring to do fundamental research whilst simultaneously keeping real-world applications in mind. I would also like to thank SURFSara for allowing me access to their LISA system, which was used to obtain the majority of the numerical results in this work.

Furthermore, I would like to thank everyone in the QuSoft group for the time spend at the institute (when possible), but more profoundly, the QuTea-times. I am looking forward to continue being a member, but now as a PhD student.

Finally, I would like to thank my family and friends, for all your support over the past year and before.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Over the past decades the world has undergone drastic changes as it entered a new technological era, generally referred to as the *information age*. Tasks, previously done by humans, are automatised by the use of machines and information has never been as accessible as it is now. And this process has been growing in its capabilities ever since its first invention: computer manufacturers have so far been able to exponentially increase the amount of transistors—the fundamental building block of computers—that are put in computer circuits.

But this process is now reaching its physical limit: as the length scale that the transistors operate on becomes smaller and smaller, quantum mechanical effects start to dominate the physics of the system. In particular, a specific quantum effect called *quantum tunnelling* causes source-to-drain leakage, destroying the functionality of the transistor. A lot of research has been performed in looking for workarounds this problem, but one might also ask: can we actually use these quantum effects to our advantage instead?

This started an entire new field of *quantum technologies*, where quantum effects are exploited for practical applications. Examples are the design of ultra-sensitive quantum sensors, quantum encryption that is unbreakable, and quantum computers to perform quantum computations (generally referred to as the field of *quantum computing*). When Peter Shor in 1994 showed that a quantum algorithm run on such a quantum computer could solve the prime factorisation problem exponentially faster than any classical algorithm, he was the first to show that quantum computers could be fundamentally more powerful than classical computers on certain problems [1]. More algorithms have been designed ever since, showing potential quantum speed-ups in solving linear systems of equations [2], unstructured search [3], simulation of quantum systems [4] and more.

However, running most of these algorithms requires a large fault-tolerant quantum computer: this computer would have a large amount of working qubits, gate operations with low error probabilities and even more qubits to allow for an error correction scheme. This could be decades away or, taking a pessimist view, even be a technological challenge that is simply too difficult to overcome. At the time of writing the largest circuit-based quantum processor is Google's Bristlecone, which has 72 qubits [5]. But as hardware research continues to result in larger and better hardware, we want to know whether there are specific applications and algorithms that *are* viable for these *Noisy Intermediate-Scale Quantum* (NISQ) devices.

One of the proposed algorithms that might be suitable for these devices is the *quantum approximate optimisation algorithm*, abbreviated as *QAOA*, proposed by Farhi, Goldstone and Gutmann in 2014 [6]. QAOA is hybrid algorithm that uses a parametrised quantum processor in conjunction with a classical processor used to tune the parameters that describe the quantum system. Due to its heuristic nature and the curse of dimensionality when the depth of the circuit increases, it is very difficult to prove performance guarantees on specific problems (for some this has been achieved though), which increases the reliance on numerical studies to investigate its potential. Most studies focus on specific, easy-to-analyse computational problems, and in particular the MAXCUT problem. However, the quantum speed-ups are not generic, and for industrial computational applications problems might not be as clearly defined as those in theoretical computer science. Therefore, more and more QAOA research focuses on more applied problems that can be found in for example biology [7, 8], physics [9, 10], computer science [11, 12, 13, 14, 15, 16, 17, 18, 19] and finance [20].

A problem that is both fundamental and has industrial applications in modelling social networks, logistics, machine learning and more, is the correlation clustering problem: clustering is the problem of partitioning data points into groups based on their similarity, and in correlation clustering this is done without specifying the number of clusters in advance. The two most common objectives are either minimising the disagreements or maximising the agreements between the input estimates and the output clustering. For both objectives the decision versions of the corresponding optimisation problem is known to be NP-complete [21]. However, both objectives differ in the difficulty of their approximabilities. The best classical algorithm for maximising the agreements has an approximation ratio of 0.7666 [22], and will be frequently used as a benchmark for the results in this work. This thesis will look at the potential of using of QAOA-based algorithms in approximating correlation clustering problems.

1.1 Research questions and objectives

The goal of this research can perhaps be better framed into an objective than a question: we want to try to create the best possible performance for some QAOA-based algorithm in solving correlation clustering problems, whilst keeping actual hardware considerations in mind. Framing this into a research question to help us work towards this objective, we define our main research question to be:

Main question:

- What is the best (empirical) performance (computation time and approximation ratio) we can obtain using some form of QAOA in solving correlation clustering?

It is important to note that the word ‘can’ in this question has the meaning of ‘the best we have achieved with our efforts so far’ instead of ‘the best that can be fundamentally achieved’. This is also reflected in the following sub-questions we consider, helping us in answering the main research question:

Sub-questions:

- What is the impact of different aspects of the algorithm (e.g. the initial state, the choice of optimiser and the Hamiltonian formulations?)
- What heuristics can we add to improve our algorithms performance?
- Which properties of the correlation clustering problem determines the expected performance of our algorithm? Are there differences in performance on different variants of correlation clustering? And what about different formulations within these variants?
- How does our algorithm compare against state-of-the-art classical solvers?

1.2 Structure of this thesis

This thesis is structured into three different parts consisting of a total of 8 chapters. In Part I we will focus on introducing background information such that readers with little to no background in quantum computing should be able to follow most of the work performed.[1] We will give a brief introduction to quantum computing in Chapter 2 and look at the correlation clustering problem in Chapter 3. In Part II one can find a survey of the quantum approximate optimisation algorithm. Whilst this does not fit into our previously established research questions and objectives, we felt that a survey is currently missing in QAOA literature, and this part provides building blocks to create one. Chapter 4 will be concerned with the fundamentals of QAOA and Chapter 5 will look into more recent results in literature. In part III, we will try to address our main research question. In Chapter 6 we will propose different formalisms to solve the correlation clustering problem and benchmark their performances. We decide upon a formalism to improve upon, and Chapter 7 deals with all the steps we took in order to achieve substantial improvements. In Chapter 8 we summarise our conclusions and propose future work, concluding the main body of this thesis.

[1] Basic knowledge of physics, mathematics (in particular linear algebra) and computer science is assumed though.

PART I

Introduction to key concepts

CHAPTER 2

A primer on quantum computing

Since readers of this thesis will have backgrounds ranging from physics to mathematics to quantum computing, or any combination of those, this chapter will introduce the basic concepts of quantum computing. Throughout the chapter, most material that has been used has been taken from the standard text-book of the field, written by Nielsen and Chang [23]. We assume the reader is familiar with basic concepts in linear algebra, and in particular Hilbert spaces.

2.1 Fundamentals of quantum mechanics: the postulates

The discovery of modern *quantum theory* in the 1920s brought about one of the greatest revolutions in our thinking about nature since the days of Isaac Newton. By unifying the wave and particle interpretations of light and matter, scientists were finally able to find explanations to problems as the photoelectric effect, Compton scattering and black-body radiation. Quantum theory is founded on a set of *postulates*, resulting from experiments and theoretical analysis. The postulates describe how microscopic particles must be represented, how to obtain quantities that can be observed, how time evolution must be described and what the logical structure of a measurement is. The postulates are as follows:

Postulate 1: State space
Any isolated physical system has an associated Hilbert space known as the *state space* of the system. The *state vector*, which is a unit vector in the system's state space, uniquely describes this system.

The state space of a *specific* system is not given by quantum mechanics, and it can be a difficult problem to figure it out. The simplest example, which we will discuss in more detail later, is the *qubit*, which has a two-dimensional state-space.

Postulate 2: evolution
The evolution of a *closed* quantum system is described by a *unitary* transformation:

$$|\psi'\rangle = U\,|\psi\rangle\,.$$

Just as we saw for the state space postulate, quantum mechanics itself does not tell us which unitary operators U describe the evolution of a particular real-world quantum system. If we consider continuous time evolution, we can define a more refined version of the postulate, familiar to all physicists:

$$i\hbar\frac{d}{dt}\left|\psi(t)\right\rangle = H(t)\left|\psi(t)\right\rangle, \tag{2.1}$$

where H is the energy operator, the so-called *Hamiltonian* of the closed system, and $\hbar$ a physical constant known as Planck's constant. Once you know the Hamiltonian of a closed system, you essentially fully understand its dynamics. However, determining a Hamiltonian of a given system is very difficult problem—in fact a large share of twentieth century physics was dedicated to figuring them out.

Postulate 3: Quantum measurement

Associated to any measurement of a physical system with corresponding state space $\mathcal{H}$ is a set of operators $\{M_m\}_{m\in I}$ acting on $\mathcal{H}$ which satisfies (completion relation):

$$\sum_m M_m^\dagger M_m = I,$$

where the index m refers to the measurement outcomes that may occur in the experiment. If the system is in state $|\psi\rangle \in \mathcal{H}$, the probability that one measures the outcome m is

$$P(m) = \langle\psi| M_m^\dagger M_m \left|\psi\right\rangle.$$

If prior to the measurement, the physical system was in state $|\psi\rangle \in \mathcal{H}$ and the measurement outcome was m, the resulting state of the system, directly after the measurement, is given by

$$\frac{M_m\left|\psi\right\rangle}{\sqrt{\langle\psi| M_m^\dagger M_m \left|\psi\right\rangle}}.$$

The third postulate tells us two very important - and perhaps mind-boggling - results of quantum mechanics: 1) measurement of a quantum system interferes with the system's state, and causes the system to collapse into one of the eigenstates of the observable and 2) there is no quantum measurement capable of distinguishing between non-orthogonal quantum states.

Postulate 4: Composite systems

The state space of a composite physical system is the tensor product of the state spaces of the component physical systems. Furthermore, if we have n systems labelled $1,\ldots,n$, and system i is prepared in the state $|\psi_i\rangle$, the joint state of the total system is $|\psi_1\rangle \otimes |\psi_2\rangle \otimes \cdots \otimes |\psi_n\rangle$.

If we cannot write the state of the composite system as a simple tensor, we say that its (isolated) physical systems are *entangled*.

One of the best examples to get more comfortable with these postulates is the so-called *particle in a box*-problem, which can be found in any standard textbook on quantum mechanics. Now that we have established the fundamental rules of quantum mechanics, let us see how these allow us to create a formalism to perform computations.

2.2 Qubits and qudits

In classical computation, the fundamental unit of information is a *bit*, which can take a value of 0 or 1. Hence, we have two possible computational states per a single bit. In quantum computation, the fundamental unit of information is a called a *qubit* (shorthand for quantum bit). Taking $\{|0\rangle, |1\rangle\}$ as our computational basis vectors, this qubit can be in $|0\rangle$ and $|1\rangle$, similarly to the classical bit, but it can also be any other state that satisfies

$$\alpha |0\rangle + \beta |1\rangle \quad \text{where} \quad |\alpha|^2 + |\beta|^2 = 1. \tag{2.2}$$

Any of these states for which both α and β are non-zero are in a so-called *superposition*, a type of state which knows no classical counterpart. Physically, any quantum mechanical system that can be modelled by a two-dimensional complex vector space can be viewed as a qubit. Real-world examples of such systems are the polarisation of a photon, the orientation of an electron spin and the ground state combined with some excited state of an atom. A common way to pictorially view qubits is through the use of the Bloch sphere, as depicted in Figure 2.1:

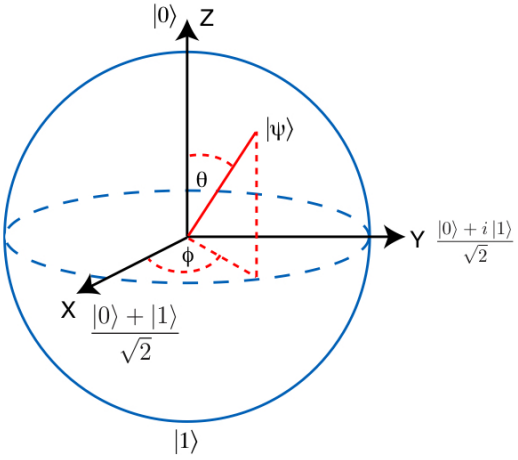


Figure 2.1: Bloch sphere representation of a single qubit. The state vectors aligning with the x- and y-axis have different relative phases, but are impossible to distinguish from one another through measurement in computational basis states $\{|0\rangle, |1\rangle\}$ since they share the same probability distribution over these states.

From postulate 4 we know that a general state of n qubits is written as a tensor product of single qubits, i.e.

$$|\psi\rangle = \bigotimes_{i=1}^{n} (a_i |0\rangle + b_i |1\rangle) = \sum_{j=0}^{2^n-1} |\alpha_j|^2 |j\rangle , \tag{2.3}$$

where $\sum_{j=0}^{2^n-1} |\alpha_j|^2 = 1$. Here we represented our n-qubit system on our new basis states $|j\rangle$. Note that a system of n qubits lives in $\mathbb{C}^{2^n}$, and this exponential growth of the Hilbert space with n explains the difficulty of classically simulating quantum mechanical processes: one needs an exponential number of bits to represent the n qubits.

It is important to stress the difference between superposition and entanglement: we say that an n-qubit system is in superposition if it is not in one of the computational basis states, and we say that an n-qubit state is entangled when it cannot be written as a simple tensor. This means that all entangled states must necessarily be in a superposition, but this does not hold in the other direction. For example, take the two-qubit state

$$\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle),$$

which is in a superposition but can also be written as a tensor product of $|+\rangle \otimes |-\rangle$, and therefore is not an entangled state. Here $\{|+\rangle, |-\rangle\}$ form a different set of computational basis vectors, which can be written as functions of $|0\rangle$ and $|1\rangle$ as $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle) + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle) - |1\rangle)$, respectively. We now take another state as our example, defined as

$$\frac{1}{2}(|00\rangle + |11\rangle). \tag{2.4}$$

By defining two arbitrary single-qubit states $|\psi_A\rangle = \alpha_A |0\rangle + \beta_A |1\rangle$ and $|\psi_B\rangle = \alpha_B |0\rangle + \beta_B |1\rangle$, we observe that no solutions of $\alpha_A, \beta_A, \alpha_B, \beta_B$ satisfying (2.2) exist such that the tensor product

$$(\alpha_A |0\rangle + \beta_A |1\rangle) \otimes (\alpha_B |0\rangle + \beta_B |1\rangle) = \alpha_A\alpha_B |00\rangle + \alpha_A\beta_B |01\rangle + \beta_A\alpha_B |10\rangle + \beta_B\beta_B |11\rangle$$

is equal to (2.4).

So far we only concerned ourselves with two-level systems that were used to encode information in the form of qubits, but it is possible to extend this idea to systems with any amount of levels. A system with three computational basis states $(|0\rangle, |1\rangle, |2\rangle)$ is called a *qutrit*, and any d-level system $(|0\rangle, |1\rangle, \ldots, |d-1\rangle)$ can be used to form a *qudit*. Examples of these are photonic systems that can be in a superposition of multiple possible wavelengths [24] or neutral atoms with a larger range of intrinsic spin states [25]. Every d-level qudit can be represented by $\lceil \log_2 d \rceil$ qubits. Hence, there is no formal argument of why you would have an advantage of using qudits over qubits, as one can always be mathematically represented in the other. However, on the hardware level, the interactions describing the system might be more suitable to either qubit or qudit operations, which means that you sometimes gain something (e.g. smaller errors, larger accessible Hilbert space) from using qudits instead of qubits.

2.3 Quantum gates

Quantum gates provide us with the basic operations needed to manipulate qubits—just as in classical complexity theory a *Boolean circuit* uses basic logic gates on its bits. Well-known examples are the OR, AND, and NOT gates. By postulate 2, we know that every quantum gate must perform a unitary transformation. Consequently, every quantum gate operation is *reversible*. Quantum gates can be represented by matrices, and we refer to this matrix as the *matrix representation* of a quantum gate.

One of the simplest quantum gate example is the X-gate, which is the quantum analogue of the classical NOT-gate. Its matrix representation is

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

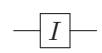
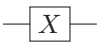
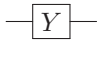
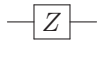
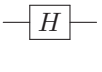
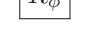
Identity :	$\|0\rangle\langle 0\| + \|1\rangle\langle 1\|$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	
Pauli-X $-$ gate :	$\|1\rangle\langle 0\| + \|0\rangle\langle 1\|$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	
Pauli-Y $-$ gate :	$i\|1\rangle\langle 0\| - i\|0\rangle\langle 1\|$	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	
Pauli-Z $-$ gate :	$\|0\rangle\langle 0\| - \|1\rangle\langle 1\|$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	
Hadamard gate :	$\frac{\|0\rangle + \|1\rangle}{\sqrt{2}}\langle 0\| + \frac{\|0\rangle - \|1\rangle}{\sqrt{2}}\langle 1\|$	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	
R_ϕ-gate :	$\|0\rangle + e^{2\pi i\phi}\|1\rangle$	$\begin{bmatrix} 1 & 0 \\ 1 & e^{2\pi i\phi} \end{bmatrix}$	

Table 2.1: Some common single-qubit quantum gates.

We define our computation basis states as $|0\rangle$ and $|1\rangle$ in the following vector representation

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

such that X operates on these computational basis states in the following way:

$$X|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle \quad \text{and} \quad X|1\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle.$$

Hence, X essentially performs a bit flipping operation, just as a NOT-gate would do classically. X is part of a larger group of single-qubit gate operations, called the *Pauli matrices*. An overview of all Pauli matrices, as well as some other common examples of single-qubit gates are shown in Table 2.1. The last column of this table indicates the circuit representation, which will be explained in more detail later. From the Pauli quantum transformations we can also construct the so-called *rotation gates*:

$$R_X(\theta) = e^{-i\theta X/2}, \qquad R_Y(\theta) = e^{-i\theta Y/2}, \qquad R_Z(\theta) = e^{-i\theta Z/2},$$

which will be commonly used throughout this thesis. Every single-qubit unitary transformation U can be written as $U = R_X(\alpha)R_Y(\beta)R_Z(\gamma)$, for some α, β and γ.

One of the most important gates in Table 2.1 is the Hadamard gate H. Applying H to initial state $|0\rangle$ results in a new state with equal probabilities of measuring $|0\rangle$ or $|1\rangle$. Applying H on this new state gives us back our initial state $|0\rangle$. This effect is called *interference* due to the fact that the amplitudes of the $|1\rangle$ state have cancelled out, similar to the effect one would observe in classical wave mechanics.

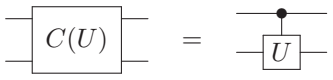
So far we have only considered single-qubit gates, but the notion of a gate operation can be generalised to any amount of qubits. Every n-qubit quantum gate can be represented by a $2^n \times 2^n$ unitary matrix (with complex entries), and every $2^n \times 2^n$ unitary matrix can in theory be a quantum gate. We will now discuss some of the most important

multiple-qubit quantum gates.

Let us first consider the so-called *controlled gates*. These gates act on two or more qubits, where one or more qubits act as a control for some operation. We define some general single-qubit operation U with matrix elements $\{u_{0,0}, u_{0,1}, u_{1,0}, u_{1,1}\}$, then its controlled operation $C(U)$ where qubit 1 acts as the control qubit and qubit 2 acts as the target qubit is given by

$$C(U) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{0,0} & u_{0,1} \\ 0 & 0 & u_{1,0} & u_{1,1} \end{bmatrix}.$$

In quantum circuit notation, we use the following graphical depiction



where the 'large dot' indicates the control qubit. Common examples of controlled two-qubit operations are the CNOT operation (CX), as well as other controlled Pauli operations and their controlled rotations (for example CR_X). It is also possible to swap the target and control qubit or extend this idea to multiple control qubits. A common example of the latter is the Toffoli gate (CCNOT), represented by the following matrix

$$\text{CCNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

The circuit symbols of both the CNOT and Toffoli (CCNOT) are by convention drawn as

2.4 Quantum circuits

A *Quantum circuit* is a model describing the 'recipe' for a quantum computation. The graphical depiction of quantum circuit elements is described using a variant of the Penrose graphical notation, and includes initialised qubits (often in the $|0\rangle$-state), a sequence of quantum gates and measurement of the qubits. Some conventions are:

- In a quantum circuit diagram, moving from left to right corresponds to moving forwards in time.
- Each qubit is represented by a wire and has an initial state (often $|0\rangle$).
- Quantum gate operations are denoted by symbols in a box, which spans over the wires of the qubits it operates on.

- The final state always has to be measured and is therefore often left out of the notation, unless only specific qubits have to be measured or some measurements have to be performed along the computation.

Let us consider a simple example of a 2-qubit quantum circuit, of which the circuit description is given in Figure 2.2:

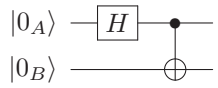


Figure 2.2: Quantum circuit to create an entangled state.

This circuit consists of two qubits, labelled with subscripts 'A' and 'B', that are both initialized in state $|0\rangle$ and uses two gate operations: a Hadamard transform applied only to qubit A and a CNOT applied to control qubit A and target qubit B. The state of the system can be represented by a vector describing the amplitudes of each computational two-qubit basis state in $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, where the first entry indicates the state of qubit A and the second of qubit B. Just as it is possible to find a matrix representation of a quantum gate, there is a matrix representation for each quantum circuit. The matrix representation of this circuit is given by

$$[CNOT]_{i,j} \cdot [H \otimes I]_{i,j} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 \end{bmatrix}$$

So what happens when we apply this circuit operation to our initial state $|00\rangle$, ?

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle),$$

This state, already encountered in section 2.2, is one of the *Bell states*: a maximally entangled two-qubit state.

Some important complexity measures of quantum circuits are the *elementary gate complexity* and *query complexity*. The *elementary gate complexity* of a quantum circuit is defined as the number of elementary gates it consists of. We are free to choose any set of gates we define to be elementary to our definition of gate complexity, keeping in mind that some are more convenient than others. Some common ones are:

- The set of all single-qubit operations and the two-qubit CNOT gate. This set is universal, meaning that any other unitary operation can be built from these gates.

- The set of CNOT, Hadamard and the phase-gate $T = R_{\pi/4}$, which is universal in the sense of approximation. The Solovay-Kitaev theorem states that this approximation is in fact quite efficient: simulating arbitrary gates up to an exponentially small error costs only a polynomial overhead.

- The set of Hadamard and Toffoli (CCNOT) is universal for all unitaries with real entries in the sense of approximation.

In the *query complexity* model, the input is given as an oracle (a black box function). The algorithm gets information about the input only by *querying* this oracle ('calls' the black box). The algorithm starts in some fixed quantum state and the state evolves as it queries the oracle. The query complexity is then defined as the number of queries the algorithm makes to the oracle. Therefore, query complexity provides a lower bound on the overall time complexity of an algorithm as it only takes the oracle into account.

For hardware considerations, one also very often talks about the depth and width of a quantum circuit. The *circuit depth* is the length of the longest path from the input (or from a preparation) to the output (or a measurement gate), moving forward in time along qubit wires. This takes into account that, in actual quantum hardware, some operations can be performed parallel. The *circuit width* is the number of qubits (and bits) used in the quantum circuit (the number of wires in the diagram).

2.5 Quantum algorithms and their complexity

An *algorithm* solves a given class of problems using a finite sequence of well-defined (computer-implementable) instructions. An algorithm in which all of these steps can be executed on a universal *Turing machine* will be referred to as a *classical algorithm*, whilst an algorithm that requires at least some inputs to be operated on by a quantum circuit to be a *quantum algorithm*. The part of the algorithm that can be implemented on a universal Turing machine is referred to as the classical part of the quantum algorithm, which is also something we will encounter when dealing with the Quantum Approximate Optimisation Algorithm this thesis is concerned with.

Arguably, the greatest success of quantum computing to date is that research over the past decades has shown that quantum algorithms exist that provide a speed-up over the best classical algorithms. The first example of this was provided in the form of Shor's factoring algorithm, which provides a *super-polynomial* speed-up in finding the prime factorisation of an n-bit integer. To be precise: this means that for the prime factorisation problem, the quantum algorithm can find solutions with time and space requirements that are bounded by a polynomial in the size of the input, while it is conjectured that a classical computer would need an exponential amount of resources for the same task.

(Quantum) computational complexity theory provides a general framework to quantify the resources algorithms need for the problems they attempt to solve. The class of problems that are in polynomial time on a quantum computer is called *Bounded-error Quantum polynomial* (BQP), analogous to the classical *Polynomial* (P) complexity class. It is generally conjectured that P $\subsetneq$ BQP, which implies that there exist problems

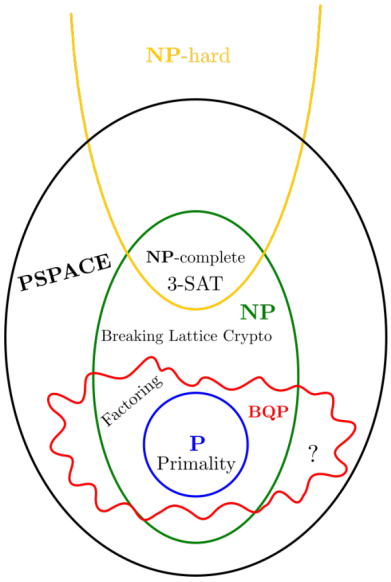


Figure 2.3: Conjectured relations between different complexity classes, including some example problems within certain classes. Picture taken from Ref. [27].

that are in BQP but not in P (as we already saw for prime factorisation).[1] If an algorithm can verify whether the problem is solved if given both the input and the output, and this algorithm takes polynomial time on a classical computer, the problem is *Non-deterministic Polynomial* (NP). The quantum analogue for these problems is called *Quantum Merlin Arthur* (QMA). Since solving a problem in polynomial time necessarily requires you to be able to verify an output in polynomial time, we have that P$\subseteq$NP and BQP $\subseteq$ QMA. We also have that any classical computation can be implemented on a quantum computer, and therefore and NP $\subseteq$ QMA. In addition, the class *NP-hard* (*QMA-hard* for quantum) includes all problems to which all problems in NP can be reduced to in polynomial time: this means that these problems are as 'difficult' as any problem in NP. If a problem is both in NP and NP-hard, we call it *NP-complete (QMA-complete* for quantum).

So BQP does not seem to be in P, and it is also unknown whether it is in NP. So where does it actually fit in relation to other complexity classes? Bernstein and Vazirani showed that it is possible to simulate a quantum computer classically with exponential time and polynomial memory, which means that the following upper bound exists [26]: BQP $\subseteq$ PSPACE, where *PSPACE* is the class solvable on a digital computer using a polynomial amount of memory, but possibly exponential time. Figure 2.3 shows the current best guess of where BQP fits in. An overview of quantum algorithms and the speed-up they provide over certain problem can be found on online, see *https://quantumalgorithmzoo.org/*.

[1] In fact this statement is also based on a conjecture: neither the existence nor non-existence of such algorithms for prime factorisation has been proved, but it is generally suspected that no classical algorithms exist that are able to solve prime factorisation in polynomial time.

2.6 An example: Grover's algorithm

With all established so far, we now look at a non-trivial example of a quantum algorithm that provides a quantum speed-up. Consider the following problem:

The unstructured search problem:
For $N = 2^n$, we are given an arbitrary $x \in \{0,1\}^N$. The goal is to find i such that $x_i = 1$ and to output 'no solutions' if there are no such i.

We assume that we have access to a (quantum) oracle: a black box function that is able to recognise solutions to the search problem. To be precise, we define this oracle in the quantum setting to be

$$O_{x,\pm} |i\rangle = (-1)^{x_i} |i\rangle , \tag{2.5}$$

which means that the oracle *marks* the solutions to the search problem by shifting the phase of the solution. We also define R as a unitary operation that puts a '-1' in front of all basis states $|i\rangle$ where $i \neq 0^n$ and does nothing to the other basis states:

$$R = 2 |0^n\rangle \langle 0^n| - I \tag{2.6}$$

We define the *Grover iteration* (or *Grover operator*) $\mathcal{G}$ as[2]

$$\mathcal{G} = H^{\otimes n} R H^{\otimes n} O_{x,\pm}. \tag{2.7}$$

Let us define so-called 'good' states $|G\rangle$ and 'bad' states $|B\rangle$ as

$$|G\rangle = \frac{1}{\sqrt{t}} \sum_{i:x_i=1} |i\rangle \quad \text{and} \quad |B\rangle = \frac{1}{\sqrt{N}-1} \sum_{i:x_i=0} |i\rangle .$$

When inspecting the Grover iterate $\mathcal{G}$ one observes that it is actually the product of two *reflections* in the 2-dimensional space spanned by $|G\rangle$ and $|B\rangle$: $O_{x,\pm}$ is a reflection through $|B\rangle$ and

$$H^{\otimes n} R H^{\otimes n} = H^{\otimes n}(2 |0^n\rangle \langle 0^n| - I) H^{\otimes n} = 2 |U\rangle \langle U| - I \tag{2.8}$$

is a reflection through $|U\rangle$. The circuit design for a single Grover iterate is given by,

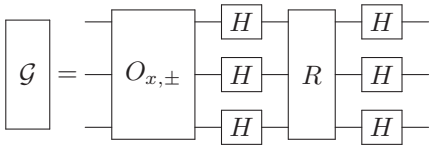


Figure 2.4: Quantum circuit for a single Grover iteration $\mathcal{G}$.

where R can be implemented using $O(n)$ elementary gates.

Assuming that we know that the fraction of solutions is $\epsilon = t/N$, Grover's algorithm is then given by

[2] To simulate this operator classically, one would of course need an exponentially growing amount of resources.

1. Set up the starting state $|U\rangle = H^{\otimes n}|0\rangle$.
2. Repeat the following $k = O(1/\sqrt{\epsilon})$ times:
   a) Reflect through $|B\rangle$ (apply $O_{x,pm}$).
   b) Reflect through $|U\rangle$ (apply $H^{\otimes n} R H^{\otimes n}$).
3. Perform the measurement and check that the resulting i is a solution.

The quantum circuit representation of the complete algorithm is given in Figure 2.5:

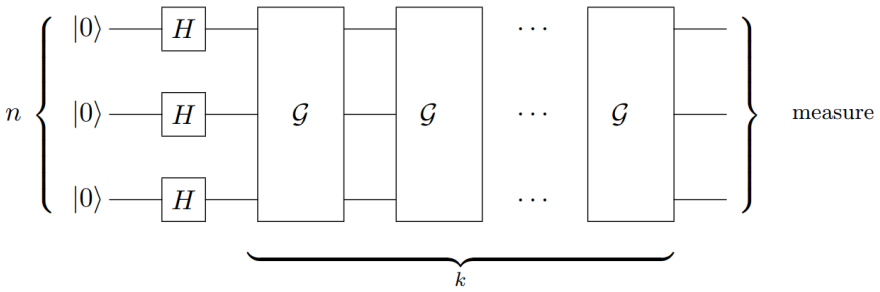


Figure 2.5: Quantum circuit for Grover's algorithm with k iterations. Picture taken from Ref. [28].

So why does this algorithm work? There are two main arguments used to illustrate this, but we will focus on the geometric argument. We start in the state

$$|U\rangle = \sin(\theta)|G\rangle + \cos(\theta)|B\rangle . \tag{2.9}$$

We note that the reflections (a) and (b), as described in the steps of the Grover iterate, increase the angle from θ to 3θ, which moves us towards a good state as illustrated in Figure 2.6:

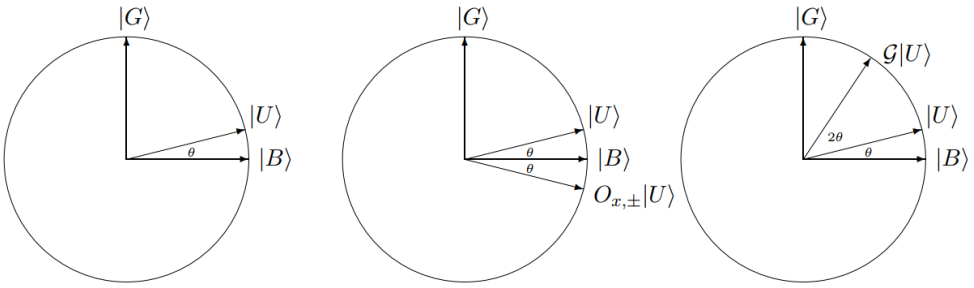


Figure 2.6: The rotations of a single Grover iterate $\mathcal{G}$. Picture taken from Ref. [28].

Additional reflections (a) and (b) increase the angles with another 2θ, so in general we have that after k applications of (a) and (b) we find ourselves with the following state

$$\sin((2k+1)\theta)|G\rangle + \cos((2k+1)\theta)|B\rangle .$$

If we measure this state, the probability of observing a solution is $P_k = \sin((2k+1)\theta)^2$. We want to have P_k as close to 1 as possible, which is the case if $k = \pi/4\pi - 1/2$. However, since we can only do an integer amount of Grover iterations, k has to be an integer as well. Choosing $\tilde{k}$ to be the integer closest to k our failure probability is ((assuming $t \ll N$)

$$1 - P_{\tilde{k}} = \cos\big((2\tilde{k}+1)\theta\big)^2 = \cos((2k+1)\theta) + 3(\tilde{k}-k)\theta)^2 \leq \sin(\theta)^2 = \frac{t}{N},$$

where we used that $|\tilde{k} - k| \leq 1/2$. Since $\arcsin(\theta) \geq \theta$, the total number of queries is $k \leq \pi/4\theta \leq \frac{\pi}{4}\sqrt{\frac{N}{t}}$.

Since Grover queries the oracle only once for during every Grover iterate $\mathcal{G}$, Grover's algorithm solves the unstructured search problem with $O(\sqrt{N})$ queries and $\lceil \log_2 N \rceil$ qubits. A randomised algorithm (which is in fact optimal in expectation) would need $O(N)$ queries to solve this problem, and therefore Grover's algorithm provides us with a quadratic speed-up.

Even though a quadratic speed-up is not as spectacular as some other quantum algorithms (i.e. the ones that provide exponential speed-ups), Grover's search is important because of its applicability: basically any classical algorithm that has some search component can be improved using Grover's algorithm as a subroutine. This also includes many basic optimisation applications where we are interested in finding optimal, not near-optimal, solutions. Examples are finding shortest paths, minimum spanning trees, and various other graph algorithms.

CHAPTER 3

Correlation clustering

In general, the objective of *clustering problems* is to group elements into a family of subsets, which we will name *clusters*, such that the elements within a cluster are more like to one another than elements in different clusters. This problem can be visualised by *graph clustering*. In this clustering formulation the nodes are the elements to be grouped in clusters and edges represent similarities between these elements. In our work, we will consider the *correlation clustering* problem. First defined by Harary in 1955 [29], the problem has a history of being repeatedly rediscovered under different names ever since [21, 30, 31, 32, 33]. Correlation clustering has proven to be a popular topic of research as many real world problems can be modelled using correlation clustering—examples are found in social psychology, statistical mechanics and biological networks. It is also common among computer vision tasks: in section 3.4 we will consider a specific example of one of those problems.

A notable survey paper on the topic is that of Schaeffer [34] in which she focuses on different definitions of clusters and measures of cluster quality, as well as presenting some general approaches to solving clustering problems. Another is the work by Böcker and

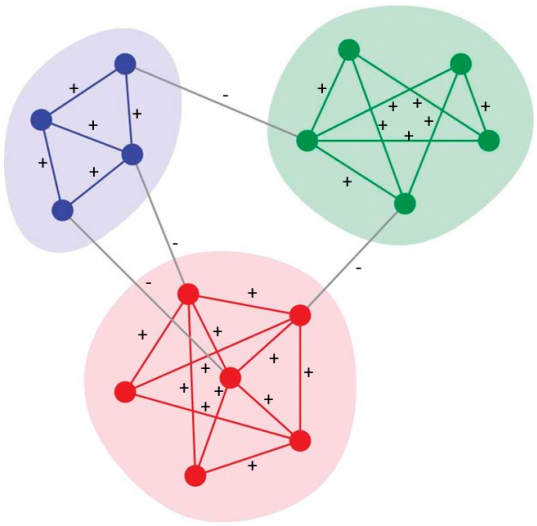


Figure 3.1: Example of a correlation clustering problem for which a solution exists without any disagreements, using a total of three different clusters.

Baumbach [31], which surveys exact methods for solving correlation clustering. However, since the quantum approximate optimisation algorithm is primarily an approximation method, we will mainly use a chapter on correlation clustering by Immorlica and Wirth in *Constrained Clustering: Advances in Algorithms, Theory and Applications* [35]: this chapter puts large emphasis on approximation methods and is therefore ideal for our purposes.

For approximation algorithms the *approximation ratio* is often used as the most important performance measure. Consider some class of optimisation problems P. For maximisation objectives, it is the convention that the approximation ratio of an r-approximation algorithm A, which finds approximate solutions for instance X in the class of problems P, is some number $r \in [0, 1]$ such that $\mathrm{OPT(X)} \leq f(A(X)) \leq r\mathrm{OPT(X)}$ for every $X \in P$, where $f(A(X))$ is the objective function value of approximation $A(X)$ and OPT(X) the optimal objective value of X. When dealing with minimisation objectives, $r \in [1, \infty)$ and we have that an r-approximation algorithm A needs to satisfy $r\mathrm{OPT(X)} \leq f(A(X)) \leq \mathrm{OPT(X)}$ for every $X \in P$.

3.1 Problem definition

There are some basic variants to correlation clustering, where you pick one out of i) minimise disagreements/maximise agreements or a combination of those, ii) unweighted/weighted and iii) bounded or unbounded number of clusters. Also, one can vary the type of graph structure that is studied. The most basic form is the following:

Definition 3.1. Let $G(V, E)$ be an undirected graph of $N = |V|$ nodes. Consider the clustering C to be a mapping from the elements to be clustered, V, to a set of cluster labels, so that u and v are in the same cluster if and only if $C(u) = C(v)$. Given edges (u, v) having two weights $w^+_{u,v}$ and $w^-_{u,v}$, the objective of correlation clustering is to find a clustering C that minimises

$$\sum_{C(u)=C(v)} w^-_{u,v} + \sum_{C(u)\neq C(v)} w^+_{u,v},$$

or equivalently, maximises

$$\sum_{C(u)=C(v)} w^+_{u,v} + \sum_{C(u)\neq C(v)} w^-_{u,v}.$$

The weights $w^+_{u,v}$ and $w^-_{u,v}$ are in this formulation non-negative and can be though of as positive and negative evidence towards co-association. For unweighted graphs, we have that every edge only has one weight labelled either $\langle + \rangle$ (equivalent to '+1') or $\langle - \rangle$ (equivalent to '-1'). Both problems are equivalent in the exact setting but differ in the approximation setting, as we will see in the next section.

Throughout this thesis, when we talk of MIN-DISAGREE, MAX-AGREE and MIN(DISAGREE-AGREE) as objectives of the correlation clustering problem we always have that for all weights $w_{u,v} \in \{-1, +1\}$ and the amount of clusters is unbounded (except for the amount of nodes which always upper bounds the correlation clustering problem).

3.2 Classical results

Some standard techniques that have been developed over the years to solve correlation clustering problems are region growing [36], techniques based on rounding linear programs, and combinatorial approaches that draw upon connections to sorting. This section will elaborate on how these techniques have been used for both the maximisation and minimisation variants to correlation clustering.

3.2.1 Maximising agreements

Results by Charikar et al., later extended by Tan [37], show that the maximisation problem on general graphs is *APX-hard*: it is in NP but does allow polynomial-time approximation algorithms with an approximation ratio bounded by a constant. To be precise, it is proven that it is NP-hard to to obtain results that are strictly better than a factor of $79/80 \approx 0.99$. The 2004 paper [21] by Bansal et al., famous for introducing correlation clustering to the computer science community, proposes a (trivial) method that produces a 0.5-approximation algorithm for the maximisation of agreements. Their algorithm looks at the total of the positive weights: if this exceeds the total of the negative weights all the items are placed in a single cluster, otherwise they put each item a singleton cluster. The 0.5-approximation factor is still quite far from the predictions by the complexity studies, and indeed considerable improvements have been made over the years. A factor of 0.7664 was obtained using semi-definite programming by Charikar [38] et al., and further improved to 0.7666 by Swamy [22] by utilising improved rounding techniques. We now give a sketch of this algorithm:

Swamy algorithm Let $e_i \in \mathbb{R}^n$ be a vector with all zeros except for the i^{th} element. This vector e_i represents a possible cluster i. The correlation clustering problem can now be formulated as

$$\begin{aligned} \max_{x} \quad & \sum_{e=(u,v)} w_e^+ x_u \cdot x_v + w_e^- (1 - x_u \cdot x_v) \\ \text{s.t.} \quad & x_v \in \{e_1, \ldots, e_n\} \text{ for every } v \in V. \end{aligned} \tag{3.1}$$

$x_v = e_i$ for any clustering, and for every vertex v assigned to cluster i, the objective function value becomes the weight of the agreements in clustering. We consider at most k clusters. We relax the constraints to get a semi-definite program:

$$\begin{aligned} \max_{x} \quad & \sum_{e=(u,v)} w_e^+ x_u \cdot x_v + w_e^- (1 - x_u \cdot x_v) \\ \text{s.t.} \quad & x_u \cdot x_v = 1 \quad \text{for all } v, \\ & x_u \cdot x_v \geq 0 \quad \text{for all } u, v, u \neq v. \end{aligned} \tag{3.2}$$

We solve the SDP to obtain solution $\{x_v \in \mathbb{R}^n\}$, which can be done up to an additive error ϵ in time that is polynomial in the program description size and $\log 1/\epsilon)$. Choosing a rounding procedure similar to Goemans-Williamson, we split our algorithm in two cases $k \geq 6$ and $k \leq 5$. For $k \geq 6$: choose 6 random vectors $r_1, \ldots, r_6 \in \mathbb{R}^n$ whose coordinates have the standard normal distribution. Randomly choosing this scheme gives a 0.7666-approximation

algorithm that produces at most 6 clusters. For $k \leq 5$: we now use a different relaxation:

$$\begin{aligned} \max_{x} \quad & \sum_{e=(u,v)} w_e^+ \frac{1+(k-1)(x_u \cdot x_v)}{k} + w_e^- \frac{(k-1)(1-x_u \cdot x_v)}{k} \\ \text{s.t.} \quad & x_u \cdot x_v = 1 \quad \text{for all } v, \\ & x_u \cdot x_v \geq \frac{-1}{k-1} \quad \text{for all } u, v, u \neq v, \end{aligned} \tag{3.3}$$

of which the solution can be rounded by choosing either 1 or 2 hyper-planes. This achieves an approximation ratio of 0.77. Therefore, this algorithm gives a 0.7666-approximation algorithm for maximising agreements in correlation clustering. Note that the same approximation ratio holds for the $k-$clustering variant. More information on this algorithm as well as references to the proofs can be found in the original work, Ref. [22].

The algorithms concerned so far work for general graphs—if the graph is complete the problem becomes significantly easier although still NP-hard. For complete graphs, Bansal et al. provided a *polynomial time approximation scheme* (PTAS) [21]: an algorithm that for some $\epsilon > 0$ is able to produce a solution that is within a factor $1-\epsilon$ of being optimal and runs in time polynomial in ϵ.

3.2.2 Minimising disagreements

The minimisation variant is considerably more difficult to solve than the maximisation problem for both general [21] and complete [38] graphs it is APX-hard. Though Karpinski and Schudy [39] proved the existence of a polynomial time approximation scheme on complete graphs and a fixed number of clusters, the design of sub-logarithmic approximations for general graphs is still an open problem [40, 41]. Just as was the case for the maximisation problem, Bansal et al. [21] provided the first constant-factor approximation algorithm for complete inputs: the algorithm adopts a local search method and has a constant-factor that is quite large. A significant improvement can be made at the expense of solving the natural linear program formulation of the problem, as was pointed out by Bertolacci and Wirth [42]. However, this comes with the cost of long computation times and huge memory demands. Nonetheless, using region-growing type rounding procedures Charikar et al. [38] were able to produce a 4-approximation algorithm which could eventually be improved further to 2.5. In 2015, Chawla et al. [43] proposed new rounding schemes for the standard linear programming relaxation of the correlation clustering problem, achieving approximation factors almost matching the integrality gap. For complete graphs their approximation is $2.06-\epsilon$ (integrality gap of 2), for complete k-partite graphs the approximation is 3 (matching the integrality gap) and for complete graphs with edge weights satisfying triangle inequalities and probability constraints, their approximation is 1.5 (integrality gap of 1.2). This is the best polynomial-time approximation algorithm known at the moment.

3.3 Quantum approaches

Whilst there are some papers that use QAOA to work on some form of graph clustering problems [44, 45, 46, 47, 48], the only paper we know of that considers correlation clustering in particular and mentions QAOA is the work by Pramanik and Chandra [49]. They use generalised Pauli operators to solve graph clustering, framed as a Max-d-Cut problems, with qudits ($d-$levels). They only show a derivation of a possible Hamiltonian formulation

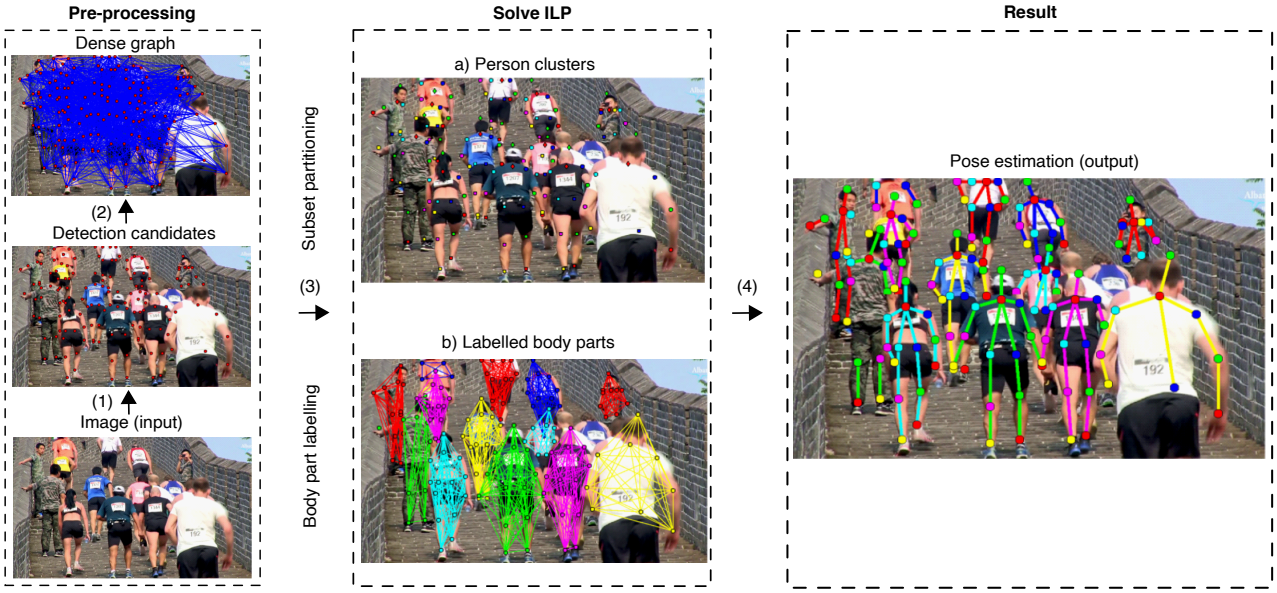


Figure 3.2: General procedure of the Deep(er)Cut algorithm. Starting from a single monocular image of multiple individuals, a neural network computes a sparse set of candidate body parts (1). Next, a densely connected graph is constructed which incorporates various types of interactions between the candidate body parts (2). The multi-person pose estimation is now formulated as an integer linear program (ILP) with an objective consisting of a clustering (a) and labelling (b) part. Solutions to the ILP describe the labelling of the edges and the clustering of the nodes, and therefore gives a joint pose estimation of multiple people. Adapted Figure taken from Ref. [51]

which is also suitable for correlation clustering, but do not show any extensive results on the performance of a QAOA implementation of this formulation. Their formulation is listed in Appendix A for completeness, will not be used in this work as we expect it to be inferior to the formulation we will propose for qudit systems.

3.4 An example: multi-person pose estimation

The following example was put forward by Bosch Research as an example of an optimisation problem they encountered in practice. It was the original motivation for this work to focus on correlation clustering.

Given an image that contains an unknown amount of persons, the goal of *multi-person pose estimation* is to estimate poses of the persons in the image by representing the orientation of every person in a graphical format. This is usually done in the form of connected nodes that form a *skeleton*, where any valid connection between two nodes is called a *limb*. There are several ways to solve the multi-person pose estimation, but we will focus on a method (and its improved version) called *Deep(er)Cut* as its objective generalises the correlation clustering objective [50, 51].

The general procedure of Deep(er)Cut is explained in Figure 3.2. In Deep(er)Cut, a neural network is used to compute a sparse set of *candidate body parts* D. Each candidate

$d \in D$ has a *unary score* for every body part $c \in C$, where C is a pre-defined set of body part classes (e.g. neck, head). Based on the *unary* scores parameters $\alpha_{dc} \in \mathbb{R}$ are computed, which can be viewed as the 'cost' of placing body part d in class c. Next, for every pair of distinct body part candidates $d, d' \in D$ and every two body part classes $c, c' \in C$ the *pairwise term* generates parameters $\beta_{dd'cc'} \in \mathbb{R}$ which represent the 'cost' of having body part d, belonging to body part class c, and body part c', classified as c', being part of the same person. Using the defined sets and parameters, the pose estimation problem is framed as an ILP with two types of binary variables: $x : D \times C \to \{0, 1\}$ for which holds that $x_{dc} = 1$ if body part candidate d is of body part class c, and $y : \binom{D}{2} \to \{0, 1\}$ such that $y_{dd'} = 1$ indicates that body candidates d and d' belong to the same person. Using the defined variables, one can construct the following objective that consists of a labelling and clustering part

$$\min_{x,y} \underbrace{\sum_{d \in D} \sum_{c \in C} \alpha_{dc} x_{dc}}_{\text{Labelling part}} + \underbrace{\sum_{dd' \in \binom{D}{2}} \sum_{cc' \in C} \beta_{dd'cc'} x_{dc} x_{d'c'} y_{dd'}}_{\text{Clustering part}}, \tag{3.4}$$

which can be constructed into an ILP by introducing new variables $z_{dd'cc'} = x_{dc} x_{d'c'} y_{dd'}$ and some constraints:

$$\begin{aligned} \min_{x,y} \quad & \sum_{d \in D} \sum_{c \in C} \alpha_{dc} x_{dc} + \sum_{dd' \in \binom{D}{2}} \sum_{cc' \in C} \beta_{dd'cc'} z_{dd'cc'} \\ & y_{dd'} \leq \sum_{c \in C} x_{dc} \text{ for all } dd' \in \binom{D}{2} \\ & y_{dd'} \leq \sum_{c \in C} x_{d'c} \text{ for all } dd' \in \binom{D}{2} \\ & y_{dd'} + y_{dd''} - 1 \leq y_{dd''} \text{ for all } dd'd'' \in \binom{D}{3}. \end{aligned} \tag{3.5}$$

Note how (3.5) is hard and hard to approximate, as it is a generalisation of the correlation clustering objective which is central to this thesis.

PART II

The quantum approximate optimisation algorithm: a survey

CHAPTER 4

The fundamentals

In 2014 Farhi et al. [6] introduced a new quantum algorithm that produces approximate solutions for combinatorial optimisation problems: *the quantum approximate optimisation algorithm*, often abbreviated as *QAOA*. QAOA can be thought of as a heuristic to prepare a superposition of bit strings with probability amplitudes heavily concentrated around the solution of an optimisation problem. Due to its shallow circuit depth and its hybrid nature—it is usually used in conjunction with a classical optimiser[1]—it is relatively robust to circuit errors, making it a candidate algorithm to run on NISQ devices. Soon after the publication of this seminal work, Farhi et al. [52] published another work in which they applied the algorithm to the combinatorial problem MAX-3-XOR. They showed that QAOA at $p = 1$ already produces a string that satisfies a better number of equations than the at that time best classical algorithm. Even though the classical algorithm community responded quickly by producing an algorithm with asymptotically even better performance [53], QAOA had attracted the attention of researchers hoping that it would be a candidate to first exhibit practical quantum supremacy.

The following two chapters provide a survey concerning the QAOA algorithm. In this chapter we will deal with the fundamentals of QAOA: we give a general introduction to the algorithm, apply the algorithm to an example in the form of the MAXCUT problem, show that the values of the strings it outputs are concentrated around some mean value and look at the link with quantum adiabatic computing. In Chapter 5 we will look more closely at more recent results in QAOA literature: we will discuss properties, generalisations and variants to the algorithm, parameter optimisation, practical applications and list some open problems in the field of QAOA.

4.1 The algorithm

Suppose we have an objective function $f(x)$ acting on n-bit strings $x \in \{0,1\}^n$, some domain $\mathcal{F}$, and our objective is

$$\max_{x \in \mathcal{F}} f(x). \tag{4.1}$$

This function can be mapped to a Hamiltonian that is diagonal in the computational basis, and we say that a Hamiltonian H_C *represents* a function $f : \{0,1\}^n \to \mathbb{R}$ if its eigenvalues

[1] For the reader familiar with variational quantum algorithms: QAOA can be thought of as a special Ansatz within the variational quantum algorithm approach.

satisfy

$$H_C |x\rangle = f(x) |x\rangle \text{ for all } x \in \{0,1\}^n. \tag{4.2}$$

Here we have that $x \in \{0,1\}^n$ labels the computational basis states $|x\rangle \in \mathbb{C}^{2^n}$.

Such a Hamiltonian H_C can be easily simulated using Controlled-NOT and Z-rotation gates if we can compute its expansion in terms of Pauli Z operators as

$$H_C = c_0 I + \sum_{j=1}^{n} c_1 Z_j + \sum_{j<k} c_{jk} Z_j Z_k + \ldots \qquad c_\alpha \in \mathbb{R}, \tag{4.3}$$

implementing the operation as unitary

$$U_C(t) = e^{-iH_C t}, \qquad t \in [0, 2\pi]. \tag{4.4}$$

For a problem on n binary variables $x \in \{0,1\}^n$, the quantum approximate optimisation algorithm (QAOA) consists primarily of the three components[2]:

1. **The initial state**
   The initial state should be trivial to implement, and should be a (superposition of) feasible solutions. For unconstrained optimisation, or when using a penalty function, it is often taken to be the equal superposition state, i.e.

   $$|s\rangle = |+\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle . \tag{4.5}$$

   .

2. **A family of cost (phase separation) operators**

   $$U_C(\gamma) = e^{-i\gamma H_C}, \tag{4.6}$$

   where $\gamma \in \{0, 2\pi\}$ and H_C acts on the basis states $|x\rangle$ as $H_C |x\rangle = f(x) |x\rangle$. Soft constraints can also be encoded in H_C through the use of a penalty function.

3. **A family of mixing operators**

   $$U_M(\beta) = e^{-i\beta H_M}, \tag{4.7}$$

   where $\beta \in \{0, 2\pi\}$ and H_M is usually written as a sum of tensor products of X operators. The mixing operators are required to:

   a) Preserve the feasible subspace $\mathcal{F}$.

[2] We have used the generalisation by Hadfield et al. [54], *the Quantum Alternating Operator Ansatz* that has the same abbreviation by design. In the current QAOA literature, people often do not distinguish between the original QAOA—which restricts itself to the uniform superposition as the initial state and the X-mixer—and the more general version by Hadfield.

b) Provide transitions between all pairs of states $\mathbf{x}, \mathbf{y}$ corresponding to feasible points:

$$\text{for all } \mathbf{x}, \mathbf{y} \in \mathcal{F} \text{ there exists } \beta^*, r \text{ such that } |\langle \mathbf{x}| U_M^r(\beta^*) |\mathbf{y}\rangle| > 0$$

Once the initial state and operators are established, the QAOA circuit is further specified by circuit depth parameter $p \in \mathbb{N}$ and the $2p$ angles $\gamma_1, \ldots, \gamma_p$ and $\beta_1, \ldots, \beta_p$. In QAOA, the problem instance is encoded in the QAOA operators and therefore this circuit itself is dependent on the problem instance. The algorithm consists of applying the cost (phase separating) operators and mixing operators in alternation to the initial state to obtain the state

$$|\gamma, \beta\rangle = \underbrace{U_M(\beta_p)U_C(\gamma_p)\ldots U_M(\beta_1)U_C(\gamma_1)}_{2p \text{ unitary operations}} |s\rangle = \prod_{k=1}^{p} U_M(\beta_k)U_C(\gamma_k) |s\rangle , \tag{4.8}$$

on which a computational basis measurement is performed.

Now that we have defined all components of QAOA, we can give the general procedure for the algorithm under the variational approach:

1. Begin with initial state $|s\rangle$
2. Initialise $2p$ parameters $\vec{\beta} = (\beta_1, \ldots, \beta_p), \vec{\gamma} = (\gamma_1, \ldots, \gamma_p)$
3. Construct $\left|\vec{\gamma}, \vec{\beta}\right\rangle$ from Equation (4.8).
4. A computational basis measurement is performed on this state, returning candidate solution $|x\rangle$ with probability $|\left\langle x \middle| \vec{\gamma}, \vec{\beta} \right\rangle|^2$. Repeating this procedure, the expected value of the cost function is given by

$$F_p(\vec{\gamma}, \vec{\beta}) = \left\langle \vec{\gamma}, \vec{\beta} \middle| H_C \middle| \vec{\gamma}, \vec{\beta} \right\rangle , \tag{4.9}$$

   which can be statistically estimated from the produced samples.
5. Repeat the above steps with updated sets of time parameters as part of a classical optimisation loop, used to update β, γ. We will denote the maximum over the different parameters as

$$M_p = \max_{\vec{\gamma}, \vec{\beta}} F_p(\vec{\gamma}, \vec{\beta}). \tag{4.10}$$

6. Stop when the convergence condition on M is reached, output the corresponding solution $|x^*\rangle$ and value M_p^*.

The quality of the final solution M_p^* strongly depends on p. In fact, in the original paper they show that

$$\lim_{p \to \infty} M_p = \max_x f(x), \tag{4.11}$$

where M_p depends on p through Equation (4.8). Hence, for p sufficiently large optimal angles exist such that the expected value of (4.9) is arbitrarily close to the optimal solution.

However, we do not know how large p has to be in order to obtain a certain quality of approximation, nor what the values of these parameters must be. Figure 4.1 gives a schematic overview of the QAOA procedure.

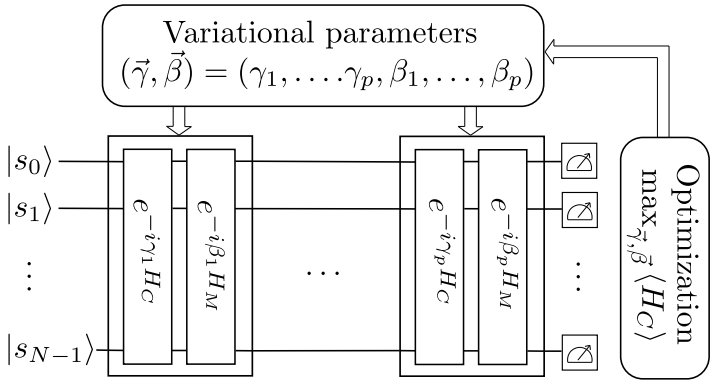


Figure 4.1: Schematic illustration of the variational quantum approach for QAOA. Optimal parameters are found through a loop with a classical optimiser.

4.1.1 An example: unweighted MAXCUT

We define some undirected graph $G = (V, E)$, with $N = |V|$. In unweighted MAXCUT, one tries to find a partition of the graph's vertices into two complementary sets S and $T = S^c = V \setminus S$, such that the number of edges between the set S and the set T is as large as possible. The objective function is

$$f(x) = \frac{1}{2} \sum_{(i,j)\in E} 1 - x_i x_j, \tag{4.12}$$

where $x_i \in \{-1, 1\}$. The interpretation of this variable is that $x_i = 1$ when node i is in the cut S and $x_i = 1$ if node i is in the complementary set T.

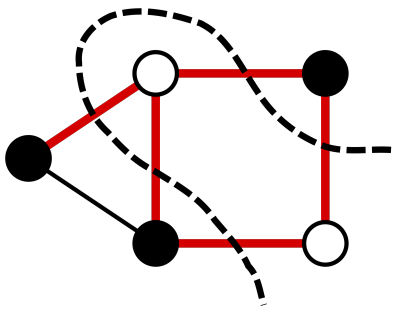


Figure 4.2: Example of MAXCUT problem instance with 5 nodes. The dotted line represents the optimal cut, creating two subsets of two and three nodes. This results in a cut where 4 out of 5 edges are shared between the subsets which is optimal for this instance.

The quantum analogue of this problem has the following (cost) Hamiltonian

$$H_C = \frac{1}{2} \sum_{(i,j)\in E} I - Z_i Z_j, \tag{4.13}$$

where Z_i is the previously introduced Pauli Z-operator. This is representation works because the eigenstates $|0\rangle$ and $|1\rangle$ have eigenvalues 1 and -1 respectively, so minimising this Hamiltonian is equivalent to maximising the MAXCUT objective. Since we have no constraints, we have that our feasible region $F = \{0,1\}^N$. As a mixer, we choose the standard mixing Hamiltonian

$$H_M = \sum_{i\in V} X_i, \tag{4.14}$$

and the initial state will be the superposition over all computational basis states

$$|s\rangle = \frac{1}{\sqrt{2^N}} \sum_x |x\rangle . \tag{4.15}$$

From these expressions we can use Equation (4.7) and (4.6) to construct our unitary operations, and define a QAOA circuit for any desired value of p. To illustrate why this is useful in the first place, note that in this case H_C can be written as

$$H_C = \sum_{(i,j)} h_{C,(i,j)}, \tag{4.16}$$

where $h_{C,(i,j)} = \frac{1}{2} I - Z_j Z_k$ and (i,j) describes the edge between nodes i and j. The expectation value of our quantum circuit is then

$$F_p(\vec{\beta},\vec{\gamma}) = \sum_{(i,j)} \left\langle \vec{\beta},\vec{\gamma} \right| h_{C,(i,j)} \left| \vec{\beta},\vec{\gamma} \right\rangle . \tag{4.17}$$

We will now consider $p = 1$. Note that each term in summation (4.17) is then of the form

$$\langle s| U_C(\gamma_1)^\dagger U_M(\beta_1)^\dagger h_{C,(i,j)} U_M(\beta_1) U_C(\gamma_1) |s\rangle . \tag{4.18}$$

Terms that do not involve qubits j and k commute through $h_{C,(i,j)}$, and therefore all terms in $U_M(\beta_1)$ and $U_C(\gamma_1)$ that do not depend on either one of these qubits cancel out. Our final expression involves only qubits on edge (i,j) and edges adjacent to (i,j). In fact, for general p one can show that the expression only depends on at most p edges away from edge (i,j). Therefore, the calculation of $F_p(\vec{\beta},\vec{\gamma})$ depends only on p and does not grow with the number of qubits n. However, some research indicates that for many practical cases p might have to scale with n in order to maintain good performance, but this will be elaborated on throughout the next chapter.

4.2 Concentration around the mean

We will now build upon our MAXCUT example to show some upper bound on the spread of H_C measured in the state $\left|\vec{\gamma},\vec{\beta}\right\rangle$. First, consider a regular graph of degree v. For finite

p there are only a limited amount of possible sub-graphs $g(j,k)$ possible, such that the following holds:

$$F_p(\vec{\beta},\vec{\gamma}) = \sum_g w_g f_g(\vec{\beta},\vec{\gamma}), \tag{4.19}$$

where w_g is the number of occurrences of sub-graph g and $f_g(\vec{\beta},\vec{\gamma})$ is the contribution of each sub-graph to the expectation value. The maximum number of qubits $q_{\max}$ are needed for f_g when the sub-graph forms a tree structure, and this number is that case equal to

$$q_{\max} = 2\left[\frac{(v-1)^{p+1}-1}{(v-1)-1}\right]. \tag{4.20}$$

For p fixed, we want to calculate

$$\begin{aligned}
&\left\langle \vec{\gamma},\vec{\beta}\right| H_C^2 \left|\vec{\gamma},\vec{\beta}\right\rangle - \left\langle \vec{\gamma},\vec{\beta}\right| H_C \left|\vec{\gamma},\vec{\beta}\right\rangle^2 \qquad (4.21)\\
&= \sum_{(j,k),(j',k')} [\langle s| U_C^\dagger(\gamma_1)\dots U_M^\dagger(\beta_p) H_{C,(j,k)} H_{C,(j',k')} U_M(\beta_p)\dots U_C(\gamma_1)|s\rangle\\
&\quad - \langle s| U_C^\dagger(\gamma_1)\dots U_M^\dagger(\beta_p) H_{C,(j,k)} U_M(\beta_p)\dots U_C(\gamma_1)|s\rangle\\
&\quad\quad \cdot \langle s| U_C^\dagger(\gamma_1)\dots U_M^\dagger(\beta_p) H_{C,(j',k')} U_M(\beta_p)\dots U_C(\gamma_1)|s\rangle].
\end{aligned}$$

Now consider sub-graphs $g(j,k)$ and $g(j',k')$. If both sub-graphs do not share common qubits, equation (4.21) will be zero. The sub-graphs $g(j,k)$ and $g(j',k')$ will not have any common qubits as long as there is no path in the instance graph from (j,k) to (j',k') of length $2p+1$ or shorter, hence we can replace p in Equation (4.20) with $2p+1$ and note that there at most

$$2\left[\frac{(v+1)^{2p+2}-1}{(v-1)-1}\right] \tag{4.22}$$

edges (j',k') which could contribute to the sum of (4.21). Therefore,

$$\left\langle \vec{\gamma},\vec{\beta}\right| H_C^2 \left|\vec{\gamma},\vec{\beta}\right\rangle - \left\langle \vec{\gamma},\vec{\beta}\right| H_C \left|\vec{\gamma},\vec{\beta}\right\rangle^2 \leq 2\left[\frac{(v+1)^{2p+2}-1}{(v-1)-1}\right]\cdot|E| \tag{4.23}$$

where we used that each summand is at most 1 in norm. This result implies that the sample mean of order $|E|^2$ values of $H_C(x)$ satisfies $|F_p(\vec{\gamma},\vec{\beta}) - H_C(x)| \leq 1$ with probability $1-\frac{1}{|E|}$, as we can estimate $F_p(\vec{\gamma},\vec{\beta})$ with a reasonable amount of samples.

4.3 Relation to the quantum adiabatic algorithm

Back in 2000, the same authors that proposed QAOA (Farhi, Goldstone, Gutmann but this time with Sipser as well) published a work concerning another, but related, quantum algorithm: the *Quantum Adiabatic Algorithm*, sometimes referred to as *Quantum Annealing* (QA) [55]. The cornerstone of the algorithm is the *Adiabatic Theorem*:

Theorem 4.1. *Suppose that the Hamiltonian of a system gradually changes from an initial form H_i to some final form H_f. The Adiabatic Theorem states that if the system was initially in the n^{th} eigenstate of H_i, then at the end of the process, the system will still be in the n^{th} eigenstate of H_f.*

A proof of this theorem can be found in most introductory textbooks for quantum mechanics, for example Griffiths' *Introduction to Quantum Mechanics* [56]. In QA one uses a time-dependent Hamiltonian $H(t)$ that interpolates between an initial Hamiltonian $H_i = H(0)$, whose ground state is easy to prepare, and a final Hamiltonian $H_f = H(T)$ whose ground state encodes the satisfying assignment for the problem you want to solve. T is defined as the total time of evolution. Commonly, an initial Hamiltonian H_i that has the uniform superposition $|+\rangle^{\otimes n}$ as ground state is used, because it can be easily prepared on a quantum computer. Generally, the time evolution can be represented as

$$H(t) = f(t)H_i + g(t)H_f, \tag{4.24}$$

where $f(t)$ and $g(t)$ are smooth functions with boundary conditions $f(0) = g(T) = 1$ and $f(T) = g(0) = 0$. A commonly used evolution scheme is linear evolution, using $f(t) = 1 - t/T$ and $g(t) = t/T$, but any scheme that satisfies (4.24) and the boundary conditions is possible.

Let us now look at the unitary time evolution in quantum annealing to observe the relationship with QAOA, where the roles of H_M and H_M are similar to the H_i and H_F, respectively, in the quantum annealing setting. For this, we start from the time-dependent Schrödinger equation (See Chapter 2, Equation (2.1)). We set $\hbar = 1$ for convenience, start at some initial time t_0 and describe the time evolution of some quantum state $|\psi(t)\rangle$ with the unitary operation $U(t, t_0)$ as

$$|\psi(t)\rangle = U(t, t_0)\,|\psi(t_0)\rangle\,. \tag{4.25}$$

The operator $U(t, t_0)$ is transitive, i.e. $U(t_2, t_0) = U(t_2, t_1)U(t_1, t_0)$. If we now combine (2.1) and (4.25) we obtain the following expression

$$i\frac{\partial}{\partial t}U(t, t_0)\,|\psi(t_0)\rangle = H(t)U(t, t_0)\,|\psi(t_0)\rangle\,. \tag{4.26}$$

This expression must hold for any normalised $|\psi(t)\rangle$, so we can define

$$i\frac{\partial}{\partial t}U(t, t_0) = H(t)U(t, t_0), \tag{4.27}$$

subject to the initial condition $U(t_0, t_0) = I$ as $|\psi(t)\rangle = |\psi(t_0)\rangle$ if $t = t_0$ (so when no time has passed yet). To find an expression for $U(t + \Delta t, t_0)$, we Taylor expand (4.27) up to second order in Δt and substitute the first order temporal derivative to find

$$U(t + \Delta t, t_0) = U(t, t_0) - iH(t)U(t, t_0)\Delta t + O(\Delta t^2). \tag{4.28}$$

By using the initial condition we obtain

$$U(t + \Delta t, t) = I - iH(t)\Delta t + O(\Delta t^2) = \exp\left(-iH(t)\Delta t\right) + O(\Delta t^2), \tag{4.29}$$

which holds if we only concern ourselves with terms up to Δt. We now use the transitivity property to derive an expression for $U(t, t_0)$ for arbitrary time steps $t - t_0$. Define time steps $\epsilon = (t - t_0)/N$, with $N \gg t - t_0$ such that (4.29) is approximately precise. Plugging this into (4.29) we obtain the following expression

$$U(t, t_0) = \prod_{k=1}^{N} U(t_0 + k\epsilon, t_0 + (k-1)\epsilon) = \lim_{\epsilon \to 0} \prod_{k=1}^{N} \exp\{-i\epsilon H(t_0 + (k-1)\epsilon)\}. \tag{4.30}$$

If we are now able to write this as an exponential of a sum, we might also be able to write it as an exponential of an integral when taking the limit $\epsilon \to 0$. However, this would require commutivity of $H(t_i)$ with $H(t_j)$ for every combination of $t_i, t_j \in [t_0, t]$. To maintain generality, we will therefore not make this assumption and continue by using the so-called *time-ordered exponential* instead to describe the evolution [57]:

$$U(t, t_0) = \mathcal{T} \exp\left\{-i \int_{t_0}^{t} d\tau H(\tau)\right\}, \tag{4.31}$$

where

$$\mathcal{T} \exp\left\{-i \int_{t_0}^{t} d\tau H(\tau)\right\} = I + \sum_{k=1}^{\infty} \frac{(-i)^k}{k!} \int_{t_0}^{t} dt_1 \int_{t_0}^{t_1} dt_2 \cdots \int_{t_0}^{t_{n-1}} dt_n H(t_1) H(t_2) \dots H(t_n). \tag{4.32}$$

This integral is very hard to evaluate, but we can approximate it by using the so-called *Suzuki-Trotter* decomposition. Define two operators A and B with some commutation relation $[A, B] \neq 0$. The first order Suzuki-Trotter decomposition is then given by

$$e^{x(A+B)} = e^{xA} e^{xB} + O(x^2), \tag{4.33}$$

for some parameter x. Usually, one applies the decomposition in the following way

$$\left(e^{\frac{x}{N}A} e^{\frac{x}{N}B}\right)^N = e^{x(A+B) + \frac{x^2}{2}[A,B] + O(\frac{x^3}{N^2})}, \tag{4.34}$$

which approximates the exponential up to an arbitrarily small error as N becomes larger and larger. We use our previously defined discretisation of $t - t_0$ into N small intervals of length Δt such that we can use the Suzuki-Trotter expansion to approximate the time-ordered exponential (4.31)

$$U(t, t_0) = \mathcal{T} \exp\left\{-i \int_{t_0}^{t} d\tau H(\tau)\right\} \approx \prod_{k=0}^{N-1} \exp\{-iH(k\Delta t)\Delta t\}. \tag{4.35}$$

We now take the $H(t)$ from QA, as defined in Equation (4.24)

$$\prod_{k=0}^{N-1} \exp\{-iH(k\Delta t)\Delta t\} = \prod_{k=0}^{N-1} \exp\{-i(f(k\Delta t)H_i + g(k\Delta t)H_f)\Delta t\}, \tag{4.36}$$

which can be further approximated using (4.33) to obtain our final unitary operation that describes the quantum annealing scheme

$$U_{\mathrm{QA}}(t, t_0) \approx \prod_{k=0}^{N-1} \exp\{-i(f(k\Delta t)H_i\} \exp\{-ig(k\Delta t)H_f)\Delta t\}. \tag{4.37}$$

Let us slightly rewrite (4.8) in order to compare this operation with QAOA, which has its unitary evolution defined by

$$U_{\mathrm{QAOA}}(\vec{\beta}, \vec{\gamma}) = \prod_{k=1}^{p} \exp\{-i\beta_k H_M\} \exp\{-i\gamma_k H_C\}. \tag{4.38}$$

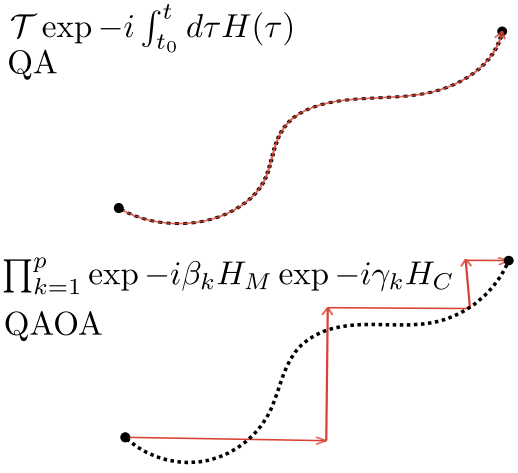


Figure 4.3: Comparison evolutions as a path through state space: QAOA (bottom) versus QA (top). Picture inspired from a figure in Ref. [58].

Inspecting equations (4.37) and (4.38), one observes that taking $\beta_k = f(k\Delta t)$, $\gamma_k = g(k\Delta t)$ and $N = p$ one finds a way to construct the approximated quantum annealing unitary from the QAOA unitary. Therefore, QAOA can be considered as some kind of discretised version of QA. However, even though QAOA can be viewed as some kind of trotterised version of QA, this does not mean that it also performs as a mere approximation. In fact, some works use optimal control theory arguments to show how QAOA separates from QA [59, 60]. In next chapter this will be explained in more detail.

But there is still one big elephant left in the room that we have not addressed so far: the adiabatic theorem holds when the evolution is gradual, so how large does the evolution time T have to be to in order to satisfy this criterion? It turns out that this depends on the minimum energy gap $\Delta_{\min}$ between the ground state and the first excited state during the evolution:

$$\Delta_{\min} = \inf\{|E_1(t) - E_0(t)| : t \in [0, T]\}, \tag{4.39}$$

where $E_0(t)$ is the ground state energy of $H(t)$ and $E_1(t)$ is the energy of the first excited state. To guarantee that the systems remains in the ground state, the runtime of the algorithm should typically scale as $T = O(1/\Delta_{\min}^2)$ [61]. For some Hamiltonians, this energy gap can become exponentially small. Finding the ground state of a problem of size N would in this case require a runtime of $O(\sqrt{N})$, which for problems that grow exponentially in problem size means that we can at most hope for a quadratic speedup (Grover-like) [62].

CHAPTER 5

Literature review

In this chapter we will dive more deeply into the state-of-the-art QAOA literature. At the moment of writing, no extensive survey on the topic exists. Throughout the time that I worked on this thesis, I tried to keep up-to-date with all published QAOA literature, of which the results are compiled into this chapter.

5.1 Properties, generalisations and variants

This section is structured such that it first covers the most important properties of QAOA, which are the characteristics of the depth p, its behaviour for different problems and their instances, its relation to quantum annealing and optimal control theory and the fact that it is computationally universal. In the last two subsections, we provide the reader with some generalisations and variants of QAOA.

5.1.1 The depth p

One of the most fundamental questions of QAOA is its performance for different values of p. Whilst a lot of studies perform numerical experiments in which they vary p (to some extent) for their problems, more rigorous analytical work showing performance guarantees have been done as well. In the previous chapter we already mentioned how Farhi et al. [52] provided performance guarantees for Max-3-XOR at $p = 1$. The original work that proposes QAOA also contains a $p = 1$ performance guarantee for MAXCUT on 3-regular graphs [6]. Wang et al. [63] extended this result, by providing an analytical expression of $p = 1$ performance on MAXCUT for general graphs. They also show that, for a special case of MAXCUT, the analogy with the 1D anti-ferromagnetic ring can be used for analysis. In the particular instance of MAXCUT similar to the 1D anti-ferromagnetic ring, called the ring of disagrees, they derive analytical expressions resemble the performance of QAOA for any p. Niu et al. [64] take a different approach to study the influence of p on the performance of QAOA. By analysing the success probability for realising state transfer in a 1D qubit chain using two qubit XY Hamiltonians and a single-qubit Hamiltonian, they obtain analytic expressions for the success probability as a function of p. In the derivation of these expressions, they assume that the time evolution under the mixing Hamiltonian is short and the same for all iterations whilst the cost Hamiltonian is described by a time evolution resembling a Grover oracle. In the limit of small p they show that the total number of steps QAOA requires to reach the target state has a Grover-like dependence on the circuit depth, and for p large the success probability actually grows exponentially in p. Defining the physical runtime as the total number of applied unitaries, their numerical results show three different scenarios of

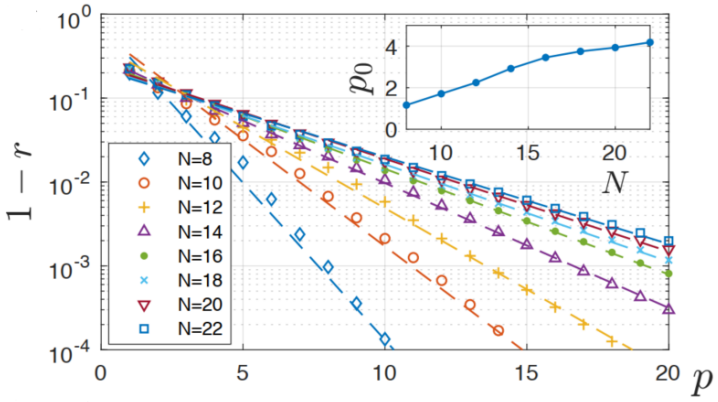


Figure 5.1: Average performance of QAOA on unweighted MAXCUT (100 instances) as measured by the fractional error $1 - r$, plotted on log-linear scale. Lines of different colours correspond to fitted lines for different problem sizes N, where the model function is $1 - r \propto e^{-p/p_0}$. The inset shows the dependence of the fit parameter p_0 on the system size N, indicating that p has to scale with N in order to maintain a desired performance. Figure taken from Ref. [61].

success probability scaling as a function of total run time: 1) exponentially suppressed, 2) exponentially growing and 3) steadily growing, happening in this order.

As already mentioned, most results rely on numerical work to study the behaviour of QAOA at higher p. The result of Zhou et al. [61] also shows exponentially (stretched) growing performance of QAOA applied on MAXCUT with increasing p, but only up to the system size that they study. Their results indicate that p has to increase with increasing system size N, as can be observed in Figure 5.1. Additionally, in practice the complexity of the optimisation of the variational parameters increases with increasing p as well: in a large numerical case study by Shaydulin and Alexeev [65] they find that the approximation ratio at some point only marginally increases with p as the gains in QAOA are cancelled out by the increased complexity of the variational parameter optimisation.

5.1.2 Problems and instances

What makes a problem suitable for QAOA in the first place? The performance of QAOA seems to strongly differ for different problems. Willsch et al. [66] evaluate the performance of QAOA by using three different measures on a set of problem instances, consisting of weighted MAXCUT problems and 2-satisfiability problems. Their results confirm that the overall performance of the quantum approximate optimisation algorithm strongly depends on problem type. Streif and Leib [67] investigated the kind of problems that can be solved exactly with level 1 QAOA. For one-dimensional target sub-spaces they identify instances within the implicitly defined class of Hamiltonians for which Quantum Annealing (QA) and Simulated Annealing (SA) have an exponentially small probability to find the solution. For two-dimensional solution sub-spaces they show that the depth of the QAOA circuit grows linearly with the Hamming distance between the two target states: this points to a new research direction of new encodings of combinatorial optimisation problems into problem Hamiltonians, where the desired solution is not necessarily the ground state but rather

exceptional in its interference (many states are close in Hamming distance).

Interestingly, in a large numerical case study by Shaydulin and Alexeev [65] it is shown that large variations can even exist within the same class of problem instances. For MAXCUT, Moussa et al. [68] find that the Laplacian spectrum and the density of the graph are critical aspects of the optimisation problem to determine the suitability of QAOA to solve this problem. They also use a machine learning approach to decide, given a certain MAXCUT configuration, whether QAOA or the classical approximation algorithm of Goemans and Williamson (GW) will yield the best results.

5.1.3 More on Quantum annealing

In the previous chapter we have shown how QAOA can be thought of as a discretised version of quantum annealing (QA). Some works consider the relation and (performance) differences between QAOA and QA more in-depth. Streif et al. [59] show how interference effects separate QAOA from Simulated and Quantum Annealing. They find problem instances that are exactly solvable for QAOA but for which QA and SA have an exponentially small probability to find the solution. Interestingly, they also show that for the problem instances they consider an efficient classical algorithm exists that also is able to find the solutions.

In a bench-marking study comparing the performance differences of QA and QAOA on problem instances consisting of weighted MAXCUT problems and 2-satisfiability problems, Willsch et al. [66] show that the D-Wave 2000Q quantum annealer outperforms QAOA executed on a simulator (IBM simulator and IBM Q Experience) based on their three different measures.

5.1.4 Relation to optimal control theory

Various authors also made the link between QAOA and optimal control theory. Yang et al. [69] view VQA as a closed-loop learning control problem and apply Pontryagin's minimum principle of optimal control theory to show that the optimal protocol for VQA has a "bang-bang" (square pulse) form. They also show that operations of QAOA are of this form, providing justification for the algorithm's performance. However, more recent work by Brady et al. [60] in the context of optimal control theory shows that in general, the optimal procedure has the pulsed (or 'bang-bang') structure of QAOA at the beginning and end but can have a smooth annealing structure in between ('bang-anneal-bang'). This would mean that a procedure that combines QAOA and quantum annealing would be the optimal procedure according to optimal control theory.

5.1.5 Universal computation

Interestingly, QAOA also turns out to be a universal quantum algorithm. In 2018, Lloyd [70] was the first to show that QAOA can be used to perform universal quantum computation: the dynamics of QAOA can be programmed to perform any desired quantum computation. The Hamiltonians required for this can be as simple as homogeneous sums of single-qubit Pauli X's and two-local ZZ Hamiltonians on a one-dimensional line of qubits. Morales et al. [71] extended upon this work. Their work provides complete proof that, under some precise conditions that are defined in the paper, one-dimensional QAOA is quantum computationally universal.

5.1.6 Generalisations

Perhaps as a stepping stone to his more seminal work we will discuss in the next paragraph, Hadfield et al.'s [72] 2017 paper provided a framework for designing QAOA circuits for a variety of combinatorial optimisation problems, with both hard (must be met) and soft constraints (want to minimise the violation). For a large variety of problems they discuss the design of the problem and mixing Hamiltonians, amongst which are: maximising properly coloured edges, finding graphs' chromatic number, the travelling salesman problem and single machine scheduling. This year, Ruan et al. [73] further generalised this by formalising different constraint types to linear equalities, linear inequalities, and arbitrary form in the context of QAOA.

Arguably, the most important generalisation is also by Hadfield et al. [54] in the form of the Quantum Alternating Operator Ansatz [1]. This generalises QAOA by considering more general parametrised families of unitaries rather than only those corresponding to the time evolution under a fixed local Hamiltonian, for a time specified by the parameter. This Ansatz supports the representation of a larger, and potentially more useful, set of states than the original formulation. In the same work they also lay out design criteria for mixing operators, detail mappings for eight problems, and provide a compendium with brief descriptions of mappings for a diverse array of problems. This establishes this work firmly as a reference in the field of QAOA. In the work of Wang et al. [74], they show one of the first results using this generalisation. By using the generalised W-state and an XY-Hamiltonian as mixer, they show that for the MAX-κ-Colourable-Sub-graph problem this setup outperforms the original QAOA formulation that uses a penalty function in the problem Hamiltonian to encode the constraints.

5.1.7 Variants

QAOA has also inspired researchers to come up with similar algorithms. Wei Ho and Hsieh [75] define the variational quantum-classical simulation (VQCS), which utilises a quantum simulator and a classical computer in a feedback loop for the purpose of preparing a non-trivial quantum state. Marsch and Wang [76] change the QAOA state evolution to alternating quantum walks and solution-quality-dependent phase shifts, and use the quantum walks to integrate the problem constraints of NPO problems. They also apply this scheme to minimum vertex cover, showing promising results using only a fixed and low number of optimisation parameters. Bapat and Jordan [77], using bang-bang control as a design principle for quantum optimisation algorithms, define a new version of simulated annealing, BBSA, which uses a bang-bang procedure similar to QAOA. They show that on their two bench-marking instances the bang-bang control algorithms (QAOA, BBSA) exponentially outperform both classical and quantum annealing-based algorithms (QAO and SA). An iterative version of QAOA that is problem-tailored, developed by Zhu et al. [78], is shown to converge much faster than conventional QAOA on a class of MAXCUT problems. Bärtschi and Eidenbenz [79] propose another variation to QAOA, called GM-QAOA, which uses Grover-like selective phase shift mixing operators. It is designed to perform well for constraint optimisation problems, but in theory works on any NP optimisation problem for which it is possible to efficiently prepare an equal superposition of all feasible solutions. They apply their formalism on MAX-k-vertex-Cover, TSP and Discrete Portfolio Rebalancing.

[1] This is also the generalisation we used in Chapter 4. As a bonus they were even able to maintain the original abbreviation for the quantum approximate optimisation algorithm (QAOA) for their generalisation.

Zhang, Zhang and Potter [80] propose a general framework for modifying QAOA, using QED inspired mixing Hamiltonians that preserve flow constrains, to solve constrained network flow problems. They numerically compare the performance of modified QED-QAOA and original (X-mixer) QAOA on a (classically easy) flow maximisation problem, and show that the quality of approximate solutions increases in a way that is consistent with exponential-in-problem size scaling. Egger, Mareček and Woerner [81] create a new way of QAOA initial state generation by using fractional solutions from relaxed combinatorial optimisation. Results for recursive QAOA applied on MAXCUT problems show a systematic increase in the size of the obtained cut for fully connected graphs with random weights, when Goemans-Williamson randomised rounding is utilised as a warm start. Finally, we would like to highlight the work by Li et al. [82] in which they propose modifications to both the Ansatz and variational parameter prescription of QAOA. The authors define the Gibbs objective function and Ansatz Architecture Search (AAS), which has the same variational parameters as QAOA but improved performance on certain Ising-type problems. The Gibbs objective function is an alternative to the energy expectation value for optimising the variational parameters, and AAS is a method for searching the discrete space of quantum circuit architectures for superior gate layouts. The reason that the Gibbs objective function has an advantage over using the expected energy of the cost Hamiltonian, is due to the fact that the exponential profile rewards increasing the probability of low energy, and de-emphasises the shape of the probability distribution at higher energies.

5.2 Parameter optimisation

The optimisation of QAOA parameters is itself a NP-hard problem: the optimisation objective is non-convex with low-quality non-degenerate (the Hessian has an eigenvalue 0) local optima [47, 61]. Therefore, classical optimisation or any other strategy to find good parameters are crucial in order to obtain good performance. One of the first works that specifically focuses on parameter finding was the work by Guerreschi and Smelyanskiy [83], in which they study how the overall performance of the variational algorithm is affected by the precision level and the choice of the optimisation method. Their results indicate that gradient methods (and quasi-Newton optimisers in particular) seem to be more effective than gradient-free ones for QAOA, however this does not suffice to claim that such optimisation procedures are the most suitable for hybrid schemes in general. In addition, we will later see that this might considerably change when we introduce noise into the system.

Another approach was to adopt machine learning to train the QAOA algorithm by Wecker et al. [84]. In their approach, the goal is to find a quantum algorithm that, given an instance of Max2Sat, will produce a state with high overlap with the optimal state. Using machine learning, a set of instances and optimised parameters was chosen to produce a large overlap for the training set. Testing the trained quantum optimiser on other random instances show improvement over annealing, with the improvement being most notable on the hardest instances. More groups have taken machine-learning approaches since. Khairy et al. [85] report that their policy network, trained through reinforcement learning, can reduce the optimality gap by a factor up to 8.61 compared with other off-the-shelf optimisers tested. They later extend upon this work, including another machine-learning approach in the form of kernel density estimation (RDE), and show that they can reduce this optimality gap even further by factors up to 30.15 (compared with the other commonly used optimisers) [86]. Other works using forms of reinforcement learning include that of Garcia-Saez et al. [87] and Yao et al. [88].

Other groups try to enhance the procedure whilst sticking to more conventional optimisation methods. Shaydulin et al. [89] study the use of a multi-start optimisation approach, using six derivative-free optimisers, within QAOA to improve the performance on graph clustering problems. Roch et al. [90] show that the usage of a Cross-Entropy method, which shapes the solution landscape of the variational parameters in QAOA, allows the classical optimiser to escape local optima more easily. They also empirically demonstrate that this approach can reach a significant better solution quality for the Knapsack Problem. Sung et al. introduced a new optimisation method designed for superconducting qubit processors called Model-Gradient-Descent (MGD). It is a surrogate model-based algorithm designed to improve the reuse of collected data, by estimating the gradient using a least-squares quadratic fit of sampled function values within a moving trusted region. In a large-study of hyper-parameter optimisation for a broad class of optimisers, they show that 1) hyper-parameter optimisation can be very important and that 2) MGD gives good results on their experiments with MAXCUT on 3-regular graphs, the Sherrington-Kirkpatrick model and the Hubbard model. They also show how hyper-parameter optimisation can greatly enhance the performance.

A lot of the work uses optimisers from the SciPy package [91], but for more noisy simulation one might also consider optimisers provided in the scikit-quant package by Lavrijsen et al. [92]. They study more advanced optimisation methods, specifically designed for noisy functions, from the field of applied mathematics and show how these outperform gradient-based optimisers in the presence of noise.

As p increases the optimisation problem generally increases in difficulty. However, might there be a connection between different parameters that we can exploit? In a work we already mentioned before, Zhou et al. also investigated the relation of the values of higher depth parameters to lower depth [61]. They exploit this structure to create two efficient parameter-optimisation heuristics to generate initial pints for the QAOA parameter optimisation, called FOURIER and INTERP:

INTERP For a given instance, iteratively optimise QAOA starting from $p = 1$ and increment p after obtaining a local minimum $(\vec{\gamma}_p^L, \vec{\beta}_p^L)$. To optimise for $p+1$, we take the optimised initial parameters from level p and produce initial points $(\vec{\gamma}_{p+1}^0, \vec{\beta}_{p+1}^0)$ according to

$$\left[\vec{\gamma}_{p+1}^0\right]_i = \frac{i-1}{p}\left[\vec{\gamma}_p^L\right]_{i-1} + \frac{p-i+1}{p}\left[\vec{\gamma}_p^L\right]_i,$$

for $i = 1, 2, \ldots, p+1$. $[\vec{\gamma}]_i$ denotes the i-th element of parameter vector $\vec{\gamma}$, and $\left[\vec{\gamma}_p^L\right]_0 = \left[\vec{\gamma}_p^L\right]_{p+1} = 0$. The expression for $\vec{\beta}_{p+1}^0$ is the same but we replace $\gamma \to \beta$.

FOURIER Instead of using $2p$ parameters $(\vec{\gamma}, \vec{\beta}) \in (0, 2\pi)^{2p}$ we define $2q$ parameters $(\vec{u}, \vec{v}) \in \mathbb{R}^{2q}$, where γ_i, β_i are written as functions of $(\vec{u}, \vec{v})$ through the following transformation:

$$\gamma_i = \sum_{k=1}^{q} u_k \sin\left[\left(k - \frac{1}{2}\right)\left(i - \frac{1}{2}\right)\frac{\pi}{p}\right],$$
$$\beta_i = \sum_{k=1}^{q} v_k \sin\left[\left(k - \frac{1}{2}\right)\left(i - \frac{1}{2}\right)\frac{\pi}{p}\right],$$

which is also known as the Discrete Sine/Cosine Transform, where u_k and v_k can be interpreted as the amplitude of the k-th frequency component for $\vec{\gamma}$ and $\vec{\beta}$, respectively. When optimising for $p+1$, the initial parameters are generated by re-using optimised amplitudes $(\vec{u}, \vec{v})$ from level p.
A comparison of three different optimisation routines, with and without these strategies, applied to 10 MAXCUT instances of 14-vertex w3R graphs is shown in Figure 5.2.

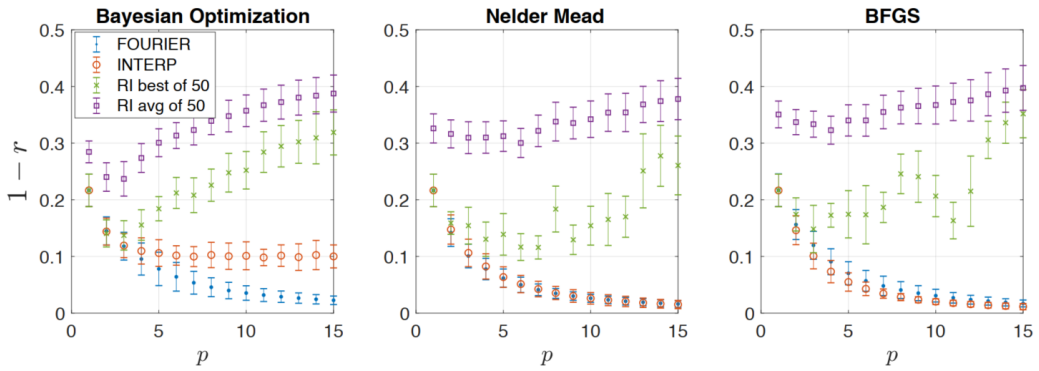


Figure 5.2: Comparison of different optimisers applied to 10 instances of weighted 3-regular MAXCUT problems with 14 nodes. Initial points are generated through FOURIER, INTERP or at random (RI). r is defined as the approximation ratio. Picture taken from Ref. [61].

We already showed in Chapter 4 how the work by Brandao et al. [93] shows that, if the parameters are fixed and the instance comes from a reasonable distribution, the objective function value is concentrated in the sense that typical instances have (nearly) the same value of the objective function. This indicates that it is possible to run QAOA in a way that reduces, or even eliminates, the use of the optimisation loop and may allow us to find good solutions with fewer calls to the quantum computer. Many works use this principle for initial state generation: they try very hard to solve one instance, and use this as an initial point for similar instances.

Some works also illustrate that it is possible to find values of the variational parameters without using classical optimisation at all. In the work by Streif and Leib [94], they present a strategy to find good parameters for QAOA based on topological arguments of the problem graph and tensor network techniques. In all their investigated cases, the results using the Tensor Network methods were either comparable or better than QAOA with training.

5.3 Applications, practical aspects and experiments

The following section gives an overview of all work in which QAOA is applied to specific problems in a certain field, and serves as a reference for the reader interested in performance results within a specific field.

5.3.1 Mathematics & Computer Science

Cook, Eidenbenz and Bärtschi [11] apply QAOA to the problem of MAX-k-vertex cover, extensively varying in different problem setups to investigate QAOA's performance. Their work includes a performance comparison between easy-to-prepare classical states and Dicke states, a performance comparison between two XY-Hamiltonian mixing operators, an analysis of the distribution of solutions via Monte Carlo sampling, and the exploration of efficient angle selection strategies. They show that the usage of Dicke states improves the performance compared to the easy-to-prepare classical states, the complete graph mixer improves performance relative to the ring mixer confirming the results by Wang et al. [63]. Their results also indicate that the standard deviation of the distribution of solutions decreases exponentially in p and that angle parameters are correlated such that they behave similarly to a discretised version of the Quantum Adiabatic Algorithm.

Multi-colouring problems, which have applications in flight scheduling, frequency allocation in networking and register allocation, were solved using QAOA by Oh et al. [12]. Their results show that QAOA (and VQE) can find one of the best solutions for each application they investigated, strengthening the case that QAOA can find an optimal solution in polynomial time for combinatorial problems in various fields.

Multiple formalisms have been proposed for solving MAX-k-CUT using QAOA. Fuchs et al. [13] provide an encoding that provides an exponential improvement for the number of qubits needed compared to previous encodings with respect to k. They test the algorithm and show that for $k = 2, 3, 4$ the algorithm is a good candidate to show quantum advantage on NISQ devices.

Shor's algorithm for factoring was (and still is) one of the key algorithms to propel the quantum computing field, but its execution requires hardware still way beyond the currently accessible NISQ devices. Anschuetz et al. [14] apply QAOA on the factoring problem by mapping the factoring problem to the ground state of an Ising Hamiltonian which energy is to be minimised. They show that it is in principle possible to factor using QAOA, but note that it is still an open question whether it will work under realistic constraints posed by imperfect optimisation methods and noise on quantum devices.

An and Lin [15] demonstrate that with an optimally tuned scheduling function, adiabatic quantum computing is able to solve a quantum linear system problem (QLSP) with polynomial runtime. This result also has implications for QAOA: with an optimal control protocol it should be able to achieve the same complexity in terms of the runtime, making it suitable to solve QLSP.

We have already seen that ML techniques can help in the variational process of QAOA, but Verdon et al. [16] show that a QAOA-like algorithm can also in itself solve deep learning problems. They introduce the Backwards Quantum Propagation of Phase errors (Baqprop) principle, a central theme upon which they construct multiple universal optimisation heuristics for training both parametrised quantum circuits and classical deep neural networks on a quantum computer. The Quantum Dynamical Descent unitary is of the form of QAOA, with the cost Hamiltonian being the effective phase shift induced on the parameters and the mixer Hamiltonian made up of generators of shifts of each register.

Matsumi et al. [17] investigate the use of QAOA employing quasi-maximum-likelihood for the decoding of classical channel codes. For $p = 1$, they derive theoretical expressions for the cost expectation for arbitrary binary linear codes. In addition, for the $(7, 4)$-Hamming code they analyse the impact of the degree distribution in associated generator matrix on the quantum decoding performance. Finally, they demonstrate the QAOA decoding performance in a real quantum device.

A perhaps more surprising application of QAOA was proposed by Szegedy [19], who showed that QAOA can be used for graph structure discovery—the most important property being isomorphism—by omitting the time-consuming parameter optimisation phase and utilising the dependence of QAOA energy on the graph structure for (randomly) chosen parameters to learn about graphs.

Verdon et al. [95] show that it is possible to apply QAOA to continuous optimisation problems. They introduce QAOA in the context of continuous optimisation by describing the algorithm in the model of continuous-variable quantum computing, where registers are quantum harmonic oscillators characterised by position and momentum operators. In addition, they also show that the algorithm allows for quadratic speed-up (similar to Grover's algorithm) for search problems.

5.3.2 Physics

The Sherrington-Kirkpatrick model is a mean-field model for a *spin glass*: a disordered magnetic alloy that exhibits unusual magnetic behaviour. Farhi et al. apply QAOA to this problem and propose a method for calculating, in advance, what energy the QAOA will produce for given parameters at fixed p in the infinite size limit for this model [10]. They find optimal parameters up to $p = 8$.

Wauters, Mbeng and Santoro [9] study the performance of QAOA on the fully connected p-spin model. Traditionally, QA was the tool to solve this problem but it is limited by the smallest gap encountered during the evolution, which vanishes in the thermodynamic limit when the system crosses a phase transition. They show that QAOA is able to find exactly the ferromagnetic ground state with polynomial resources, even when the system encounters a first order phase transition. However, they are unable to construct minima in the energy landscape associated with smooth parameters γ^*, β^*.

5.3.3 Biology

In 2018 Fingerhuth et al. [7] were the first to apply QAOA on a biology problem: lattice protein folding. Lattice protein models are coarse-grained representations of proteins that can be used to explore a vast number of possible protein conformations and to infer structural properties of more complex atomistic protein structures. Using different types of mixer Hamiltonians, they find that the best one obtains a maximum ground state probability of 0.477.

Tse et al. [8] use QAOA as an approach to image segmentation. They demonstrate their approach on small artificial and medical datasets, which comes from a coronary angiogram of the artery. A coronary angiogram is a procedure that uses X-ray imaging to see your heart's blood vessels. The test is generally conducted to see if there is a restriction in

blood flow to the heart. The data size is currently constrained only by the size of currently available quantum hardware, so the authors note that future development depends on the development of quantum hardware.

5.3.4 Other applications

Another notable application of QAOA is in logistics. Vikstal et al. [96] apply QAOA to the tail assignment problem in airplane allocation (mapped onto the Exact Cover problem), and use real world data to test their algorithm. They do not benchmark their results against classical algorithms though. Utkarsh, Behera and Panigrahi [97] attempt to solve the Vehicle Routing Problem (VRP). They conclude that in general, for a finite value of p, there is no guarantee that the solution achieved by QAOA corresponds to the most optimal solution of the original combinatorial optimisation problem.

QAOA was also applied on the discrete portfolio optimisation problem in finance, by Hodson et al. [20]. Portfolio rebalancing is the process of realigning the weightings of a portfolio of assets—it involves periodically buying or selling assets in a portfolio to maintain an original or desired level of asset allocation or risk. Their results give an indication the potential tractability of this application on Noisy Intermediate-Scale Quantum (NISQ) hardware, identifying portfolios within 5% of the optimal adjusted returns and with the optimal risk for a small eight-stock portfolio. They also compare the performance of the original QAOA [6] to the Quantum Alternating Operator Ansatz [54], and find that for their specific case the quantum alternating operator Ansatz is superior compared to the original quantum approximate optimisation algorithm.

5.4 Practical aspects

In this section we will consider the practical aspects that deal with the actual hardware implementation of QAOA. In general, it has been shown that the general circuit depth of the QAOA formulation of a combinatorial optimisation problem has to be at least the chromatic index of the corresponding graph G by Ostrowski et al. [98].

5.4.1 Compilation

One of the first works focusing on the compilation of QAOA is by Venturelli et al. [99]. In particular, they look at the compilation of QAOA to superconducting hardware architectures by framing compilation as a temporal planning problem. They verify this approach numerically by testing it on a range of compilation problems of QAOA circuits of various sizes to a realistic hardware architecture. This approach is surpassed by the later work of Oddi and Rasconi [100]. Their GRS (greedy randomised search) procedure, which synthesises NN-compliant quantum circuit realisations starting from a set of instances of QAOA tailored for the MAXCUT problem, outperforms the temporal planner approach by Venturelli et al. [99].

Another contribution to the compilation on superconducting hardware is by Abrams et al. [101]. They present an implementation of $XY(\beta, \theta)$ in a superconducting qubit architecture, making it possible to reduce circuit depth in (for example) MAXCUT QAOA.

Pichler et al. [102] devise an architecture to solve the maximum independent set

(MIS) problems with QAOA using neutral atom arrays trapped in optical tweezers. Appendix D gives a short introduction to such quantum computing hardware systems. They show that solutions of MIS problems can be efficiently encoded in the ground state of interacting atoms in 2D arrays by utilising the Rydberg blockade mechanism.

5.4.2 Noise resilience

Real quantum computers will be noisy—in particular the NISQ devices are expected to be subjected to noise. Several studies tried to focus on the noise resilience of QAOA. Alam, Ask-Saki and Ghosh [103] investigate the impact of various noise sources on the performance of QAOA both in simulation and on a real quantum computer from IBM (superconducting). Their results indicate that optimal number of stages (value of the depth p) for any QAOA instance is limited by the noise characteristics (gate error, coherence time, etc.) of the target hardware, which contradicted the at that time current perception that higher depth QAOA will provide monotonically better performance for a given problem compared to the low-depth implementations.

A more general work by Xue et al. [104] studies the effects of typical quantum noise channels on QAOA. The output state fidelity, the cost function, and its gradient obtained from QAOA decrease exponentially with respect to the number of gates and noise strength. They conclude that noise merely flattens the parameter space without changing its structure, so optimised parameters will not deviate from their ideal values.

Dong et al. [105] looks at potential solutions to enhance the noise resilience: they demonstrate that the error of QAOA simulation can be significantly reduced by robust control optimisation techniques, specifically, by sequential convex programming (SCP). The achievable fidelity of QAOA can significantly decrease in the presence of uncertainties in the Hamiltonian.

Wang et al. [106] show that noise in variational quantum algorithms causes the training landscape to have a barren plateau (vanishing gradient), for which the gradient vanishes exponentially in the number of layers. This is illustrated in Figure 5.3, which shows the concept of the Noise-Induced Barren Plateau (NIBP) and its effect on QAOA performance. This means that any variational quantum algorithm with a noise-induced barren plateau will have exponential scaling, which destroys the quantum speed-up. This has potentially large consequences for QAOA if p has to scale with problem size. Error-reduction is proposed as the (obvious) strategy to overcome this problem.

5.4.3 Experiments on quantum Hardware

We would like to highlight three papers that focus specifically on QAOA experiments performed on two different types of quantum computers: superconducting qubits and trapped ions.

Superconducting

Otterbach et al. [46] use QAOA in conjunction with a gradient-free Bayesian optimisation to train the quantum machine to solve clustering. The QAOA optimiser was run on a

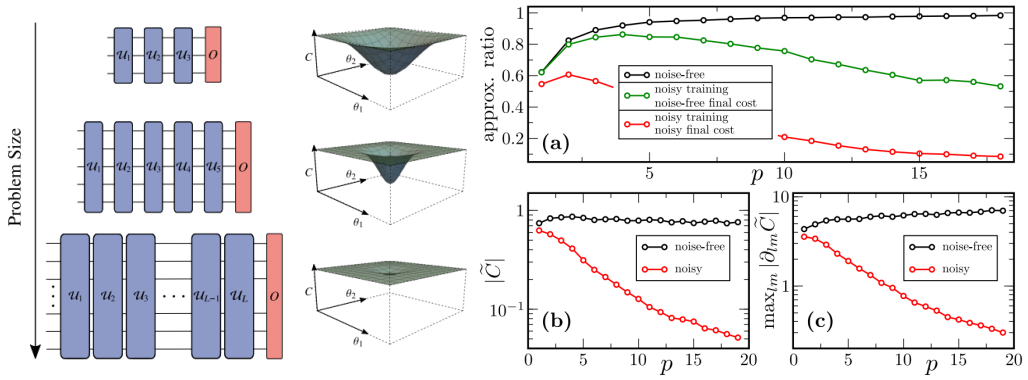


Figure 5.3: Left: Schematic diagram of the Noise-Induced Barren Plateau phenomenon. Note how the cost function landscape changes as the problem size increases. Right: QAOA performance in the presence of noise. Pictures taken from Ref. [106].

quantum processor consisting of 20 superconducting transmon qubits[2] with fixed capacitive coupling. The run time for 55 Bayesian optimisation steps with 2500 measurements per step is approximately 10 minutes. This run time includes network latency in the communication between the quantum processor and the classical processor running the high-level hybrid algorithm, as well as many other factors such as circuit compilation and job scheduling. For random problem instances, the vast majority of the traces reach the optimum of the clustering problem in fewer than 55 steps, demonstrating the potential of a hybrid quantum algorithm for clustering on a NISQ.

Google AI and collaborators. [107] use Google's Sycamore superconducting qubit quantum processor to run QAOA algorithms in solving Hardware grid problems with graphs matching the hardware connectivity, MAXCUT on 3-regular graphs and the fully connected Sherrington-Kirkpatrick model (see Figure 5.4). For problems defined on the hardware graph topology they obtain an approximation ratio that is independent of problem size and observe, for the first time, that performance increases with circuit depth. For problems requiring compilation (the MAXCUT and Sherrington-Kirkpatrick model), performance decreases with problem size but still provides an advantage over random guessing for circuits involving several thousand gates. This emphasises the importance of compilation of problems into real quantum hardware.

Trapped Ion

The first report of an experimental implementation of QAOA on a trapped ion quantum simulator was by Pagano et al. [108]. Their goal is to estimate the ground state energy of the transverse field Ising model with tunable long-range interactions. Their algorithm uses up to 40 trapped-ion qubits, which was at that point the largest ever realised on a quantum device. Single-shot high-efficiency qubit measurements in different bases give them access to the full distribution of bit-strings that is difficult (or potentially impossible) to model classically.

[2]Due to a fabrication defect, one of the qubits not tunable. Consequently, the device is treated as a 19-qubit processor (hence the name Rigetti 19Q) instead.

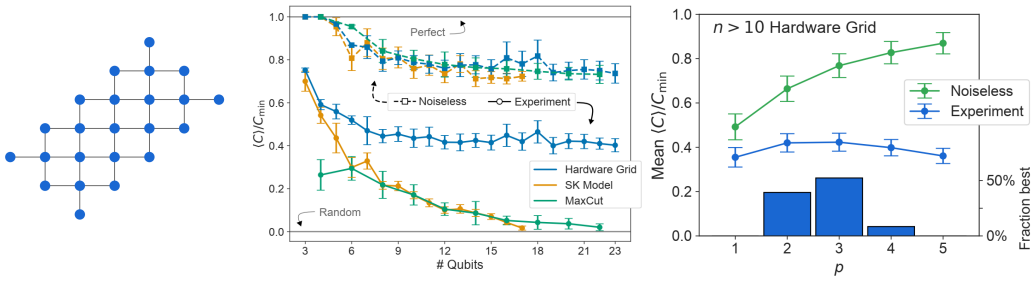


Figure 5.4: Left: hardware topology of Google's Sycamore. Middle: QAOA performance as a function of problem size, n. Each data point is the average over ten random instances (standard deviation given by error bars). Right: QAOA performance as a function of p on the hardware grid problems. In ideal simulation, increasing p increases the quality of solutions. However, for larger p the hardware errors dominate the potential gain. Pictures taken from Ref. [107].

With the addition of individual control over the interactions between qubits, their approach can be employed in this experimental platform to give insight into quantum chemistry and hard optimisation problems, such as MAX-SAT or exact cover, or be used for the production of highly entangled states of metrological interest.

5.5 Quantum supremacy?

But perhaps the most interesting question is whether QAOA can be used to obtain quantum supremacy. The first attempt to answer this came from Farhi [109], one of the original contributors to the original paper. He and co-author Harrow show that, based on plausible conjectures from complexity theory, there are choices of γ, β and the cost function such that even at the lowest depth ($p = 1$) QAOA can not be efficiently simulated using a classical computer. Their argument is based on the fact that if this would be possible, then this would also imply that P = NP. From this they conclude that QAOA is a great candidate for early demonstration of quantum supremacy.

In 2018, Crooks et al. [110] published the results of QAOA optimised on batches of problem instances. They report that their results exceed the performance of the classical polynomial time Goemans-Williamson [111] algorithm (the best known classical algorithm for MAXCUT) with modest circuit depth in solving MAXCUT problems, as displayed in Figure 5.5. The performance with fixed circuit depth is insensitive to problem size. However, they also state in their conclusions that their observations are suggestive only—it is prohibitively expensive to classically simulate the quantum MAXCUT algorithm on anything but small graphs. Definitive proof will have to await the anticipated arrival of a quantum computer with sufficient gate fidelity able to execute the algorithm on a larger scale. A month after this publication, Guerreschi and Matsuura [112] tried to quantify the order of number of qubits we would need to reach this point (Figure 5.6). Their numerical results show that classical solvers are very competitive until several hundreds of variables are considered for MAXCUT. Therefore, quantum speed-up would require hundreds of qubits which is still out of reach for the current state-of-the-art quantum hardware (53 qubits, IBM [113]).

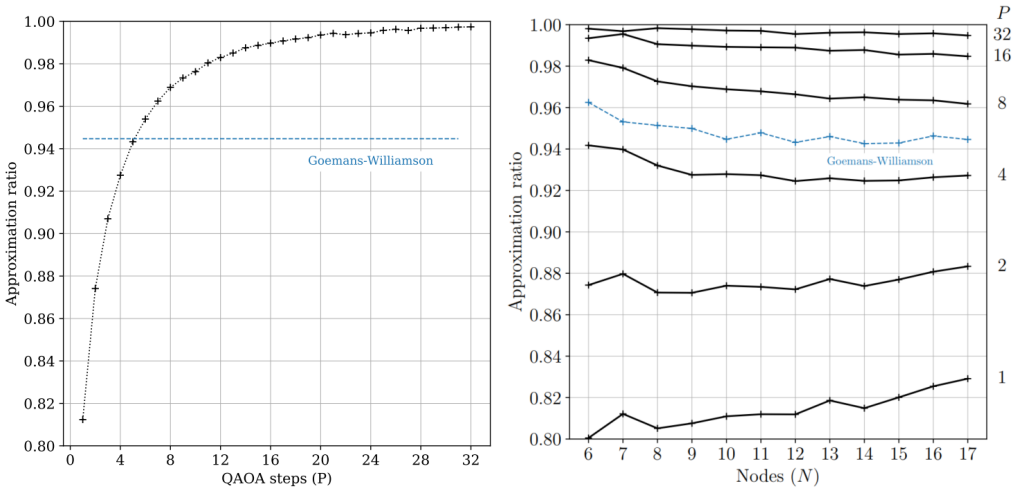


Figure 5.5: Left: Average approximation ratio of QAOA on MAXCUT with 10 nodes. The total data-set consisted of Erdős–Rényi 100 graphs with edge probability 0.5. Right: Approximation ratios of QAOA on MAXCUT as a function of the problem size N, also showing the performance of the Goemans-Williamson algorithm on the same test sets. QAOA exceeds the performance of the Goemans-Williamson algorithm by $p = 8$ (P represents the QAOA depth p in these figures). Picture taken from Ref. [110].

Imposing fine-grained versions of the non-collapse assumption, Dalzell et al. show that Quantum Approximate Optimisation Algorithm (QAOA) circuits with 420 qubits are large enough for the task of producing samples from their output distributions up to constant multiplicative error to be intractable on current technology [114].

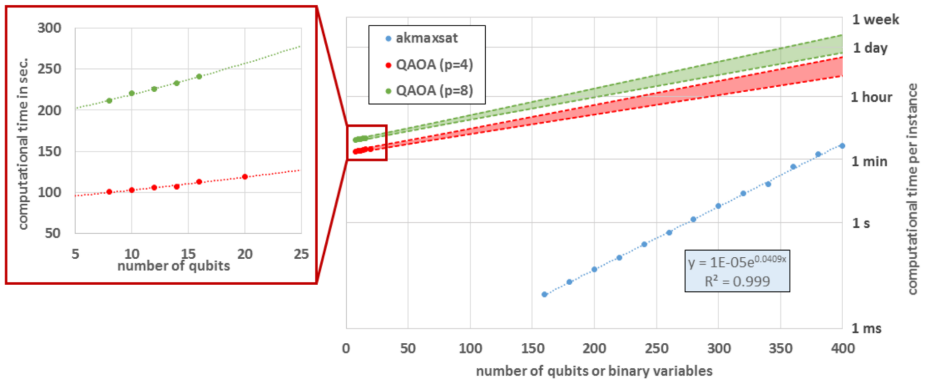


Figure 5.6: Computational cost of solving 3-regular MAXCUT with QAOA. The blue lines correspond to the (classical) AKMAXSAT solver, and the red and green marks to QAOA for $p = 4$ and $p = 8$, respectively. The areas indicate a 95% confidence interval for linear regression performed on the actual data for the QAOA algorithm. Picture taken from Ref. [112].

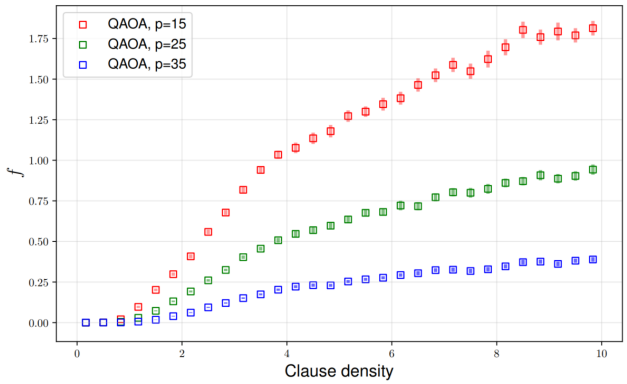


Figure 5.7: $f = E_q^{\mathrm{QAOA}} - \min(H_{\mathrm{SAT}})$ plotted against clause density for the 3-SAT problem. Note how f increases as the clause density increases, which means that the expectation value of the QAOA state is further away from the optimal solution.

However, there is also work that nuances our expectations for QAOA. Results by Hastings [115] suggest that local classical algorithms are likely to be at least as promising as the QAOA for (some problems of) approximate optimisation. By investigating instances of triangle-free MAXCUT and Max3Lin2 with different degrees, he finds that QAOA cannot achieve the same scaling as can be done by their defined class of global classical algorithms. In addition, Akshay et al. [116] report that QAOA exhibits a strong dependence on a problem instances constraint to variable ratio (Figure 5.7): this problem density places a limiting restriction on the algorithm's capacity to minimise a corresponding objective function (and hence solve optimisation problem instances), indicating that some classes of optimisation problems might be more suitable to QAOA than others.

5.6 Summary

QAOA has been generalised into the Quantum Alternating Operator Ansatz. The work by Hadfield et al. [54] provides guidelines to transform an optimisation problem into a QAOA formulation. It is preferred to have the initial state and mixer encoding the feasible subspace, as the use of penalty functions generally leads to inferior performance. There are also many variants to QAOA that might be more suitable for some problems.

Increasing the circuit depth p in theory always increases the performance of QAOA, but in practice this is limited (in a sense that it might lead to marginal gains or even a decrease in performance) by the increased complexity of the variational optimisation and additionally introduced noise in the quantum circuit.

There is no such thing as a 'free lunch' when it comes to classical optimisers: different optimisers perform differently on different problems, and hyper-parameters optimisation might turn one of the worst-performing optimisers in one of the best. However, we can generally say that in noise-free simulations, gradient-based optimisers might lead to the fastest convergence but in the presence of noise gradient-free black-box based optimisers might be more suitable. This is due to the fact that noise causes the training landscape to have a barren plateau, for which the gradient vanishes exponentially in the number of layers

of gate operations.

However, the good news is that good parameters values for instances belonging to the same class of problems are concentrated in parameter space—this greatly helps in the classical optimisation step. In fact, there are works that even argue that QAOA might be adopted without the classical optimisation step as long as you have access to good initial points. Additionally, methods like FOURIER and INTERP exploit relationships between different $\vec{\beta}, \vec{\gamma}$ for different values of p such that good initial points can be obtained. This is particularly useful for problems that require high values of p in order to obtain sufficient performance.

The performance of QAOA very much depends on the type of problem. Amongst the factors that need to be incorporated in determining the potential of QAOA on a problem are the constraint to variable ratio, the complexity of the cost and mixing Hamiltonians and chromatic number in case of a graph problem. In particular whether the constraints can be encoded into an efficient mixer is an important measure, as penalty function approaches have generally shown to have inferior performance.

There have been experimental studies on superconducting and trapped ion hardware. However, the performance generally degrades as the problem geometry (graph problems) is different from the actual hardware structure. Improved compilation, error reduction and error mitigation is needed to obtain better performances on these problems.

5.7 Open problems

Only for very specific problems and usually at low depth p, researchers have so far been able to obtain performance guarantees for QAOA, therefore most work relies on numerical bench-marking. It remains to be shown whether bounds can be obtained for more problems, in particular for $p > 1$.

The biggest problem with numerical bench-marking is the fact that it only allows for the study of small instances. Since even for these small instance sizes some results indicate that p already has to grow with the problem size, the optimisation landscapes increase in difficulty (but the way in which this happens is also not yet clear) and noise leads to vanishing gradients in these landscapes we are not sure how scalable QAOA truly is, or whether relevant quantum supremacy can be achieved on NISQ devices in the first place. And if this is possible, for which problems does this hold and for which not?

On a more specific level, more research is also needed into the variational parameters: for some problems it has been shown that the values of optimal parameters can be derived analytically, but is this also the case for any type of problem? And if not so, what are the best alternatives to derive these values?

Experimental considerations greatly restrict the potential of the algorithm. Effects of noise and compilation, as well as other problems that arise from hardware implementation, have so far been understudied. However, progress here is mainly tightened to the hardware developments, as actual experiments are the best way to test the actual performance.

QAOA essentially seems to be able to provide some sort of polynomial time approximation scheme for every optimisation problem: in theory we can get arbitrarily close to

the optimal solution as we increase p (and therefore the runtime), if we have access to good variational parameters β, γ. This means that QAOA has the potential of better quality of solutions, a better runtime or both. However, how large p has to be in order to obtain some approximation ratio that cannot be obtained through a classical algorithm for a certain problem, and whether this is still polynomial in p has yet to be proven.

PART III

Solving correlation clustering using an improved quantum approximate optimisation algorithm

CHAPTER 6

Hamiltonian formulations for correlation clustering

In this chapter several Hamiltonian formulations for correlation clustering will be introduced. Some are naturally more suitable for different types as hardware, and they generally have different energy landscapes in their QAOA implementation. We consider three main encodings: an edge-based, one-hot and multi-level encoding. After the encodings and their Hamiltonian formulations are introduced, we will discuss their circuit implementations, circuit complexities and look at an example. Finally, we will perform numerical experiments to get an indication of how the different formulations compare to one another in terms of performance.

Our objective functions are always formulated in the MIN-(AGREE-DISAGREE)-objective.[1] In the section on the numerical results we will show how we can convert this to the MAX-AGREE-objective.

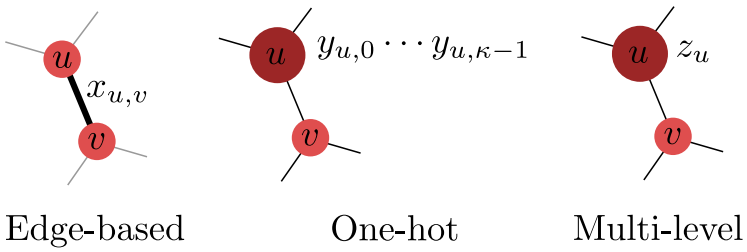


Figure 6.1: Schematic illustration indicating the way the variables encode the correlation clustering problem. The edge-based and one-hot formulations use binary variables, the multi-level formulation an integer variable. Throughout the text the different formulations will be explained.

6.1 Overview of Hamiltonian formulations

This section gives three different possible formulations for correlation clustering in the QAOA setting. Every formulation is defined by its variable domain, cost Hamiltonian, mixing Hamiltonian and initial state. In general, we consider correlation clustering problems described by a graph $G = (V, E)$ with edge weights $w_{(u,v)} \in \{-1, 1\}$ and $N = |V|$. We also

[1] This is more natural when using Pauli Z operators since this operator has eigenvalues '+1' and '-1'.

define a cluster mapping $C(u) : V \to K_\kappa$ that indicates the cluster label of node u, where the set $K_\kappa = \{0, 1, \ldots, \kappa - 1\}$ defines the labels of all possible clusters. For each formulation, an overview is given at the end of the section. Throughout the rest of the work, we will frequently refer to these formulations. Possible quantum circuit implementations as well as their gate complexities are discussed in the next section.

6.1.1 Edge-based

For this formulation we put an additional constraint on the graph-type: we consider only complete graphs $G = (V, E)$ with edges $E = \{(u, v) : u < v, v = 1, \cdots, N\}$. We define edge-encoded bit strings $x \in \{0, 1\}^{|E|}$ that represent a clustering by using entries $x_{(u,v)}$, such that

$$x_{(u,v)} = \begin{cases} 0 & \text{if } C(u) = C(v), \\ 1 & \text{otherwise} \end{cases} \tag{6.1}$$

In the MIN-(DISAGREE-AGREE) formulation of correlation clustering, the objective function can be defined using $x_{(u,v)}$ as

$$\min_x \sum_{(u,v)\in E} w_{(u,v)}(2x_{(u,v)} - 1). \tag{6.2}$$

In general, we can represent a Boolean variable in terms of a Pauli $Z-$operator as $(I - Z)/2$ [117]. The eigenvalues of Z are 1,-1 with eigenvectors $|0\rangle$ and $|1\rangle$, respectively. Therefore, measuring $(I - Z)/2$ results in either eigenstate $|0\rangle$ with eigenvalue 0 or $|1\rangle$ with eigenvalue 1, as desired. We represent our objective function in the following diagonal Hamiltonian:

$$H_{\text{problem}} = -\sum_{(u,v)\in E} w_{(u,v)} Z_{(u,v)} \tag{6.3}$$

In this sum, we use the convention that local operators are described only by their local operation, leaving out all identity operations that operate on the remaining Hilbert space. This convention will be used throughout the entire chapter, unless stated otherwise. Not every element in the space of x corresponds to a valid clustering: if for three nodes u, v, w we have that $C(u) = C(v)$ and $C(u) = C(w)$, then we must have that $C(v) = C(w)$. This property is called *transitivity*. In terms of Boolean logic operators we can write this down as

$$\begin{gathered} x_{(u,v)} = 0 \wedge x_{(u,w)} = 0 \Rightarrow x_{(v,w)} = 0 \\ \text{or, equivalently} \\ x_{(u,v)} = 1 \vee x_{(u,w)} = 1 \vee x_{(v,w)} = 0, \end{gathered} \tag{6.4}$$

which must hold for all combinations of u, v, w that are part of the set of edges. For general graphs the constraints become more complicated: cycles of every length should be considered in this case, which makes the constraints considerably more difficult.

We can take two approaches to impose the transitivity constraints within our Hamiltonian formulation:

1. Create a penalty function that adds a penalty value for every constrained that is violated.

2. Construct an initial state and driver Hamiltonian that respects this constraint.

Let us first discuss option 1 and consider 2 in the next section. In the penalty function approach, the problem Hamiltonian is now a linear combination of two Hamiltonians:

$$H_C = H_{\text{problem}} + \lambda H_{\text{constraints}}, \tag{6.5}$$

where λ is a penalty for violating the constraints. To construct $H_{\text{constraints}}$ we use (6.4) and note that the expression

$$\frac{1}{8}(1 + Z_{(u,v)})(1 + Z_{(u,w)})(1 - Z_{(v,w)}) \tag{6.6}$$

gives 1 for the quantum state $\left|0_{(u,v)}\right\rangle \left|0_{(u,w)}\right\rangle \left|1_{(v,w)}\right\rangle$ and zero for the seven other possible states. By summing over the three possible permutations we obtain the expression for a single triangle. Now we expand the expression and sum over all edges to obtain the final Hamiltonian expression for the constraints

$$\begin{aligned} H_{\text{constraints}} = \sum_{u<v<w} \frac{1}{8} \quad & [3 + Z_{(u,v)} + Z_{(u,w)} + Z_{(v,w)} - Z_{(u,v)}Z_{(u,w)} - \cdots \\ & \cdots Z_{(u,v)}Z_{(v,w)} - Z_{(u,w)}Z_{(v,w)} - 3Z_{(u,v)}Z_{(u,w)}Z_{(v,w)}]. \end{aligned} \tag{6.7}$$

Since we have no restrictions on the domain, we can use the conventional Pauli X-mixer of which the time evolution allows transitions from and to all possible 1-qubit states. The total mixing Hamiltonian is the sum of Pauli X-operations on every variable:

$$H_M = \sum_{(u,v)\in E} X_{(u,v)}. \tag{6.8}$$

Our initial state is the uniform superposition over all possible edge variable combinations, e.g.

$$|s\rangle = |+\rangle^{\otimes |E|} = \frac{1}{\sqrt{2^{|E|}}} \sum_{x\in\{0,1\}^{|E|}} |x\rangle \tag{6.9}$$

We have now arrived at the point which we can formally define all parts of our first formulation:

Formulation 6.1. Edge-based (EB)
Domain: Bit strings $x \in \{0,1\}^{|E|}$ with entries $x_{(u,v)}$, such that

$$x_{(u,v)} = \begin{cases} 0 & \text{if } C(u) = C(v), \\ 1 & \text{otherwise.} \end{cases}$$

Cost Hamiltonian:

$$H_C = H_{\text{problem}} + \lambda H_{\text{constraints}},$$

where H_{problem} and $H_{\text{constraints}}$ are given by equations (6.3) and (6.7), respectively.

The mixing Hamiltonian:

$$H_M = \sum_{(u,v)\in E} X_{(u,v)}$$

The initial state:

$$|s\rangle = |+\rangle^{\otimes |E|}$$

Constraints through initial state and mixer?

In general, it has been shown that finding a driver Hamiltonian for an arbitrary set of constraints is NP-hard [118]. Let us now show that, even if we find such a driver Hamiltonian for the transitivity constraints, it will be computationally expensive to execute this mixing on general graphs due to the fact that it cannot be decomposed into local operations.

As already established in Chapter 4 a mixing unitary $U_M(\beta)$ is required to have the following properties [54]:

1. Preserve the feasible subspace $\mathcal{F}$.
2. Provide transitions between all pairs of states x, y corresponding to feasible points:

$$\begin{aligned} &\text{for all } x, y \in \mathcal{F} \text{ there exists } \beta^*, r \text{ such that} \\ &\qquad\qquad |\langle x| U_M^r(\beta^*) |y\rangle| > 0 \end{aligned}$$

We make the following claim considering the design of a mixer in the edge-based encoding:

Theorem 6.1. *For complete graphs $G = (V, E)$ no k-local, where $k < |E|$, mixing unitary exists that satisfies the above design criteria in the edge-based Hamiltonian formulation.*

Proof. We consider some complete graph $G = (V, E)$ with $N = |V|$ nodes and $|E|$ edges. We define string x of $|E|$ variables $x_{(u,v)} \in \{0, 1\}$ according to describing the edge-based state according to (6.1). Let us first consider the transition from a state x to another state x' in which we want to change at one variable $x_{(u,v)} \to x'_{(u,v)}$. To check whether the transition

$x \to x'$ is allowed, we need to verify whether we stay in the feasible subspace, i.e. satisfies the transitivity constraints given by

$$x'_{(u,v)} + x_{(u,w)} + x_{(v,w)} \neq 2 \text{ for all } w \in V \setminus \{u, v\}. \tag{6.10}$$

The largest Hamming weight change of the set of *smallest required transitions* would be the transition to (or from) state $y = (1, 1, ..., 1)$, defined as the state in which all nodes are in the same cluster, from (or to) a state z which has all nodes but one, let us say node i, in the same cluster. The transition of $y \to z$ requires a change in Hamming weight of at least $N - 1$, since all variables $y_{(i,v)}, v \in V \setminus \{i\}$ have to change from 1 to 0 as node i is no longer in the same cluster as all other nodes $v \neq i$. Since any k-local operation can only change the Hamming weight by k, we require the operation to be at least $(N - 1)$-local. However, before flipping any of the edges (i, v) by the same argument as before we need to make sure that this operation satisfies the transitivity constraints (6.10) for all of these edges:

$$y_{(i,v)} + y_{(i,w)} + y_{(v,w)} \neq 2 \text{ for all } w \in V \setminus \{i, v\} \text{ for all } v \in V \setminus \{i\}.$$

Hence, our mixer needs to operate on a total of $N(N - 1)/2 = |E|$ variables. Therefore, the mixing operation has to be global and cannot be decomposed into local mixing operations. ■

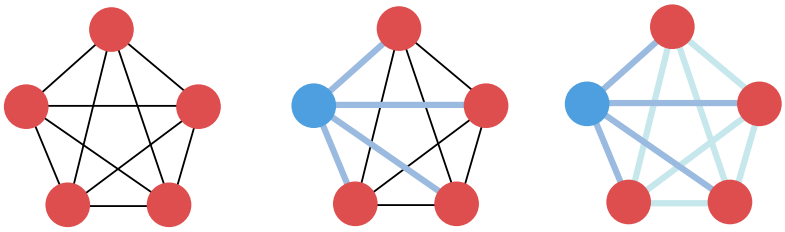


Figure 6.2: Graphical depiction of the proof for a complete graph with five nodes. When we want to transition from the singleton-cluster state to the state where all nodes but one are in the same cluster—where cluster labels are indicated by the colours of the nodes—we see that we need to change at least all variables from the edges connected to this node. However, we still need to satisfy the transitivity constraints for all triangles these edges are part of, and this accounts for all remaining edges.

6.1.2 One-hot

We define bit strings $y \in \{0, 1\}^{\kappa N}$ consisting of binary variables $y_{u,i}$ where $u \in V$ and $i \in K_\kappa$ is a cluster label, such that

$$y_{u,i} = \begin{cases} 1 \text{ if } C(u) = i, \\ 0 \text{ otherwise.} \end{cases} \tag{6.11}$$

In this formulation, contrary to what we had in the edge-based formulation, the total amounts of clusters κ can be varied. One-hot formulations have been applied to the graph colouring problem in the work of Wang et al. [74], which will be used for this work as a reference to be able to define a similar formulation for the correlation clustering problem. Our objective and

constraint are defined as

$$\begin{aligned} &\min_{y} \sum_{(u,v)\in E} \sum_{i\in K_\kappa} -w_{(u,v)} y_{u,i} y_{v,i} + \sum_{(u,v)\in E} \sum_{i\neq j\in K_\kappa} w_{(u,v)} y_{u,i} y_{v,j} \\ &\text{s.t.} \sum_{i\in K_\kappa} y_{u,i} = 1 \quad \text{for all } u \in V. \end{aligned} \tag{6.12}$$

Encoding this to a Hamiltonian requires some constants to be taken into account in order to have the same objective value for equivalent solutions as we had for the edge-based encoding. We write the cost Hamiltonian as

$$H_C = \sum_{(u,v)\in E} \left(a(u,v) + b(u,v) \sum_{i\in K_\kappa} Z_{u,i} Z_{v,i} \right), \tag{6.13}$$

where $a(u,v)$ and $b(u,v)$ are constants. We note that this Hamiltonian only consists of two-body terms, in contrast with the edge-based encoding that only used one-body operations. Even though the constants are not practically relevant in the quantum circuit implementation—they only add a global phase—let us derive their values for the sake of completeness. When two nodes are in the same cluster the corresponding bit strings will be the same on all κ positions, while if they are in different clusters the strings will differ on exactly two positions and match on $\kappa - 2$ places. This gives us a set of two equations in $a(u,v)$ and $b(u,v)$

$$\begin{aligned} &a(u,v) + \kappa b(u,v) = w_{(u,v)} && \text{(same cluster)} \\ &a(u,v) + (\kappa-2) b(u,v) - 2b(u,v) = -w_{(u,v)} && \text{(different cluster).} \end{aligned} \tag{6.14}$$

Solving this set of equations (6.14) gives $a(u,v) = (2-\kappa) w_{(u,v)}/2$ and $b(u,v) = w_{(u,v)}/2$, such that our cost Hamiltonian becomes

$$H_C = \frac{1}{2} \sum_{(u,v)\in E} \left((2-\kappa) w_{(u,v)} + w_{(u,v)} \sum_{i\in K_\kappa} Z_{u,i} Z_{v,i} \right). \tag{6.15}$$

For the edge-based encoding, we have proven that there is no 'natural' way of incorporating the constraints through the initial state and mixer. This is not the case for the one-hot encoding, as has already been shown by Wang et al. [74]. Following their approach, we define our the mixing Hamiltonian on a single node u as

$$h_M = \sum_{(i,j)\in R} X_{u,i} X_{u,j} + Y_{u,i} Y_{u,j}, \tag{6.16}$$

where the pairs (i,j) in the set R determines the *connectivity* of the mixer. For example, if $R = \{(i,j) | i < j, j = 1, 2, \cdots, \kappa\}$, we have that we have a *complete mixing Hamiltonian* and when $R = \{(i,j) | i = j+1$ if $j = 1, 2, \cdots, \kappa - 1$ and $i = 0$ if $j = \kappa\}$ (periodic boundary conditions), we refer to it as the *ring mixing Hamiltonian*. We can choose any set R in principle, but in our case we will always use the complete mixing Hamiltonian in the one-hot encoding unless stated otherwise. Our full mixing Hamiltonian is obtained by summing over all nodes u, such that

$$H_M = \sum_{u\in V} h_M. \tag{6.17}$$

For the initial state we want to use a state that is a (superposition of) feasible

state(s). Using the *generalised W-state*, which is a superposition of states with Hamming weight 1, for every node in fact provides us with an equal superposition of all feasible states. For a single node, this state is given by

$$|W_\kappa\rangle = \frac{1}{\sqrt{\kappa}}(\underbrace{|100\ldots0\rangle}_{\kappa} + \underbrace{|010\ldots0\rangle}_{\kappa} + \underbrace{|000\ldots1\rangle}_{\kappa}), \tag{6.18}$$

such that our entire initial state becomes

$$|s\rangle = |W_\kappa\rangle^{\otimes N}. \tag{6.19}$$

The full description of the one-hot encoding is summarised in the following box:

Formulation 6.2. One-hot (OH)
Parameters: R: the set of all possible transitions from cluster i to cluster j, κ: the maximum amount of clusters.

Domain: Bit strings $y \in \{0,1\}^{\kappa N}$ with entries $y_{(u,v)}$, such that

$$y_{u,i} = \begin{cases} 1 \text{ if } C(u) = i, \\ 0 \text{ otherwise.} \end{cases}$$

Cost Hamiltonian:

$$H_C = \frac{1}{2} \sum_{(u,v)\in E} \left((2-\kappa) w_{(u,v)} + w_{(u,v)} \sum_{i \in K_\kappa} Z_{u,i} Z_{v,i} \right)$$

The mixing Hamiltonian:

$$H_M = \sum_{u \in V} \sum_{(i,j)\in R} X_{u,i} X_{u,j} + Y_{u,i} Y_{u,j}$$

The initial state:

$$|s\rangle = |W_\kappa\rangle^{\otimes N}$$

Reducing the amounts of qubits

Currently, we have that every node can be put in every possible cluster. However, this creates redundancy in the total solution space: we have a lot of different strings that resemble identical correlation clustering solutions. For example, putting all nodes in a single cluster with label i is the same as putting all notes in a different cluster with label $j \neq i$. Since the one-hot encoding is very expensive in the amount of qubits it needs (κN), we propose the following reduction: we give node i access to i clusters. The amount of qubits that we then

need, given κ, is then given by[2]

$$\sum_{i \leq \kappa} i + (N - \kappa)\kappa = \frac{1}{2}\kappa(\kappa + 1) + (N - \kappa)\kappa = \frac{1}{2}\kappa(2N - \kappa + 1). \tag{6.20}$$

This requires some changes to be made in the indexing in the formulation, of which the results are summarised in the following box:

Formulation 6.3. One-hot reduced (OHr)
Parameters: R: the set of all possible transitions from cluster i to cluster j, κ: the maximum amount of clusters.

Domain: Bit strings $y \in \{0,1\}^{\frac{1}{2}\kappa(2N-\kappa+1)}$ with entries $y_{(u,v)}$, such that

$$y_{u,i} = \begin{cases} 1 \text{ if } C(u) = i, \\ 0 \text{ otherwise.} \end{cases}$$

Cost Hamiltonian:

$$H_C = \frac{1}{2} \sum_{(u,v) \in E} \left((2 - \kappa) w_{(u,v)} + w_{(u,v)} \sum_{i \in K_{\min(\kappa,v)}} Z_{u,i} Z_{v,i} \right),$$

The mixing Hamiltonian:

$$H_M = \sum_{u \in V} \sum_{i,j \in R} X_{u,i} X_{u,j} + Y_{u,i} Y_{u,j},$$

The initial state:

$$|s\rangle = \bigotimes_{u \in V} |W_{\min(\kappa,u)}\rangle$$

6.1.3 Multi-level

We now consider a multi-level system consisting of κ levels. We define string $z \in K_\kappa^N$ consisting of variables $z_u \in K_\kappa$ such that $z_u = C(u)$. Our objective function is

$$\min_z \sum_{(u,v) \in E} \begin{cases} -w_{(u,v)} & \text{if } z_u = z_v \\ w_{(u,v)} & \text{otherwise} \end{cases} \tag{6.21}$$

Having access to a multi-level system, we can now encode individual qudit states with the cluster label $|0\rangle$ for cluster 0, $|1\rangle$ for cluster 1, ... and $|\kappa - 1\rangle$ for cluster $\kappa - 1$. This formulation does not come with constraints, and therefore does not have to be formulated

[2] We can actually use even one qubit less, since the first one is fixed in the one state.

using a penalty function.

We define a two-body interaction V_κ according to

$$V_\kappa = \sum_{i \neq j \in K_\kappa} |i_u\rangle |j_v\rangle \langle i_u| \langle j_v| - \sum_{i \in K_\kappa} |i_u\rangle |i_v\rangle \langle i_u| \langle i_v| , \tag{6.22}$$

which is defined by the following $\kappa^2 \times \kappa^2$ (unitary-)matrix:

$$V_\kappa = \begin{bmatrix} -1 & 0 & \dots & \dots & \dots & \dots & \dots & 0 \\ 0 & 1 & \ddots & \dots & \dots & \dots & \dots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \dots & \dots & \dots & \vdots \\ \vdots & \vdots & \ddots & -1 & \ddots & \dots & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 & \ddots & \dots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 & 1 \end{bmatrix} . \tag{6.23}$$

Hence, for two nodes we have that when they are in the same (different) cluster(s), we measure an eigenvalue of -1 (+1). Our full cost Hamiltonian can be obtained by summing over all nodes including the weight between those nodes,

$$H_C = \sum_{(u,v)} w_{(u,v)} V_\kappa \tag{6.24}$$

For both cases we need a suitable mixing Hamiltonian that can handle qudits. An example of this is given in the work by Hadfield et al [54], where the following single-qudit mixing Hamiltonian is proposed:

$$h_M(r) = \sum_{i=1}^{r} \left((\Sigma^x)^i + (\Sigma^{x\dagger})^i \right) , \tag{6.25}$$

where Σ^x is the generalised Pauli X-operator given by

$$\Sigma^x = \begin{bmatrix} 0 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix} . \tag{6.26}$$

One observes that for $r = 1$, the single-qudit mixer is therefore given by

$$h_M(r=1) = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 1 \\ 1 & 0 & 1 & \dots & 0 & 0 \\ 0 & 1 & 0 & \ddots & 0 & 0 \\ \vdots & 0 & \ddots & \ddots & 0 & \vdots \\ 0 & \vdots & \ddots & \ddots & 0 & 1 \\ 1 & 0 & \dots & 0 & 1 & 0 \end{bmatrix} , \tag{6.27}$$

such that every level is connected to its nearest neighbour, including periodic boundary conditions. Increasing r increases the connectivity by adding more off-diagonal terms. The full mixing Hamiltonian can be written as

$$H_M = \sum_{u \in V} h_M. \tag{6.28}$$

We can pick any value of $r \in \{1, \ldots, \kappa - 1\}$, where the special cases at the boundary are called the *single-qudit ring mixer* for $r = 1$ and the *fully-connected mixer* for $r = \kappa - 1$, similar to what we had in our on-hot encoding.

We take the superposition of all qudit computational basic states as our initial state, i.e.

$$|s\rangle = |+^{\kappa}\rangle^{\otimes N} = \frac{1}{\sqrt{\kappa^N}} \sum_{z \in K_{\kappa}^N} |z\rangle \tag{6.29}$$

The following box summarises the multi-level encoding:

Formulation 6.4. Multi-level (ML)
Parameters: κ: the maximum amount of clusters. r: parameter describing the connectivity of the mixer.

Domain: Bit strings $z \in \{0, \ldots, \kappa - 1\}^N$ with entries z_u, such that

$$\min_z \sum_{(u,v) \in E} \begin{cases} -w_{u,v} & \text{if } z_u = z_v \\ w_{(u,v)} & \text{otherwise} \end{cases}$$

Cost Hamiltonian:

$$H_C = \sum_{(u,v)} w_{(u,v)} V_{\kappa},$$

where V_{κ} is given by (6.22).

The mixing Hamiltonian:

$$H_M = \sum_{u \in V} \sum_{i=1}^{r} \left((\Sigma^x)^i + (\Sigma^{x\dagger})^i \right),$$

The initial state:

$$|s\rangle = |+^{\kappa}\rangle^{\otimes N}$$

Reducing the state space?

In principle, we can apply the same trick we used for the one-hot encoding to reduce the total state space—this would mean that different qudits have access to different levels. However,

the question is whether this is possible in practice. For the one-hot formulation the physical system does not have to change (since we only use two-level physical systems), but in this case we need systems with different quantum levels, which seems less reasonable from a Hardware perspective. Therefore, we do not make this reduction in the multi-level formulation.

6.2 Circuit compilations, complexities and an example

In this section we will analyse the circuits and their complexities for three different formulations Λ: EB, OH and ML. In any complexity analysis throughout this section we will always consider some complete graph $G = (V, E)$, with arbitrary edge weights $w_{(u,v)}, (u, v) \in E$ and $N = |V|$. Only complete graphs are considered because results for their circuit complexity upper bound the results for any other graph with the same amount of nodes. OHr is not considered as it is upper-bounded by OH.

Additionally, we will also consider the implementation for a trivial example to visualise the compilation of the QAOA algorithm onto an actual quantum circuit when $p = 1$.

Example 6.1. We consider a correlation clustering problem instance on graph $G = (V, E)$ with $V = \{0, 1, 2\}$, $E = \{(0, 1), (0, 2), (1, 2)\}$ and edge weights $\{w_{(0,1)} = -1, w_{(0,2)} = 1, w_{(1,2)} = -1\}$, as depicted in Figure 6.3. The total solution space consists of $3^3 = 27$ solutions and the set of optimal solutions is

$$\{(0, 1, 0), (0, 2, 0), (1, 0, 1), (1, 2, 1), (2, 0, 2), (2, 1, 2)\},$$

with an optimal number of agreements corresponding to 3. Random guessing finds an optimal solution with probability $P = 0.222$ and has an expected approximation ratio (which is the ratio of the expected objective function value and the optimal objective function value) of $r = \frac{13}{27} \approx 0.481$.

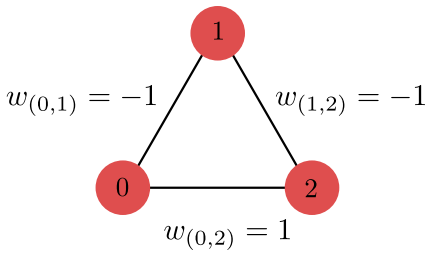


Figure 6.3: Correlation clustering example that we will use throughout this section. We have three nodes and three edges and the optimal solution can be obtained by putting node 1 in a different cluster than node 0 and 2, which are placed in a single cluster.

After all gate complexities and the quantum circuits for example 6.1 have been established, we will show the optimisation landscapes for all considered formulations in this example and give the best expected approximation ratio of all formulations when $p = 1$. From a practical point of view, the actual circuit depth might be more relevant that the total gate complexity—that is why we end this section with numerical results on the depths for all considered formalisms at different values of p.

6.2.1 Edge-based

Consider the edge-based formulation as described in EB. We start with $|E|$ qubits initialised in the zero state. To create the uniform superposition we apply a Hadamard gate H to every qubit. The unitary corresponding to the cost Hamiltonian itself is very easy to implement, requiring $|E|$ Pauli-Z rotation gates. However, the added penalty term complicates matters: we have a sum of one-body, two-body and three-body terms. Fortunately, all terms commute since our cost Hamiltonian is diagonal by construction, and can therefore be easily implemented. Figure 6.4 and 6.5 show how the two-body and three-body Pauli Z-terms can be implemented using only CNOT and $R_Z(2\theta)$ operations.

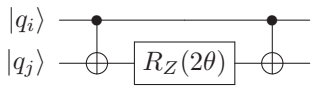


Figure 6.4: Quantum circuit performing the operation $U = \exp\{-i\theta Z_i Z_j\}$

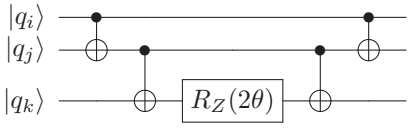


Figure 6.5: Quantum circuit performing the operation $U = \exp\{-i\theta Z_i Z_j Z_k\}$. [119]

This means that for every constraint we need 7 R_Z-gates and 10 CNOT-gates. A complete graph has $\binom{N}{3} = N(N-1)(N-2)/6$ triangles, thus the amount of constraints also scales as $O(N^3)$. The X-mixer can be implemented by $|E|$ R_X-gates. Putting everything together, the EB formulation requires a $|E|$ qudit-system and $O(N^3)$ gate operations.

In order to solve example 6.1 we choose our penalty parameter $\lambda = 3$, such that the best possible solution in the non-feasible subspace has a larger objective function value than the best possible solution in the feasible subspace. However, there is still overlap between the feasible subspace and the non-feasible subspace, which hopefully helps in smoothing out the optimisation landscape. The circuit implementation is shown in Figure 6.6:

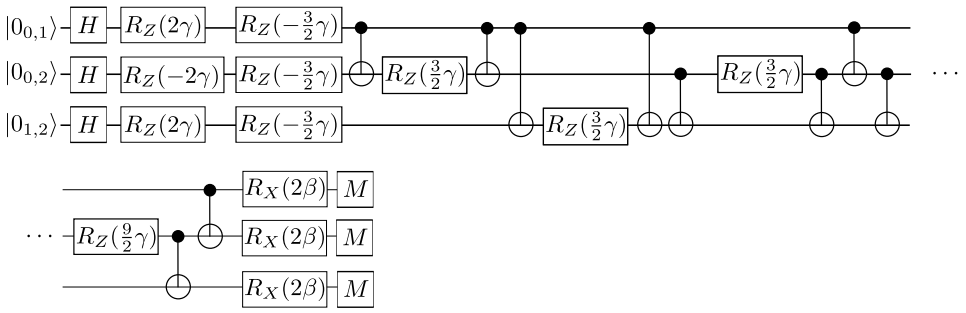


Figure 6.6: Quantum circuit to solve example 6.1 in the edge-based formulation.

6.2.2 One-hot

In the OH formulation, we have to set $\kappa = N$ as N upper bounds to maximum amount of clusters that could be needed. We start with a total of N^2-qubits in the all-zero state again. In the first step, we want to establish the in a superposition of all feasible states: the previously defined generalised W-state (6.19). Let us first define a new gate operation, called the *odds gate* which is defined as

$$M(p:q) = \sqrt{\frac{1}{p+q}} \begin{bmatrix} \sqrt{p} & \sqrt{q} \\ -\sqrt{q} & \sqrt{p} \end{bmatrix}. \tag{6.30}$$

We can then implement the generalised $W-$state for a single node u with just (controlled) odds gates and (controlled) X-gates. An example for $\kappa = 5$ is given in Figure 6.7:

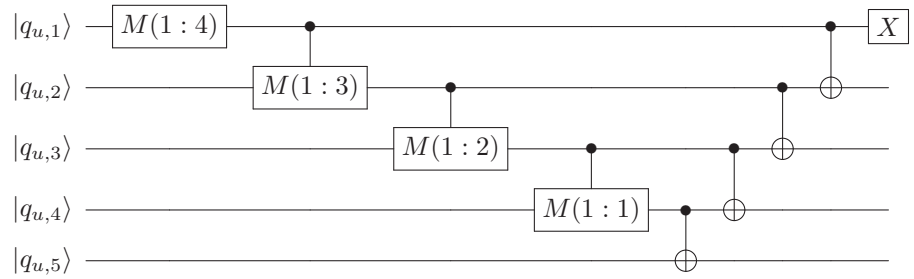


Figure 6.7: Quantum circuit implementing the generalised W-state for one node with 5 possible clusters.

Since odds gates are not 'native' gates, they must in practice be approximated in terms of elementary gates: this adds a factor $\log(1/\epsilon)$ to the complexity, where ϵ is the desired precision of the gate. In total, we require N odds gates, $N(N-2)$ controlled odds, $N(N-1)$ CNOT's and N Pauli X operations to set up our initial state.

Our cost unitary consists solely of two-body ZZ-terms (the other terms only add a global phase), that can be implemented using the circuit in Figure 6.4. Since they are applied on every edge for every cluster number, we have a total of $N^2(N-1)/2$ of these terms.

Finally, we have the XY-mixing operation to maintain our feasible subspace. Every XY-mixer on single node u that allows for transitions between cluster i and cluster j can be implemented using the circuit in Figure 6.8, requiring two CNOT's and one controlled $R_X(2\theta)$-operation.

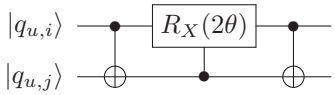


Figure 6.8: Quantum circuit for the XY-mixer on a single node u that allows for transitions between cluster i and cluster j.

The connectivity of the mixing operations to its neighbours sets the total amount of gates that are needed. Considering a fully connected mixer, which has shown to have

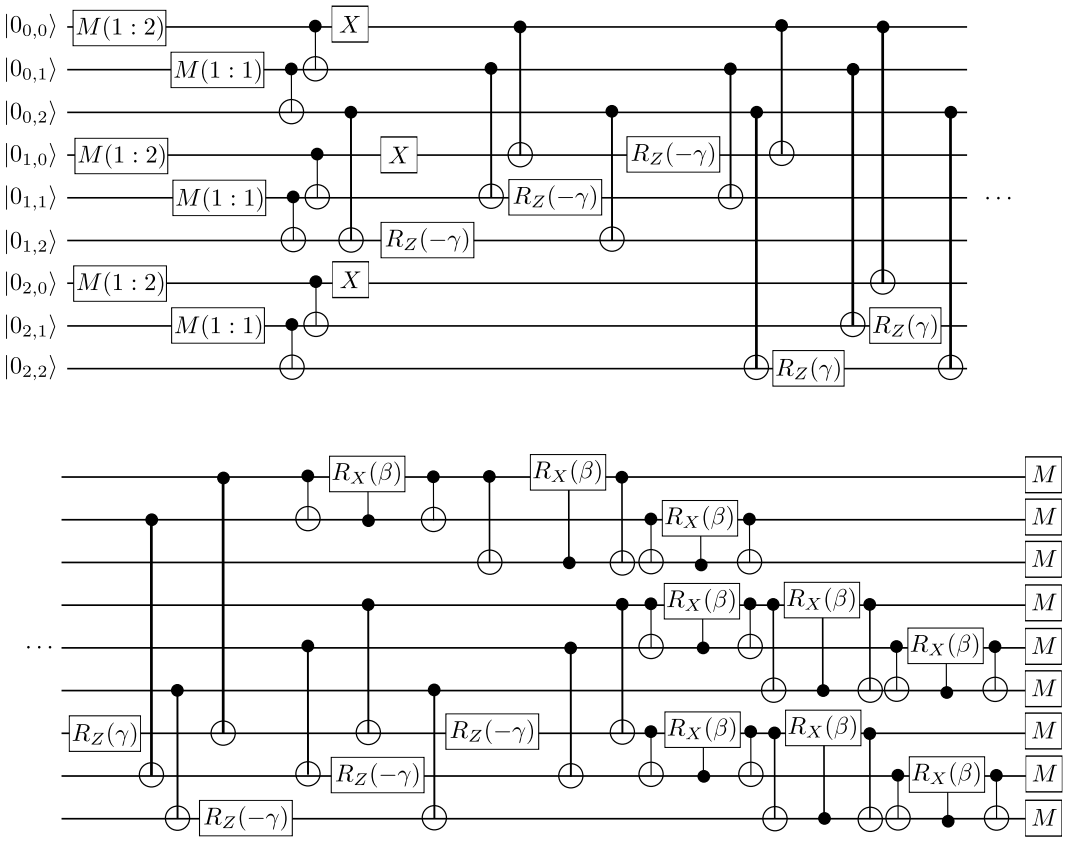


Figure 6.9: Quantum circuit in the OH formulation that encodes our correlation clustering example 6.1.

the best performance by Wang et al. [74], we need a total of $N^2(N-1)/2$ mixing operations. As a result, we end up with a N^2 qudit-system and $O(N^3)$ gate operations. To implement the circuit that solves example 6.1, we need 9 qubits as is shown in Figure 6.9.

6.2.3 Multi-level

We define an N qudit quantum system of dimension $\kappa = N$, such that the computational basis can be written as $|0\rangle, |1\rangle, \ldots, |N^N - 1\rangle$. All qudits are initialised in the all-zero state. We first consider a generalised Hadamard transform H^N, which implements the following transform

$$H^N |j\rangle = |0\rangle + e^{2\pi i 0.j} |1\rangle + \cdots + e^{2(N-1)\pi i 0.j} |N-1\rangle, \tag{6.31}$$

that can be represented by the following matrix:

$$H^N = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ 1 & e^{2\pi i 0.1} & \ldots & e^{2\pi i 0.(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{2(N-1)\pi i 0.1} & \ldots & e^{(N-1)\pi i 0.(N-1)} \end{bmatrix}. \tag{6.32}$$

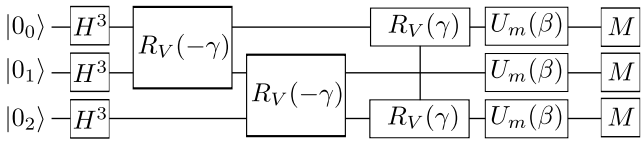


Figure 6.10: Quantum circuit to solve example 6.1 in the multi-level formulation (6.4).

We will assume that our cost Hamiltonian unitary can be implemented as a single two-qudit gate, which needs to be applied on every edge. Therefore, a total amount of $|E| = N(N-1)/2$ operations are needed. For the mixing, we have the ring-mixer that works very much like the XY-mixer we defined for the OH formulation. Considering the single-qudit ring mixer ($r = 1$) as an elementary qudit gate, we would need just N of these operations. More connectivity ($r > 1$) would increase the complexity in a similar way as increasing the connectivity of the XY-mixer would. To sum it up, we need N qudits with N levels and the total amount of qudit gates scales as $O(N^2)$. As always, Figure 6.10 shows the circuit implementation of example 6.1.

6.2.4 Optimisation landscapes for our example

The optimisation landscapes as a function of (β, γ) corresponding to our three formulations for example 6.1 are given in Figure 6.11. Every data points is generated by taking 1000 samples from the QAOA state-vector and therefore shows slight irregularities due to the sampling noise, as would also be the case in the actual experiment.

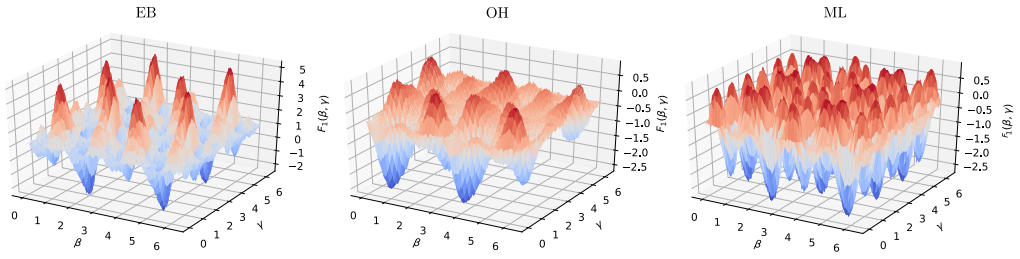


Figure 6.11: Optimisation landscapes corresponding to the expectation value of the cost Hamiltonian. 1000 measurements were taken from the QAOA state, parametrised in β, γ.

The one-hot and multi-level formulations are expected to have similar landscapes, since they would be isomorphic if the connectivity of the mixer is identical. In the MIN-(DISAGREE-AGREE)-objective, which was used for the Hamiltonian, we have that the maximum and minimum values of the objective function values are -3 and 1, respectively. Therefore, these values should bound the optimisation landscapes of the OH and ML formulations at all parameter values. Figure 6.11 shows that this is indeed the case for those two formulations, and one can also observe that all formulations do not attain the minimum value of -3 for any β, γ. The edge-based formulation has some peaks that do exceed the maximum of the objective, which correspond to parameter combinations that have a high probability amplitude for states that violate the constraint.

Λ	(γ^*, β^*)	$F_1(\gamma^*, \beta^*)$	r
EB	(0.8251, 2.094)	-2.377	0.90
OH	(0.7616, 0.6347)	-2.63	0.94
ML	(3.808, 0.6347)	-2.70	0.95

Table 6.1: Optimal results for our example for different formulations, obtained by brute-force search. The way the approximation ratio r is defined is given in the next section (See (6.33)). For now it is only important to know the definition given in Chapter 3.

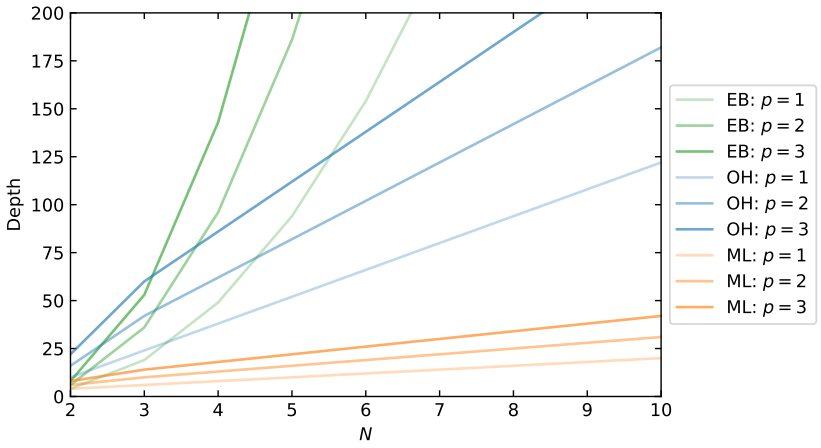


Figure 6.12: Circuit depth as a function of problem size N for different values of p in each formulation Λ.

If we perform a brute-force search over the optimisation landscapes to find the best β, γ in each formulation, we find the following optimal points, corresponding function values and approximation ratios as displayed in Table 6.1. Hence, we have that for example 6.1 all formulations have values of β, γ such that they have much better expected approximation ratios than we could get by random guessing ($r = 0.481$). Note that at $(\beta, \gamma) = (0, 0)$ all our QAOA unitaries become identity operations, and for those that use a superposition of all feasible states we retrieve the same performance as a random guessing algorithm. For the best found β, γ, we observe that the best approximation ratios are given by the ML and OH formulation with the EB formulation only slightly behind.

6.2.5 Circuit depths and scalability

So far we looked only at the total amount of gates but in practice a lot of gate operations can be parallelised. Qiskit by default, and Cirq when forced to, creates an optimal schedule of operations such that the total circuit depth is minimised. A plot of the circuit depth as a function of N for several values of p is given in Figure 6.12. We note that for both the OH and ML formulation we have that the depth scales linearly with N, whilst for the edge-based formulation it scales quadratically. Note that for OHr the scaling will be the same as OH except for a different pre-factor.

Since the amount of qubits and depth of the circuit is our most important benchmark in simulation, we can conclude that circuit complexity-wise the OH (and in particular OHr) and the ML formulation would be preferred over the EB formulation. However, in our analysis we did assume that the odds gate and qudit gates are directly implementable, whilst all gates used in the EB formulation are generally considered to be elementary gates. However, the order of scaling of the depth is the most important here; since by the Solovay-Kitaev theorem every arbitrary gate can be approximated up to an exponentially small error costing only a polynomial overhead.

6.3 Performance: numerical results

The goal of the following section is to do experiments investigating the performance of the established Hamiltonian formulations, by solving small correlation clustering instances for which the quantum processor can be simulated on a classical computer. In these simulations, we will not consider the OH formulation. This is because the amount of qubits is the dominant factor in our computation times as well as the memory requirements, and for OH we would not be able to solve the largest considered instances on individual CPU cores. Appendix C verifies whether we can expect any major performance differences by applying this reduction in qubits.

6.3.1 Experimental method and procedure

We will benchmark the performance on a data set of complete graphs of N nodes, where $N \in \{3, \ldots, 7\}$. For every N, we have sampled 50 graphs from the set of all possible weighted complete graphs with $w_{(u,v)} \in \{-1, 1\}$. The edge weight creation probabilities are P and $P - 1$ for $w_{(u,v)} = 1$ and $w_{(u,v)} = -1$, respectively. To create our data sets, we sweep P from 0 to 1 across all 50 instances, such that we have a good representation for the entire set of possible graphs.

As a performance measure to the solution quality, we will only consider the approximation ratio. Since QAOA is a stochastic algorithm, we need to use the expectation value instead of the best-found string:

$$r = \frac{\langle C \rangle}{OPT}, \tag{6.33}$$

where $\langle C \rangle$ is the expectation value of the objective function and OPT the optimal value to the correlation clustering problem. Since any we can classically simulate the full quantum state-vector, the state space is always small enough to adopt a brute-force method to determine the optimal value to the correlation clustering problem.

For the performance, we are interested in the objective function values corresponding to the MAX-AGREE formulation. Therefore, we first need to transform the objectives we used to construct our cost Hamiltonian, which are in MIN-(DISAGREE-AGREE). We define the number of agreements a and disagreements d. Note that we must always have that for every edge the solution either agrees or disagrees with the weight, and therefore we have that $a + d = |E|$. The expectation value of the cost Hamiltonian in the MIN-(DISAGREE-AGREE) formulation is then $\langle H_C \rangle = d - a$, so we can calculate the

expectation value in the MAX-AGREE setting $\langle C \rangle = a$ as

$$\langle C \rangle = \frac{1}{2}(|E| - \langle H_C \rangle). \tag{6.34}$$

Defining the weighted adjacency matrix of a correlation clustering instance as w, and the Hamiltonian formulation to be used as Λ we can summarise the adopted QAOA procedure as *QAOA-CC*. The time complexity of QAOA-CC, is

Algorithm 1: QAOA-CC

Input: w, p, s, Λ, classical optimiser
Output: $F_p(\vec{\beta}, \vec{\gamma})$

```
initialise β⃗, γ⃗ randomly
while not stopping criteria do
    for i = 1, 2, ··· , s do
        Run the QAOA quantum circuit in formulation Λ with the current β⃗, γ⃗
        Sample bit string x from the state vector by measurement
    Estimate F_p(β⃗, γ⃗) from the s collected samples
    Optimisation step of β⃗, γ⃗
```

$$O(ospd_\Lambda(N)), \tag{6.35}$$

where o is the amount of calls the classical optimiser makes to the quantum processor, s is the amount of samples it takes to estimate the energy of the QAOA state in the cost Hamiltonian, p is the QAOA depth and $d_\Lambda(N)$ the depth of the quantum circuit for formulation Λ. For $\Lambda = \text{ML}$ and $\Lambda = \text{OH}$, $d_\Lambda(N)$ scales as $O(N)$ and for $\Lambda = \text{EB}$ this is $O(N^2)$.

An important remark is that in practice, the output of QAOA-CC would actually be the best bit string x that was observed along the way. Alternative performance measurements, as for example the probability of observing a string x that has an approximation ratio above some threshold value, can then be adopted.

The QAOA quantum circuit will be simulated using Qiskit and Cirq for qubit and qudit systems, respectively. The COBYLA implementation with off-the-shelve hyperparameter settings—included in the SciPy-package [91]—will be used as our classical optimiser. The stopping criteria is set by the maximum budget (500 iterations) or the (off-the-shelf) tolerance for termination. We will always set the total amount of samples taken from the state-vector of the quantum computer to be $s = 1000$.

When $\Lambda = \text{EB}$, the value of the penalty parameter is always taken as $\lambda = 2|E| + 1$, such that separation between energies of feasible and infeasible states is guaranteed.

6.3.2 Simulation results

Figures 6.13, 6.14 and 6.15 show the main results on the complete graph data sets for the considered formulations Λ. Let us start by summarising our four main observations:

- The average performance is similar for different formulations.

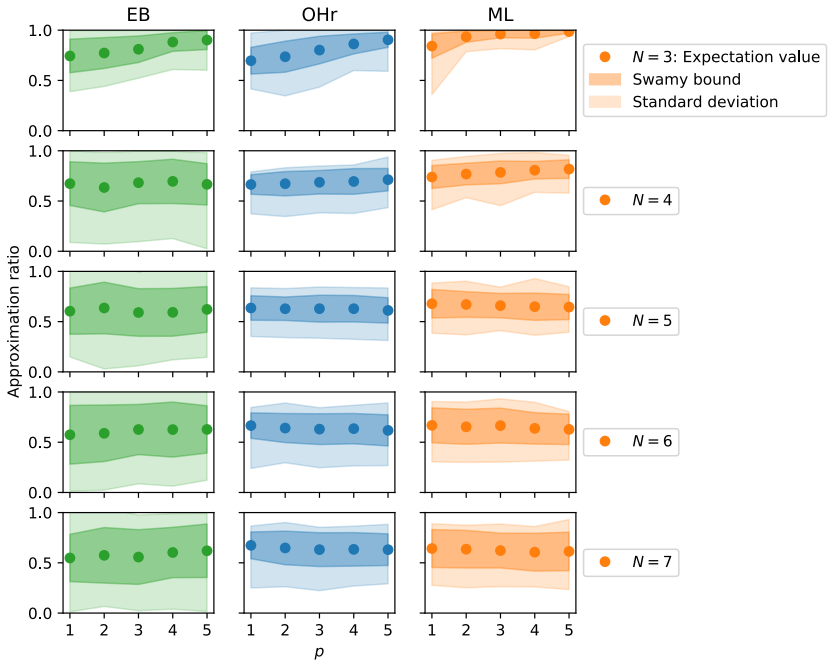


Figure 6.13: Numerical results for three different Hamiltonian formulations Λ in solving 50 random instances of complete graphs with N nodes. The expectation values, standard deviation and maximum and minimum performance results for each data set are indicated by shaded areas, which are made continuous to increase the readability.

- The standard deviation in the performance over different instances is large.
- There is no considerable difference in the difficulty experienced for different formulations for different type of instances.
- Increasing p only marginally increases the performance, if not worsens it in some cases.

Starting with the first point, this is even better illustrated in Figure 6.14, which plots the average approximation ratio as a function of the number of nodes N. For small problem sizes the ML formulation performs better than EB and EB better than OH, but this slightly changes as we move to larger N. At $N = 7$ we even have that OH outperforms ML.

The standard deviation is very large for all three formulations. This is even more so for the edge-based formulation, for which we encounter much larger extremes: for all N the algorithm is at every p able to solve some instances exactly (the approximation ratio equals one, see the maximum range of EB in Figure 6.13) but sometimes also fails terribly (we even have cases where the approximation ratio approaches zero)—this happens when the algorithm is stuck around parameter values corresponding to constraint violations. This is also reflected in Figure 6.14 (right), which shows the worst case performances. In all formulations, the large standard deviation is created by

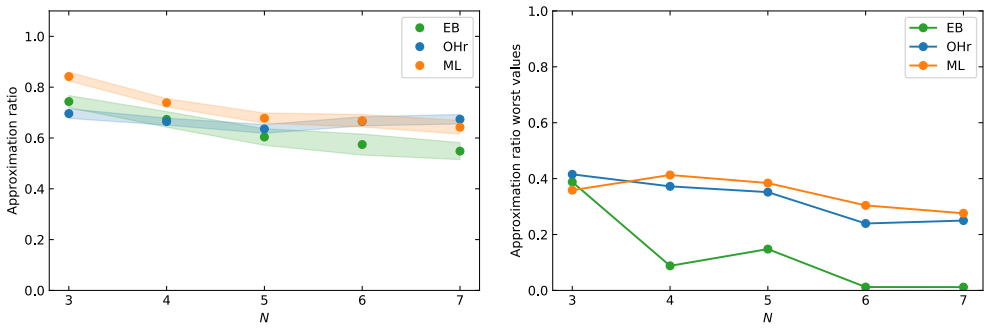


Figure 6.14: Left: Average approximation ratio at $p = 1$ as a function of problem size N. The shaded area represents the error in the mean at every discrete point N. Right: worst case approximation ratio found for different Λ when $p = 1$. In both graphs, we have added a form of artificial continuity to the plots in order to improve the readability.

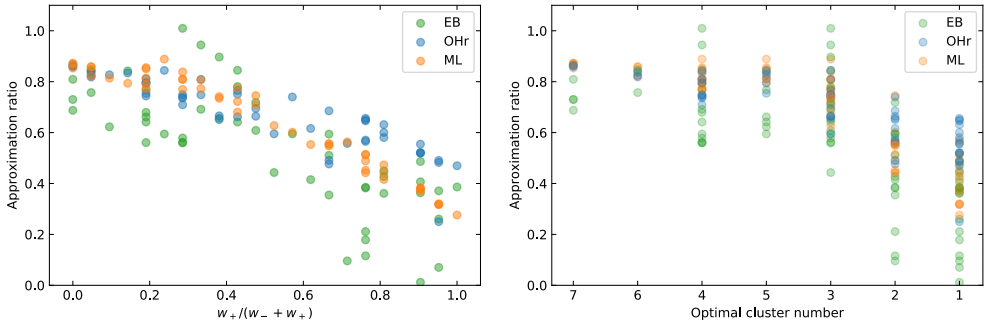


Figure 6.15: Left: Approximation ratios obtained for individual instances in the different formulations Λ, plotted as a function of the ratio of positive weights to the total amounts of weights. Right: Approximation ratios obtained for individual instances as a function of the optimal cluster number that, found using a brute-force method. When an instance has multiple possible optimal cluster numbers, the data point is plotted for every value.

1. A lack of robustness with the current classical optimisation method, which is very dependent on the quality of the initial point.
2. The difference in performance for different instances, as can be seen in the both plots in Figure 6.15.

The third point can be deducted from the data shown in Figure 6.15. The left picture shows how the performance for single instances is related to the ratio of positive weights to the total amount of edge weights. We note that for all three formulations the performance generally decreases as the amount of positive weights increases. Similar behaviour is observed for the optimal cluster number, which is shown in Figure 6.15 (right). This makes sense, as both are correlated: the more edge weights are $+1$ the more nodes are excepted to want to be put in a similar cluster, which decreases the optimal total number of clusters one would expect.

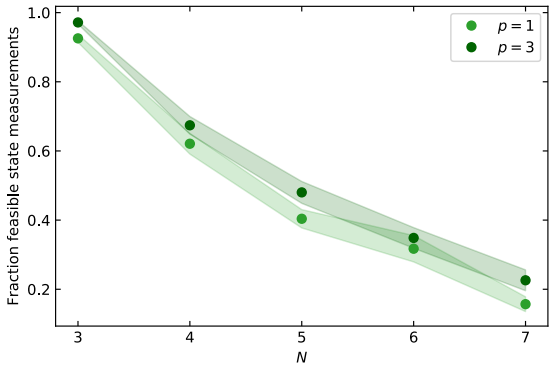


Figure 6.16: Average fraction of feasible strings over the total amount of string samples that are measured from an optimised QAOA state, when $\lambda = 2|E| + 1$. The shaded area represents the error in the mean.

And finally, regarding the lack of substantial performance improvement for higher p in all cases we note that the relative performance of the classical optimiser is the main bottleneck: as p increases the difficulty of the optimisation problem increases, and without good initial points the optimiser is unable to find a better (or even equally good) solution. In fact, the total parameter space increases exponentially with p, hence one would also have to increase the number of random points exponentially in order to expect a similar relative performance. However, we have restricted ourselves to just a single random initial point.

6.3.3 The downsides of penalty functions

Since we showed that in the EB formulation no efficient mixing operation can be constructed, we adopted a penalty function approach. Since we use all strings, feasible and non-feasible, in determining the energy over which we optimise, the optimisation landscape values are partially constructed from the contributions of states that include unfeasible solutions. Since the number of constraints grows as (N^3), these contributions become more profound as the problem size increases: there are more possible constraints to violate, each with their own penalties to the energy. Therefore, one can expect that the optimisation landscapes become increasingly complicated as N increases. This can explain why we observe such a large variance in the approximation ratios we find.

Another effect is that the amount of measurements that correspond to feasible solutions decreases as the amount of constraints increases. This is shown in Figure 6.16. The average fraction decays very fast in N, our results even show that the average fraction for both $p = 1$ and $p = 3$ is already smaller than 0.3 for $N = 7$. Since $N = 7$ is still a very small instance, this shows that is very unlikely that this approach will be able to solve any realistically-sized problem instances.

In these results we did not study how different values of λ effect the performance. The study by Wang et al. [74] shows how it is possible to optimise over different values of λ, increasing the approximation ratios. However, this does introduce yet another parameter to

be optimised over, and even with this additional optimisation step included Wang et al. make the case that penalty function approaches are still far inferior to formulations that encode the feasible subspace through the initial state and mixer.

6.4 The verdict

After all work so far, we can draw some preliminary conclusions to help us move forward:

- Performance-wise, the ML and OH formulation show better results than the edge-based formulation. This is in particular the case for larger N, where the edge-based formulation even sometimes shows approximation ratios approaching zero at it is stuck around states that have very high probabilities of violating constraints. Even though the edge-based formulation can be further improved when another optimisation over λ is performed, we still have to deal with the $O(N^3)$ scaling of the amount of constraints that leads to a decrease of the fraction of feasible solution measurements.
- In terms of gate complexity measured in the required circuit depth, the scaling of the ML and OH (OHr) formulations ($O(N)$) are better than the EB formulation ($O(N^2)$).
- For all formulations considerable improvements have to be made in order to make the performance worthwhile. In particular, we need to overcome the large variation in performance between different problem instances and marginal (or no) performance increases with increasing p.

Due to its low depth and state-space dimension (relatively low simulation times), good average (but in particular worst-case) performance and link to the Rydberg hardware that is being developed at QuSoft, we will take the ML formulation to move forward in the next chapter. However, it must be noted that the one-hot formulation does show better scaling with N than the ML formulation, as its average (worst-case) performance even increased for $N \geq 5$ ($N \geq 6$).

CHAPTER 7

Improvements to the algorithm

In the previous chapter we proposed different Hamiltonian formulations that allow us to encode the correlation clustering problem in a QAOA formulation. In bench-marking the performance, we found that considerable improvements to standard procedure of QAOA-CC are needed in order to obtain good performance. Strategies to achieve such performance improvements are central to this chapter. In the first section, we will focus on the classical optimisation step. In the following section, we propose several heuristic strategies to further increase the performance. The strategies are inspired by the results of Chapter 6. Combining all improvements, we propose a new form of the QAOA algorithm called *QAOA-CC-improved*. We again perform a numerical study and show that QAOA-CC-improved greatly outperforms QAOA-CC. In the final section, we will derive a performance bound for QAOA-CC-improved on 3-regular graphs.

Throughout this chapter, we will always use the multi-level formulation as our encoding.

7.1 The choice of the classical optimiser

Most of the work on QAOA (and VQE) use optimisers from the SciPy library [91], such as quasi-newton BFGS, Nelder-Mead and COBYLA. However, using these optimisers with off-the-shelf hyper-parameter settings, initial experiments showed a considerable variation in the quality of points that were found. Preferring not to do hyper-parameter optimisation due to its daunting computation times (evaluation the objective function already consists a considerable amount of time), we decided to instead compare different classical optimisers using their off-the-shelf hyper-parameter settings.

A recent work by Lavrijsen et al. comes with a software package[1] containing state-of-the-art optimisation methods that specifically focus on noisy black box optimisation [92]. Even though noise is at this point of lesser importance for our study (we only introduce sampling noise), the algorithms from their package showed very good performance in zero and low noise simulations. In our study, we will consider two optimisers from SciPy (COBYLA and basin-hopping) as well as three optimisers from the package by Lavrijsen et al. (ImFil, BOBYQA and SnobFit).

[1]The name of the software package is *scikit-quant*. A python implementation of the software can be installed from https://pypi.org/project/scikit-quant/.

First, we briefly introduce all the optimisers that are to be tested. Next, we discuss the results of a numerical study that compares the optimisers on a single instance of a correlation clustering problem with $N = 4$.

7.1.1 Overview of used optimisers

Let us first briefly describe all optimisers that are considered in the bench-marking. For the reader interested in further reading on a specific method: the first line always contains a reference to the original work.

COBYLA Constrained Optimisation by Linear Approximation (COBYLA) is an optimisation method for (nonlinear) constrained problems where the derivative of the objective function is not known [120]. For an objective function with N design variables, the algorithm creates $N + 1$ vertices and evaluates the objective function and the constraints at vertices of a trust region. For the optimisation process it uses linear approximations of the objective and constraints. Since it is a trust region method, performance dependence on the quality of the initial points can be expected.

Basin-hopping Basin-hopping is a global optimisation technique that works in conjunction with some local optimiser [121]. Designed to mimic the natural process of energy minimisation of clusters of atoms, it works particularly well for problems with rugged energy landscapes. At every iteration the optimiser randomly perturbs its current coordinates, performs local optimisation, and accepts or rejects the new coordinates based on their objective function value. In our simulations, we used basin-hopping in conjunction with COBYLA.

ImFil ImFil, or Implicit Filtering, is specifically designed for problems with local minima caused by high-frequency, low amplitude noise and with an underlying global structure that is easily optimised [122]. The algorithm starts with initial clusters of points, determined nearly entirely by the problem boundaries. The objective function values of these clusters are evaluated, and for the minimum of these evaluations derivative estimation by first-order interpolation is used to determine the next clusters to be evaluated. Since the boundary predominately determines the initial points, ImFil is relatively insensitive to initial points, unless they are used to restrict the boundary of the optimisation problem.

BOBYQA Bound Optimisation by Quadratic Approximation, or BOBYQA, is an algorithm for bound constrained black-box optimisation problems [123]. It is a trust region method that builds a quadratic approximation at each iteration, based on interpolation points. New sample points are created in two ways: 1) a trust region step or 2) an alternative iterations step, where steps are vectors that are added to the current iterate. In the trust region step, this vector is chosen such that it minimises the quadratic model around the current iterate and lies in the trust region, as well as the bounds set by the optimisation problem. The alternative iteration step is used when the norm of the vector is so small that it would reduce the accuracy of the quadratic model, so instead a vector is chosen such that good linear dependence of the interpolation is ensured. The point with the best objective function value is kept throughout the optimisation process. Since BOBYQA is also a trust region method, similar to COBYLA, its performance is also expected to be sensitive to the quality of the initial point.

SnobFit SnobFit, or Stable Noisy Optimisation by Branch and Fit, is an algorithm for the optimisation of problems with noisy and expensive objective functions [124]. It iteratively selects sets of evaluation points balancing between global and local search, which makes the algorithm good at escaping local optima. Evaluation points are selected by a branching algorithm that recursively subdivides the search space, as well as by randomly generating points from unexplored regions in parameter space. Locally, SnobFit builds a quadratic model around the current best point and minimises it to select a new evaluation point. It also chooses local search points as approximate minimisers within a trust region defined by safeguarded nearest neighbours.

7.1.2 Numerical results

In our study, we will consider direct energy calculations from the state-vector as well as energy estimation from state-vector sampling, in order to see effects of sampling noise on the performance. We consider only one instance of correlation clustering for bench-marking: a complete graph with just 4 nodes and all edge weights '-1' such that the optimal solution corresponds to all nodes being put in different clusters. For $p \in \{1, \ldots, 5\}$, we generate 25 random initial points in the respective parameter space. We set the maximum budget—i.e. the number of calls to the simulated quantum computer—at 500. Since basin-hopping has two different budgets, one for the basin-hopping steps and one for its local optimiser, we set the maximum budget by using the number of evaluations the local optimiser (COBYLA) used in its individual run: the number of Basin-hopping steps is the rounded ratio of the budget over this number. The results for state-vector sampling ($s = 1000$ shots) and state-vector simulation are given in Figure 7.1.

First, we note that the performance is similar for state-vector sampling and state-vector simulation. For COBYLA and basin-hopping(/COBYLA) the performance degrades slightly more compared to other optimisers, but in general the sampling noise does not seem to effect the performance too much. BOBYQA outperforms all other optimisers, but does always use up all the budget.[2] Unfortunately, the scikit-quant optimisation package does not allow us to change the tolerance, which would allow for fairer comparison. As far as the ratio of performance to number of function evaluations is concerned, ImFil performs well.

Even though adopting BOBYQA already potentially results in a large performance increase compared to using COBYLA, which was used to obtain our initial results, we still observe that a large gap exists between the best value found and the average performance. This means that there is still a lot to be gained in the classical optimisation step, the main one being the usage of good initial points, followed by hyper-parameter optimisation. Good initial points will have a larger effect on the local optimisation methods (in particular COBYLA and BOBYQA) compared to the global optimisers (e.g. ImFil).

7.2 Heuristic strategies

To further enhance the performance of QAOA, we will now propose several heuristic strategies that are inspired from the analysis of the previous chapter as well as literature in (classical) optimisation. We start this section with an overview of all strategies. One of these strategies

[2]There is one data point in the state-vector simulation data set for which BOBYQA used only 85 iterations, but this is the only time this was ever observed. This could be a consequence of the matrix that BOBYQA uses becoming singular, which terminates the algorithm.

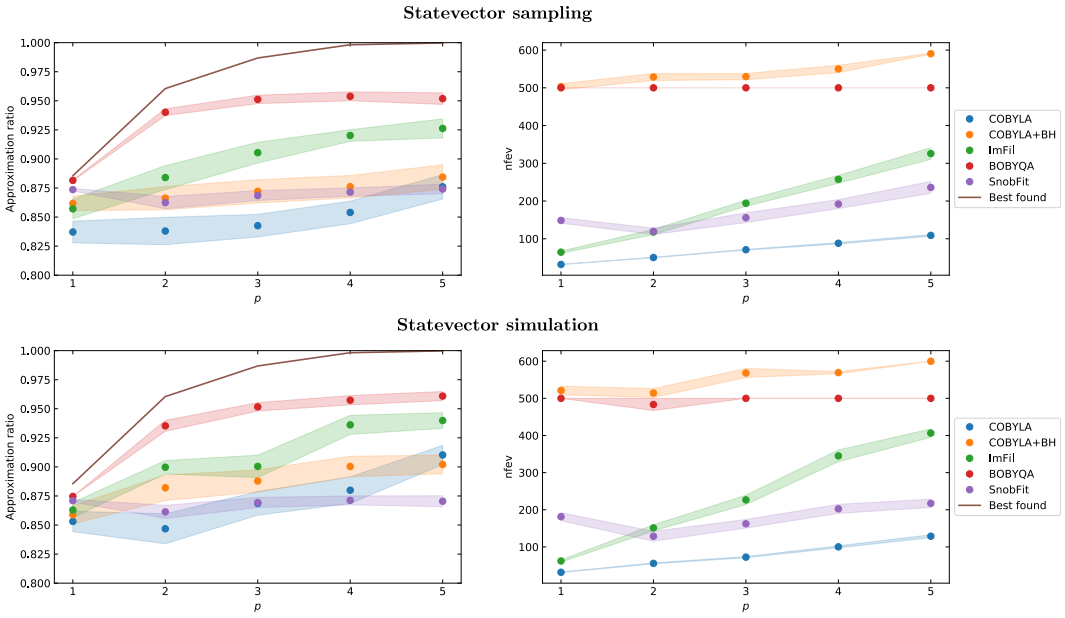


Figure 7.1: Top: State-vector sampling with 1000 measurements. Bottom: State-vector simulation. Left: found approximation ratios for 25 random points using different optimisers. Right: Number of function evaluations. The shaded area indicates the error in the mean, where the discrete points have been connected in order to improve the readability of the figure.

is the use of initial points, and in the second subsection we will give numerical evidence that strengthening that the case made by Brandao et al. [93] for MAXCUT also holds for problem: optimal values for different instances are concentrated in parameter space. We will analyse the individual and collective improvement(s) of these heuristic strategies on our $N = 4$ complete graph data set. Finally, an overview of strategies that were considered but are not included in the results is given.

7.2.1 Overview of used strategies

We propose the following heuristic strategies to improve our performance:

Restarts Local optimisers can greatly benefit from restarts, since they are sensitive to the quality of initial points. But even with fixed initial points, due to stochastic elements in both the optimisers procedure as well as the noise introduced from sampling (or gate errors), the algorithms can be made more robust by incorporating restarts. The work of Shaydulin et al. shows how multi-start methods can improve QAOA [47]. This adds a factor m, where m is the amount of (re)starts to the time complexity of the algorithm.

Optimised initial points Multiple works show how optimal parameters are concentrated in parameter space across instances that belong to 'similar classes' [93]. In practice, this means that we can select a single instance from a class, work very hard to find optimal parameters through variational optimisation or from analytical arguments, and use these values as initial points for the optimisation of other instances. In our improved algorithm, we will define some dictionary $\mathcal{D}$ in which initial points for all relevant instances are stored.

Even though it is not directly clear what the requirements are for instances to belong to a class with similar concentration, we expect this at least to hold for problems that share the same amount of nodes N and amount of clusters κ. We choose the correlation clustering instance of N nodes with all edge weights '-1' and use a large set of initial points to find optimal parameters for $\kappa \in \{2, \ldots, N\}$. We set the criteria that for each found initial point (β^*, γ^*) we must at least have that

$$F_{p+1,N,\kappa}(\beta^*, \gamma*) \geq F_{p,N,\kappa}(\beta^*, \gamma^*). \tag{7.1}$$

Of all 100 initial points we needed to find for $N = 3, \ldots, 7$ and $\kappa = 2, \ldots, N$, we were able to fulfil (7.1) for all but one case: $N = 6$, $\kappa = 5$, and $p = 6$. All initial points were stored in our dictionary $\mathcal{D}$.

As well as potentially increasing the performance compared to using random initial points, using optimised initial points can also decrease the total computation time QAOA as less quantum processor calls are needed for the optimiser. However, it is not clear how much time and effort is needed to find those initial points, so we will not take into account any benefits for the time complexity.

Optimised cluster number In the previous chapter we saw that the algorithm performed well on instances that required a large amount of clusters—i.e. when the optimal cluster number was close to the available amount of clusters κ—but not so when the amount of clusters was low. Since both the one-hot and multi-level methods allow for a variable κ, we can iterate the algorithm over the amount of the possible amount of clusters $\kappa = 1, \ldots, N$ and see for which number the algorithm gives back the best result [3]. The factor that upper bounds the added time complexity is N.

Formal definition of the improved algorithm

Incorporating all defined heuristic strategies and the existence of our initial points dictionary $\mathcal{D}$, we define QAOA-CC-improved as:

Algorithm 2: QAOA-CC-improved(Λ = ML)

Input: w, p, κ, $\mathcal{D}$, optimiser
Output: F^*_{p,κ^*}

```
for k = 1, ··· , N do
    initialise β⃗_κ, γ⃗_κ from D
    for r = 1, ··· , R do
        run QAOA-CC with Λ = ML, where κ = k
        if F_{p,k} < F*_{p,κ*} then
            update F*_{p,κ*} ← F_{p,k}, κ_* ← k
```

The time complexity of QAOA-CC-improved is simply that of QAOA-CC multiplied by the number of restarts m and the number of nodes N, to account for the loop over the different

[3] This is also possible on actual hardware: for one-hot this simply means that not all qubits are used and for a multi-level approach some hardware types, as for example with the Rydberg neutral atoms, allow for only a limited amount of states to be used.

amount of clusters:[4]

$$O(osmpN^2). \tag{7.2}$$

In practice, it is very likely that we can adopt some stop criterion for the loop over these clusters. Intuitively, one expects that the QAOA state energy is a convex function of the amount of clusters. If this is the case, one can stop the loop once we observe that the energy becomes worse as the cluster number increases. However, this is beyond the scope of this work and therefore we will always loop over the maximum amount of clusters that are needed to solve the problem instance.

7.2.2 Initial points study for $p = 1$

Whilst is evident that restarts and looping over different amount of clusters can only make the algorithm perform better, since using no restarts and only the maximum of clusters restores the original procedure of QAOA-CC, it might not be evident how fixed initial points might help when solving across different instances. In this subsection, we will give numerical evidence supporting the conclusions of the work by Brandao et al. [93]: initial points for different instances across different problem classes are concentrated. Our first intuition was that the same amount of nodes and maximum cluster number would define different problem classes, but we will show that even across these different classes we have concentration of initial points.

Concentration across instances

To investigate how concentration of good points at $p = 1$ across instances manifests itself for our problem we run the following experiment: we start with initial points we obtained from the all-negative-weights graphs for $N = 4$ and $N = 5$ and solve for all 50 instances in our correlation clustering data set, using COBYLA as an optimiser. The resulting optimal points are plotted in Figure 7.2:

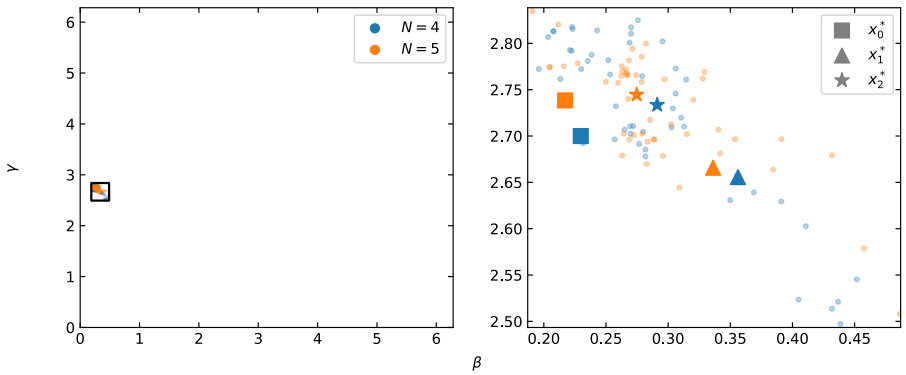


Figure 7.2: Left: locations of obtained optimal points over the entire possible parameter space. Right: close-up to the smallest possible square area that contains all optimal points. x_0^* indicates the used initial point, x_1^* is the point with the smallest maximum distance to other points and x_2^* the point with the smallest average distance to all other points.

[4] To get actual computation time estimates one needs to take actual hardware considerations into account, in particular the compiled gate operations and their latencies.

Note how all points are in the neighbourhood of our initial points. In fact, the smallest rectangular area containing all points (indicated by the black rectangular) takes about 0.2% of the entire possible parameter space. The plot on the right zooms in on this rectangular, showing how the optimal points are located relative to each other. In this plot x_0 is the used initial point, which is for both $N = 4$ and $N = 5$ positioned at the boundary of the collection of points. This makes sense due to the structure of the graph it belongs to—the all-negative-weights graph is itself an extreme case of the correlation clustering problem (requiring all clusters to be used). This also indicates that other graphs might be better suitable as initial points.

Somewhat unexpectedly, we also observed that the points for $N = 4$ and $N = 5$ are in the same neighbourhood. In the next section we run the same experiment but for a similar graph in different N and κ.

Concentration across classes

In our initial points data we observed that many initial points for different N and κ seemed to be similar, but sometimes with an added factor of π. Therefore, we used similar initial points as a starting points and verified whether optimal points for other problem sizes would exist in their neighbourhood as well. We always used the all-negative-weights graph as the correlation instance to solve, as this structure is easy to transfer between different problem sizes Figure 7.3 shows the location of the obtained points for different N and κ in parameter space ($p = 1$).

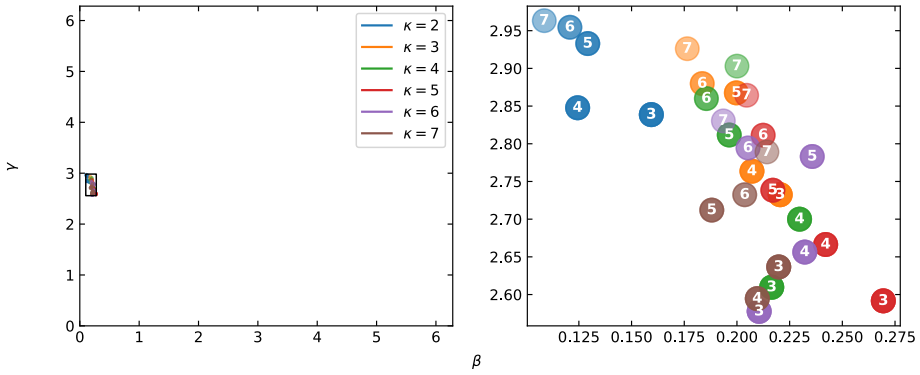


Figure 7.3: Left: locations of obtained optimal points over the entire possible parameter space. Right: close-up to the smallest possible square area that contains all optimal points. The number inside the point indicates the number of nodes.

We observe that $\kappa = 2$ is somewhat of an outsider, but again all points fall in a rectangular area encompassing (coincidentally) about 0.2% of the entire possible parameter space. Looking at the right plot of Figure 7.3, we observe that there is somewhat of a trend visible: in most cases, when N goes up γ increases and β goes down. However, this does not hold in all cases, and the data is generally too scattered to say something definitive. This could be due to the sampling noise introduced in the algorithm—a point in the neighbourhood of the optimal

point has a probability of resulting in a higher estimated energy. In order to verify whether a model can be build to predict good initial points, it would be better to use state-vector simulation. However, this greatly increases the simulation runtime for larger N.

7.2.3 Numerical results improvements

Figure 7.4 shows the numerical results on the $N = 4$ data set of complete graphs. We used COBYLA as the optimiser, since it potentially better illustrates the individual improvements compared to an optimiser that already performs better by itself. We note that looping over cluster numbers has the largest contribution to the improvement, followed by the optimised initial points which works particularly well for intermediate values of p. For small values of p the optimiser is able to find good points even with bad initial points and for larger values of p the distance from the initial point and the optimal point becomes larger in parameter space, which results in the initial point effectively becoming random again. The restarts are expected to have less of an effect when used in conjunction with good initial points.

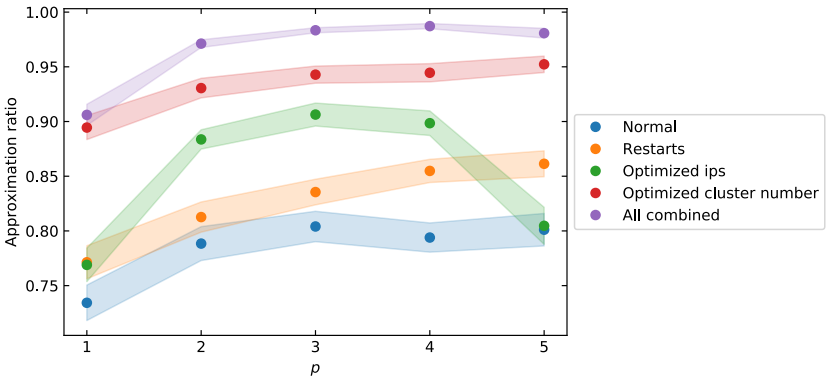


Figure 7.4: Approximation ratios for different improvements added to the QAOA algorithm, used on the $N = 4$ complete graph data set. The used optimiser is COBYLA. The dots indicate the average value over all instances and the shaded area represents the error in the mean.

7.2.4 Other considered strategies

There are two other strategies we tested for improving the algorithm, that were not included in the final results as they did not lead to improved performance. We will call the first *block-wise optimisation* and the second *block-wise initial point optimisation*. In block-wise (initial point) optimisation, one optimises for $p = 1$ to find (γ_1^*, β_1^*), and fixes (uses) these (as initial points) when optimising for $p = 2$. This process is repeated for higher values of p. Fixing these parameters showed considerable worse performance than the normal procedure: for larger p the performance only increased by a very small margin. Using these parameter values as initial points instead of fixing them resulted in a performance on par with the normal procedure, hence the points obtained by this procedure could be considered as random as any randomly generated point. A better strategy would have been to adopt the INTERP or FOURIER method as proposed by Zhou et al. [61]. An explanation of these methods can be found in Chapter 5. However, since we are primarily interested in good performance at

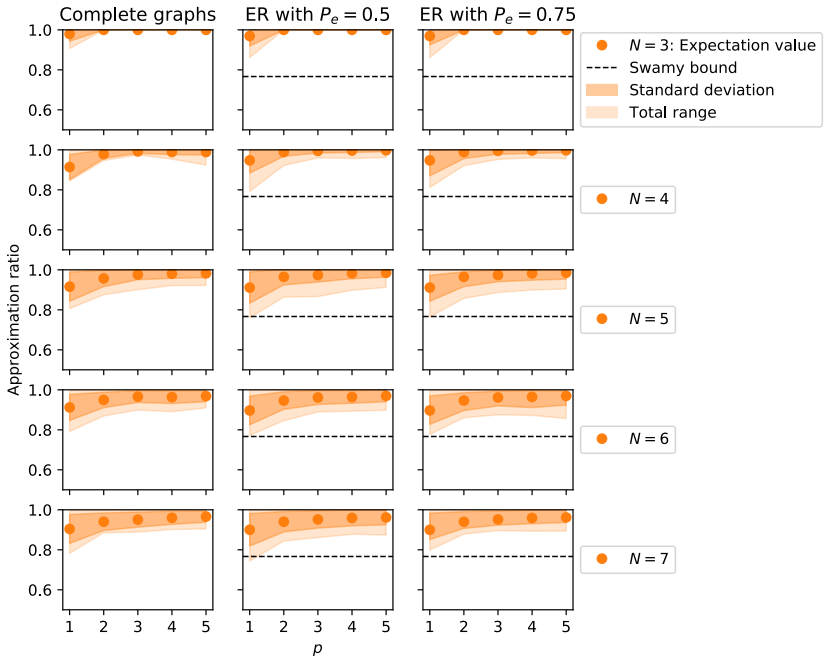


Figure 7.5: Results of QAOA-CC-improved on the data sets of complete graphs and Erdős–Rényi graphs with different edge creation probabilities P_e.

low values of p, these methods only become practically relevant once the performance has significant scaling in p.

7.3 Numerical results and analysis

Next to the already created complete graphs data sets, we will now consider an additional data set consisting of Erdős–Rényi graphs. In the creation of the Erdős–Rényi data set, we considered edge creation probabilities of 0.5 and 0.75. Similar to what we did for the complete graphs data set, we swept the edge weight probability P_e of giving an edge weight '+1' from 0 to 1 over the 50 random instances. Additionally, we set the criteria that every problem needed to have at least one edge, since the $N = 3$ data set in particular has a non-negligible probability for this to happen. Figure 7.5 shows the results for QAOA-CC-improved on all data sets.

Looking back at the observations we made for QAOA-CC in the previous chapter, we see that we have improved upon all remarks: the overall performance is increased, standard deviation is decreased and we generally have that performance increases with higher p. However, we note that the increase in performance diminishes as the approximation ratio becomes better: the difference between $p = 1$ and $p = 2$ is for example much larger than for $p = 3$ and $p = 4$.

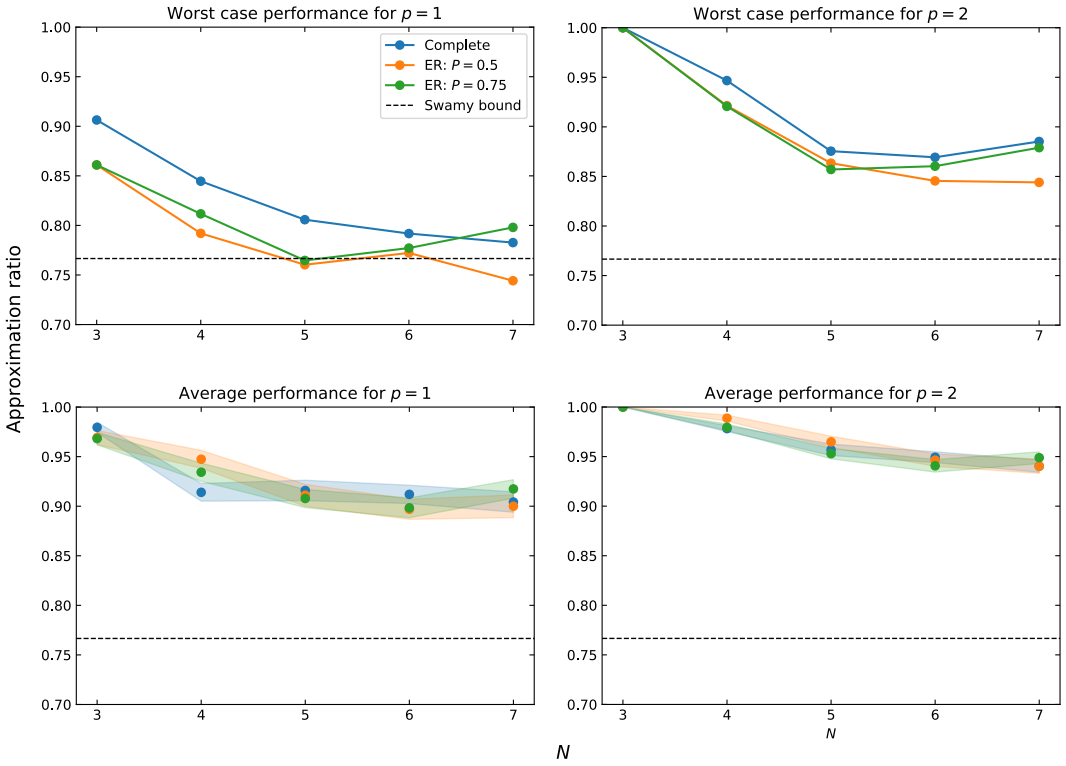


Figure 7.6: Performances plotted as a function of the amount of nodes N for the different data sets. Upper: worst case performances on all data sets. Lower: average performances. Left: results for $p = 1$. Right: results for $p = 2$. As always, any artificial continuity is added to improve readability.

Figure 7.6 shows the performance as a function of the problem size N for different graph types, giving an indication of the scalability. We observe that the algorithm performs slightly better in the worst case for complete graphs compared to the more general Erdős–Rényi graphs. This is to be expected, as it is generally assumed that quantum algorithms need some kind of internal problem structure to work well (this is also the case for classical algorithms). For average performance the performance is comparable amongst different graph types.

For $p = 1$ we observe that the worst case performance on instances from our bench-marking data sets drops slightly below the Swamy bound of 0.7666, but for $p = 2$ we have that the algorithm performs better than this bound on all instances in the data set. This statement is rather weak in the sense that all it provides is an indication of the performance, as instances outside our data set might exist for which the algorithm performs worse. Ways to overcome this are provided in the final chapter. Additionally, in the next section of this chapter will derive a performance bound for QAOA-CC-improved on 3-regular graphs.

7.3.1 Performance dependence on problem instance

Figure 7.7 shows scatter plots of the individual performance on instances in the data set of $N = 7$ for different values of p.

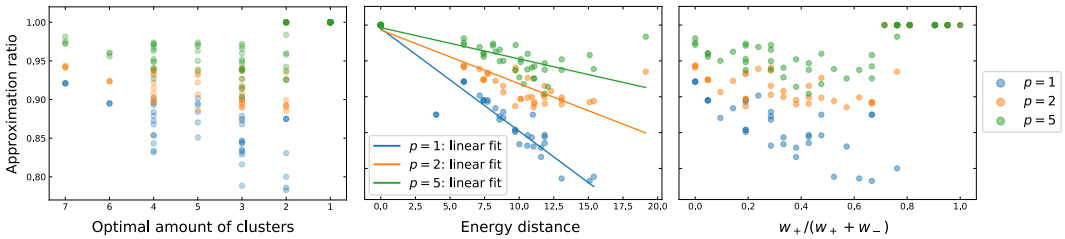


Figure 7.7: Different scatter plots of performance on the $N = 7$ complete graph data set as a function of left: the optimal amount of clusters, middle: the energy distance of the initial state with respect to the optimal solution and right: the ratio of positive weights to the total amount of weights.

We observe that the algorithm at low p has the most difficulty with instances that require a low amount of clusters (except the singleton cluster case, which is trivial for $\kappa = 1$), and as p increases the most difficult instances seem to have optimal cluster numbers in the middle of the singleton and all-different clusters.

The energy distance is defined as the difference between the function value of the optimal string and the expectation value of the cost Hamiltonian in the used initial state, corresponding to the final cluster number that was used. We observe a negative linear relationship between the energy distance and the performance. This fortifies the idea that the quality of the initial state is very important, as well as giving some idea of what kind of problems might suitable to QAOA in general: problems that change drastically on a micro-level but are still close in energy distance might still be solvable in QAOA as long as you can prepare a very good initial quantum state for some starting energy.

The optimal cluster number and the ratio of $w_+/(w_+ + w_-)$ are correlated[5], and both show a similar relation with the performance. This indicates that we can further increase the performance of the algorithm by tailoring some aspects to some properties of the correlation clustering instance, as has so far only been done by looping over the cluster number.

7.4 Performance bound on 3-regular graphs

The numerical study shows promising results for QAOA-CC-improved. However, proven lower bounds on the performance would allow us to make much stronger statements about its performance. In general, performance bounds for QAOA are difficult to derive, in particular for $p > 1$, which is why almost the entire body of literature relies on numerical studies. A recent work by Wurtz and Love [125] showed a derivation of lower bounds for QAOA at

[5]This also holds for the energy distance, but this is depends on the way the initial state is generated.

$p = 1$ and $p = 2$ (and a conjectured result on $p = 3$) on MAXCUT, and their techniques form the inspiration for this section.

We will consider correlation clustering problems described by 3-regular weighted graphs. The goal of this section is to find a lower bound at for the approximation ratio r at $p = 1$ for QAOA-CC-improved, defined as

$$r = \max_{\kappa} \frac{F_\kappa(\gamma, \beta)}{C_{\max}}, \tag{7.3}$$

where $F_\kappa(\gamma, \beta)$ is the QAOA energy of the algorithm with at most κ clusters and $C_{\max}$ the maximum amount of agreements for the correlation clustering instance. Since our graphs have degree 3, we need at most 4 clusters to solve the problem. Hence, our algorithm loops over maximum cluster values $\kappa \in \{1, 2, 3, 4\}$. We will show that for 3-regular graphs $G = (V, E)$, where G is not the complete $N = 4$ graph, initial points $\beta^*_\kappa, \gamma^*_\kappa$ exist such that QAOA-CC-improved gives a 0.670-approximation algorithm, even without the classical optimisation loop.

7.4.1 Problem setup and a lower bound for the energy

Consider some arbitrary 3-regular graph $\mathcal{G}$ of $N(\mathcal{G})$ nodes that is not the complete graph with $N = 4$. We identify for each edge $\langle i, j \rangle$ the sub-graph $\mathcal{G}^p_{<i,j>}$ of all edges and vertices within p steps of node i and j. At $p = 1$ there are only three possible kinds of sub-graph structures as indicated in Figure 7.8:

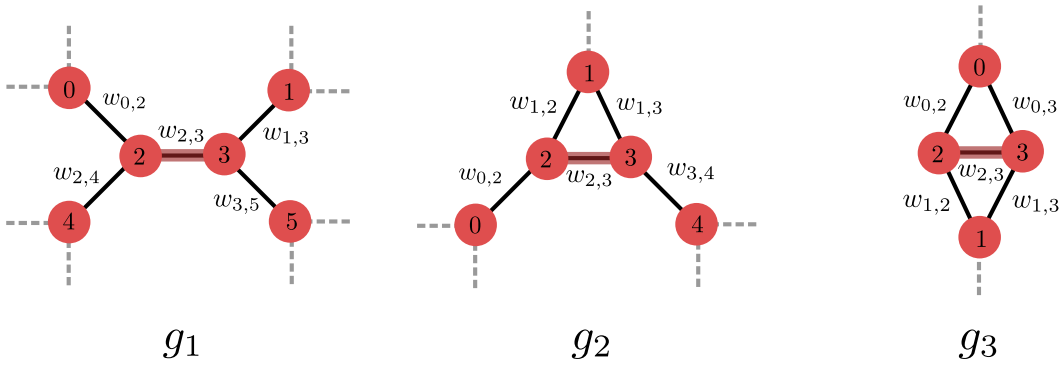


Figure 7.8: The 3 types of sub-graphs for $p = 1$. The sub-graphs form the environment of the highlighted edge, and note how only neighbouring edges are included in the sub-graph. The dotted edges indicate edges outside of the sub-graph.

Since all sub-graph types have 5 edges, we have a total of $3 \cdot 2^5 = 96$ possible sub-graphs when we include weights $w_{u,v} \in \{-1, +1\}$. However, by symmetry arguments we can reduce the total amount of weighted sub-graphs we have to consider. We define three sets of sub-graphs $g_i, i \in \{1, 2, 3\}$, representing all 3-regular sub-graph structures with 6, 5 and 4 nodes respectively (see Figure 7.8), such that for every sub-graph $\lambda \in g_i$ there exists no other graph $\lambda' \in g_i$ that is equivalent in the QAOA setting. Our total set of possible sub-graphs is then $S = g_1 \cup g_2 \cup g_3$.

We now decompose our graph $\mathcal{G}$ into sub-graph environments $\lambda \in S$ for which the

multiplicity of each λ is $N_\lambda(\mathcal{G})$. Since we have such a sub-graph environment for every edge, we must have that the sum over all $N_\lambda(\mathcal{G})$ is equal to the total amount of edges $3N(\mathcal{G})/2$. For a single edge $\langle u, v\rangle$ with sub-graph environment λ, denoted as $\langle u, v\rangle \to \lambda$, the contribution to the energy is given by

$$f_{\kappa,\langle u,v\rangle\to\lambda}(\beta_\kappa, \gamma_\kappa) = \langle \beta_\kappa, \gamma_\kappa | \, w_{\langle u,v\rangle} V_\kappa \, | \beta_\kappa, \gamma_\kappa \rangle \,, \tag{7.4}$$

where V_kappa is given by (6.22) and $|\beta_\kappa, \gamma_\kappa\rangle$ by (4.8) in the ML formulation. Note how (7.4) only contains terms that operate within λ. The expectation value of the algorithm with a maximum of k clusters at $p = 1$ for some $\beta_\kappa, \gamma_\kappa$ is given by

$$F_\kappa(\beta_\kappa, \gamma_\kappa) = \sum_\lambda N_\lambda(\mathcal{G}) f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa). \tag{7.5}$$

We must also have that this is always smaller than or equal to the expectation value using the best values of $\beta_\kappa, \gamma_\kappa$

$$F_{\kappa,\max} = \max_{\gamma_\kappa,\beta_\kappa} F_\kappa(\gamma_\kappa, \beta_\kappa) \geq \sum_\lambda N_\lambda(\mathcal{G}) f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa), \tag{7.6}$$

providing us with a lower bound on the energy of the algorithm $F_{\kappa,\max}$ at a given κ.

7.4.2 Upper bound on the amount of agreements

We are now faced with the task of finding an upper bound on the objective function value. A naive bound would be the total amount of edges, but we can do better by considering the same argument Wurtz and Love used to determine an upper bound for MAXCUT [125].

Consider the graph $\mathcal{H}$ which is a collection of $N_\lambda(\mathcal{G})$ disconnected sub-graphs $\mathcal{G}^p_{\langle i,j\rangle}$ for each edge in $\mathcal{G}$. Since the largest sub-graph has a total solution space of 6^4 possible solutions, a brute-force method can be used to find the ratio between the optimal objective function value and the number of edges for every sub-graph λ, which we will call c_λ. Since all sub-graphs are isolated, the global fraction of agreements to edges is equal to the average fraction over each sub-graph. However, we have several edges belonging to different disjoint sub-graphs in $\mathcal{H}$ that are actually the same edge in $\mathcal{G}$. In this case, we can have that for both sub-graphs a clustering exists for which the edge contributes to the objective value, but that the required clustering is different in both sub-graphs. A simplified example which illustrates this effect is to consider a 2-regular graph $\mathcal{I}$ with edge weights '+1' and '−1', decomposed into sub-graphs of single edges. In this setup, we have two possible sub-graphs corresponding to edge weights '+1' and '−1'. One easily verifies that $\mathcal{J}$, which consists of all disjoint edges, has agreements for all of its edges ($c = 1$) whilst for $\mathcal{I}$ the ratio of agreements to edges can be at most $c = 2/3$, as shown in Figure 7.9. As a result, we have that the objective function value of $\mathcal{G}$ is bounded from above by

$$C_{\max} \leq \sum_\lambda N_\lambda(\mathcal{G}) c_\lambda. \tag{7.7}$$

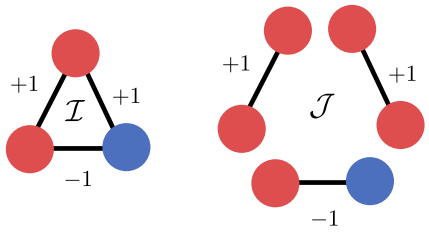


Figure 7.9: Example that illustrates how transforming graph $\mathcal{I}$ into a new graph $\mathcal{J}$, consisting of disjoint single-edge sub-graphs for every edge in $\mathcal{I}$, leads to a higher fraction of agreements per edge. The cycle of edges in $\mathcal{I}$ has two clauses that cannot be satisfied at the same time, whilst in $\mathcal{J}$ all clauses can be satisfied since all edges are disconnected.

7.4.3 Constructing the hardest graph

Since we have a lower bound for the energy and an upper bound for the objective function value, the approximation ratio r is bounded from below by

$$r_\kappa(\mathcal{G}) \geq \max_\kappa \frac{\sum_\lambda N_\lambda(\mathcal{G}) f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa)}{\sum_\lambda N_\lambda(\mathcal{G}) c_\lambda}. \tag{7.8}$$

The worst case approximation ratio is then given by the hardest graph $\mathcal{G} = \mathcal{G}^*$, which corresponds to some parameter combinations of $N_\lambda(\mathcal{G}^*) \in \{0, 1, 2, \dots\}$ for all sub-graphs λ. However, not all parameter combinations of $N_\lambda(\mathcal{G})$ correspond to valid graphs, as was already shown by Farhi et al. [6]. For now, we will only consider the structure of the graph and not take the feasibility of certain weight combinations into account. First, we note that for every edge in sub-graph g_3 there must be at least 4 edges that an environment of sub-graph g_2. Also, we have that the 'triangle' of g_2 and 'crossed square' of g_3 cannot share the same vertex, which means that the number of triangular edges and crossed square edges must be smaller than the amount of nodes $N(\mathcal{G})$. Our final constraints are that all $N_\lambda(\mathcal{G})$ are non-negative integers and must sum up to $3N(\mathcal{G})/2$. Defining $n_\lambda(\mathcal{G}) \equiv N_\lambda(\mathcal{G})/N(\mathcal{G})$ such that we can relax the integer constraint when considering the limit of large $N(\mathcal{G})$, we obtain the following *Minimax Linear Fractional Program* (MLFP):

$$\begin{aligned}
\min_{n_\lambda(\mathcal{G})} \quad & \max_\kappa \frac{\sum_\lambda n_\lambda(\mathcal{G}) f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa)}{\sum_\lambda n_\lambda(\mathcal{G}) c_\lambda} \\
\text{s.t.} \quad & \sum_{n_\lambda(\mathcal{G}) \in g_2} n_\lambda(\mathcal{G}) - 4 \sum_{n_\lambda(\mathcal{G}) \in g_3} n_\lambda(\mathcal{G}) \geq 0 \\
& - \sum_{n_\lambda(\mathcal{G}) \in g_2} n_\lambda(\mathcal{G}) \geq -1 \\
& \sum_{n_\lambda(\mathcal{G})} n_\lambda(\mathcal{G}) = \frac{3}{2} \\
& n_\lambda(\mathcal{G}) \geq 0 \text{ for all } \lambda
\end{aligned} \tag{7.9}$$

Equation 7.9 is in the form of a generalised fractional program, which is not reducible to a linear program (LP) which can be solved efficiently[6]. We choose as our primary solving

[6] There are linear relaxation bounding techniques that do allow for global optimisation up to some error ϵ. We have performed initial experiments with one of those techniques [126], but were not able to achieve desirable results so far due to the difficulties in approximating our objective function. We have also used a bisection algorithm that converges to local minima, but its performance was worse compared to COBYLA.

method a local constraint optimisation method with random initial points. Since this does not guarantee that we have obtained a global optimum, we also adopt alternative methods. Let us define a domain $\mathcal{C}$ such that $x \in \mathcal{C}$ when it satisfies the constraints of (7.9). To verify our numerical results, we will also compare it with two other measurements:

1. Take the number of edges as the upper bound instead of the fractional objectives (7.7), i.e. we let $c_\lambda \to 1$ for all λ. Under this relaxation we can write (7.9) as the following LP:

$$\begin{aligned} \min \quad & \alpha \\ \text{s.t.} \quad & \sum_\lambda n_\lambda(\mathcal{G}) f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa) \leq \frac{3}{2}\alpha \quad \text{for all } \kappa \\ & n_\lambda(\mathcal{G}) \in \mathcal{C}, \alpha \in \mathbb{R} \end{aligned} \tag{7.10}$$

   The solution of this LP is a lower bound to the actual bound, as sub-graphs that originally had $c_\lambda = 0.8$ now contribute too much to the upper bound of the optimal objective function value (7.7).

2. Similarly, we can also assume that only sub-graphs for which a perfect clustering exists contribute to the construction of the most difficult graph. Under this assumption, we define a new set $S' = \{\lambda | \lambda \in S, c_\lambda = 1\}$.

$$\begin{aligned} \min \quad & \alpha \\ \text{s.t.} \quad & \sum_\lambda n_\lambda(\mathcal{G}) f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa) \leq \frac{3}{2}\alpha \quad \text{for all } \kappa \\ & n_\lambda(\mathcal{G}) \in \mathcal{C}, \alpha \in \mathbb{R}, \lambda \in S' \end{aligned} \tag{7.11}$$

   Since $S' \subset S$, the optimal α^* provides an upper bound to the best value of r that we can find for $\lambda \in S$.

All LP's will be solved by a solver contained in the package 'lpsolvers' for Python.[7]

7.4.4 Iterative procedure for determining the bound

The mini-max optimisation problem gives us a method to determine the hardest instance $\mathcal{G}^*$, given that we fix the parameters $\beta_\kappa, \gamma_\kappa$. But how do we choose the values of these parameters? As one would normally do with QAOA, a classical optimisation loop can be adopted. First, we choose some initial values for $\beta_\kappa, \gamma_\kappa$ and calculate $f_{\kappa,\lambda}(\beta_\kappa, \gamma_\kappa)$ for all sub-graph environments λ. Next, we construct the hardest graph $\mathcal{G}^*$ by solving (7.9). In the next step we use the same objective function as in (7.9), but instead of minimising over n_λ whilst keeping the $\beta_\kappa, \gamma_\kappa$ fixed we now fix n_λ and try to maximise the objective over $\beta_\kappa, \gamma_\kappa$, i.e. we want to

$$\begin{aligned} \max_{\beta_\kappa,\gamma_\kappa} \quad & \max_\kappa \frac{\sum_\lambda n_\lambda(\mathcal{G}) f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa)}{\sum_\lambda n_\lambda(\mathcal{G}) c_\lambda} \\ \text{s.t.} \quad & \beta_\kappa, \gamma_\kappa \in [0, 2\pi) \text{ for all } \kappa, \end{aligned} \tag{7.12}$$

where we have used some cut-off $\epsilon = 10^{-7}$ to speed up the calculation as computing the values of $f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa)$ is expensive. However, there might exist some other graph $\mathcal{G}^{**}$ which is

[7] For documentation, see https://pypi.org/project/lpsolvers/.

more difficult for this particular $\beta_\kappa^*, \gamma_\kappa^*$, after which we can again try to find new parameters $\beta_\kappa^{**}, \gamma_\kappa^{**}$. This suggests the use of an iterative procedure for finding the best parameters. We do not know whether this procedure converges, but it doesn't necessarily have to: any combination of $\beta_\kappa, \gamma_\kappa$ has its own lower bound that holds specifically for these parameters, and can therefore be used as our result. Convergence would only suggest something about the tightness of this bound, guaranteed global convergence would mean that no better values of $\beta_\kappa, \gamma_\kappa$ exist.

Algorithm 3: Iterative procedure to obtain a lower bound

Data: λ, maximum number of iterations M
Input: initial $\beta_\kappa^0, \gamma_\kappa^0$,$M$
Output: optimal r^*, $\beta_\kappa^*, \gamma_\kappa^*$

```
for i = 1, 2, ··· , M do
    calculate f_{λ,κ}(β_κ^{i−1}, γ_κ^{i−1}) for all λ, κ
    Solve (7.9), store r^i and the collection of n_λ^i
    Solve (7.12) and store β_κ^i, γ_κ^i
```

7.4.5 The results

We set $\beta_\kappa = 0.15, \gamma_\kappa = 3$ for all $\kappa = 1, 2, 3, 4$. For 20 steps in our iterative procedure ($M = 20$), adopting COBYLA with 1000 restarts using random initial points, the best worst approximation ratio bound we found is

$$r \geq 0.670, \tag{7.13}$$

corresponding to the parameter combinations listed in Table 7.1. For these parameters, Table 7.2 shows the expected the edge contributions for every sub-graph that had a unique contribution (all double rows were deleted).

	$\kappa = 1$	$\kappa = 2$	$\kappa = 3$	$\kappa = 4$
β_κ^*	-	0.48326462	0.13104474	0.14350598
γ_κ^*	-	2.85741366	2.77318577	2.68161118

Table 7.1: Parameter values for different κ at which we were able to obtain performance guarantee (7.13). At $\kappa = 1$ the algorithm has only one state and hence no parameters.

For the same parameter combinations, solving (7.10) and (7.11) results in $r \geq 0.6367$ and $r \geq 0.670$, respectively. Therefore, we conclude that using the fractional objectives (7.7) instead of the amount of edges provides us with a much better bound under the assumption that (7.13) is (close to) optimal. This assumption is justified by the fact that in our results all λ that have non-negligible n_λ do have $c_\lambda = 1$, and because the best solution we obtained to (7.9) actually equals the upper bound set by (7.11).

For this particular combinations of sub-graphs belong to the hardest graph $\mathcal{G}^*$, the classical optimisation step actually does not make much of a difference: our results show that solving (7.12) with $\mathcal{G} = \mathcal{G}^*$ results in

$$\max_{\beta_\kappa, \gamma_\kappa} \quad \max_{\kappa} \frac{\sum_\lambda n_\lambda(\mathcal{G}^*) f_{\kappa,\lambda}(\gamma_\kappa, \beta_\kappa)}{\sum_\lambda n_\lambda(\mathcal{G}^*) c_\lambda} \approx 0.674,$$

which is only a small improvement on the obtained bound. This shows the quality of these values of $\beta_\kappa^*, \gamma_\kappa^*$ (as well as the hardness of the graph $\mathcal{G}^*$). Additionally, this also provides further evidence that QAOA, when having access to good initial points, can also be used without the classical optimisation step [93, 94].

7.4.6 Performance bounds for $p > 1$?

Unfortunately, we do not observe evidence for the existence of a trivial graph hierarchy as Wurtz and Love proved (and conjectured) for MAXCUT at $p \leq 2$ ($p > 2$) [125]. In fact, when we consider their large loop conjecture for our problem, we find that for our used $\beta_\kappa, \beta_\kappa$ we obtain a worst-case approximation ratio of $r = 0.693$, which is significantly larger than (7.13). Therefore, we do not conjecture the structure of the most difficult graph at any p, which would ease the determination of lower bounds at larger p. If we are to use the same method as we used for $p = 1$, we will have to consider of the order of $123 \cdot 2^{13} \approx 10^6$ different sub-graphs (not taking symmetries into account). We can reduce this number by exploiting symmetries, but since determining the energy of the largest sub-graph (14 nodes) is very computationally expensive we do not attempt to determine bounds for $p > 1$.[8]

[8] In fact, a back-of-the-envelope estimation to the amount of computing hours needed to execute such a computation—not including the symmetries—shows that we would need of the order of 10 million computing hours, which is far beyond the budget we have for SURFsara's LISA system.

Graph type	**Weights**	c	f_1	$f_2(\beta_2^*, \gamma_2^*)$	$f_3(\beta_3^*, \gamma_3^*)$	$f_4(\beta_4^*, \gamma_4^*)$
1	(-1,-1,-1,-1,-1)	1	0	0.69313	0.86386	0.90215
1	(-1,-1,-1,-1,+1)	1	0	0.69313	0.85916	0.9073
1	(-1,-1,-1,+1,+1)	1	0	0.69313	0.85894	0.90634
1	(-1,-1,+1,-1,-1)	1	1.0	0.69313	0.56993	0.44161
1	(-1,-1,+1,-1,+1)	1	1.0	0.69313	0.56611	0.44677
1	(-1,-1,+1,+1,+1)	1	1.0	0.69313	0.57151	0.45194
1	(-1,+1,-1,-1,+1)	1	0	0.69313	0.85603	0.90695
1	(-1,+1,-1,+1,+1)	1	0	0.69313	0.85659	0.90717
1	(-1,+1,+1,-1,+1)	1	1.0	0.69313	0.56668	0.44898
1	(-1,+1,+1,+1,+1)	1	1.0	0.69313	0.56921	0.44805
1	(+1,+1,-1,+1,+1)	1	0	0.69313	0.85609	0.90584
1	(+1,+1,+1,+1,+1)	1	1.0	0.69313	0.56823	0.44915
2	(-1,-1,-1,-1,-1)	1	0	0.64632	0.83114	0.88456
2	(-1,-1,-1,-1,+1)	1	0	0.64632	0.82549	0.88586
2	(-1,-1,-1,+1,-1)	1	1.0	0.73376	0.61476	0.48902
2	(-1,-1,-1,+1,+1)	1	1.0	0.73376	0.61665	0.48976
2	(-1,-1,+1,-1,-1)	1	0	0.73376	0.89578	0.9324
2	(-1,-1,+1,-1,+1)	1	0	0.73376	0.89705	0.93315
2	(-1,-1,+1,+1,-1)	4/5	1.0	0.64632	0.5218	0.41154
2	(-1,-1,+1,+1,+1)	4/5	1.0	0.64632	0.51983	0.40978
2	(-1,+1,-1,-1,+1)	1	0	0.73376	0.89618	0.93475
2	(-1,+1,-1,+1,+1)	4/5	1.0	0.64632	0.51996	0.40933
2	(-1,+1,+1,-1,-1)	4/5	0	0.64632	0.8111	0.86811
2	(-1,+1,+1,-1,+1)	4/5	0	0.64632	0.81213	0.8691
2	(-1,+1,+1,+1,-1)	1	1.0	0.73376	0.61684	0.48872
2	(-1,+1,+1,+1,+1)	1	1.0	0.73376	0.61625	0.4889
2	(+1,-1,-1,-1,+1)	1	0	0.64632	0.82679	0.88596
2	(+1,-1,-1,+1,+1)	1	1.0	0.73376	0.61844	0.48769
2	(+1,-1,-1,+1,+1)	1	0	0.73376	0.89699	0.93467
2	(+1,-1+1,+1,+1)	4/5	1.0	0.64632	0.52144	0.40756
2	(+1,+1+1,-1,+1)	4/5	0	0.64632	0.80994	0.86782
2	(+1,+1+1,+1,+1)	1	1.0	0.73376	0.6166	0.48941
3	(-1,-1,-1,-1,-1)	1	0	0.59828	0.79063	0.85454
3	(-1,-1,-1,-1,+1)	1	1.0	0.78104	0.66778	0.53755
3	(-1,-1,-1,+1,-1)	1	0	0.6901	0.8679	0.91845
3	(-1,-1,-1,+1,+1)	4/5	1.0	0.6901	0.56985	0.44932
3	(-1,-1,+1,+1,-1)	4/5	0	0.59828	0.77636	0.83979
3	(-1,-1,+1,+1,+1)	1	1.0	0.78104	0.66407	0.5299
3	(-1,+1,-1,+1,-1)	1	1.0	0.78104	0.93665	0.95846
3	(-1,+1,-1,+1,+1)	4/5	1.0	0.59828	0.46984	0.36661
3	(-1,+1,+1,-1,-1)	1	0	0.78104	0.93725	0.95799
3	(-1,+1,+1,-1,+1)	4/5	1.0	0.59828	0.47393	0.36901
3	(-1,+1,+1,+1,-1)	4/5	0	0.6901	0.85378	0.89673
3	(-1,+1,+1,+1,+1)	4/5	1.0	0.6901	0.56885	0.4488
3	(+1,+1,+1,+1,-1)	4/5	0	0.59828	0.7637	0.83102
3	(+1,+1,+1,+1,+1)	1	1.0	0.78104	0.66761	0.53522

Table 7.2: Numerical values for the edge contributions for different sub-graph environments λ corresponding to the parameter combinations $\beta_\kappa^*, \gamma_\kappa^*$ as listed in Table 7.1. Graphs with duplicate entries (i.e. that are identical under the QAOA setting) were left out of the table.

CHAPTER 8

Conclusions and future work

8.1 Conclusions

We have developed three main Hamiltonian formulations to solve correlation clustering with the quantum approximate optimisation algorithm: an edge-based and one-hot encoding designed for qubit systems and a multi-level encoding suitable for qudit hardware. We have shown that for the edge-based formulation any mixing operation that preserves the feasible subspace would have to be global, and therefore be very difficult to implement. Instead, we propose a penalty function approach that can be used to (soft-)encode the constraints. The one-hot encoding, also used in other works [74], can be reduced in qubit complexity from N^2 to $N(N+1)$.

We have shown that the total amount of elementary gate operations is $O(N^3)$ for the edge-based and one-hot formulation, and $O(N^2)$ for the multi-level formulation. However, operations can be parallelised, and we have found that the simulated circuit depths for both the one-hot and multi-level formulation scale with $O(N)$ and the edge-based formulation with $O(N^2)$, making the former two preferred over the latter for NISQ devices.

Numerical experiments for all Hamiltonian formulations in a standard QAOA setting (QAOA-CC) have shown that the average performance is somewhat similar for different Hamiltonian formulations, however the edge-based formulation has a larger range of approximation ratios and is the only formulation that for all studied problem sizes even at QAOA depth $p = 1$ is able to solve some instances exactly. However, it also has the worst worst-case performance and seems badly scalable due to the fact that the feasible subspace grows much slower than the total subspace as N increases. The variation in performance between different instances is large for all three Hamiltonian formulations. This is mainly due to the difference in performance on different graph types—in general we found that the more positive weights the problem instance has the worse the performance is. Finally, we did not observe that in these initial experiments increasing p resulted in improved performance.

Starting from the multi-level formulation we proposed several heuristic strategies to improve the performance of QAOA-CC, including restarts, improved initial points and the iteration over the cluster levels, leading to an improved algorithm called QAOA-CC-improved. We show that initial points are concentrated for different instances as well as amongst different problem sizes and maximum cluster numbers, justifying the practicability of a collection of good initial points.

Adopting QAOA-CC-improved to solve the instances in our data sets, consisting of complete as well as Erdős–Rényi graphs, we showed that the algorithms worst case performance on instances from our random data sets exceeded the Swamy bound for all simulated problem sizes for any $p \geq 2$. In addition, the way the worst case performance scales as a function of N seemed to slow down considerably for $N \geq 5$. Though it is hard to make too definitive statements as it contains only 5 data points, it does show promise to also be able to solve larger instances at low p.

Further evidence for this is provided by the results of a study to a performance guarantee, in which we showed that values of $\beta_\kappa, \gamma_\kappa$ exist such that QAOA-CC-improved has a performance guarantee of 0.670 on 3-regular graphs. For the hardest instance we could use classical optimisation to obtain an approximation ratio of 0.674, which is only slightly better than the performance bound that holds for all 3-regular graphs. This result, together with the results for the initial point study, provides strong evidence that with good initial points the classical optimisation step in QAOA becomes less important or can in fact even be eliminated.

8.2 Future work

For all our problem sizes we had that the minimum energy always corresponded to either the zero-clusters or maximum-amount of cluster numbers. The distribution of the amount of possible clusters is given by the uniform distribution of all possible initial states, and even though we saw that looping over this cluster number greatly improved the algorithm, there could still be room for improvement by generating better initial quantum states. Throughout the work on this thesis we had the idea of using the fractional solution of the semi-definite program in the classical algorithm as an initial state, and recent a publication showed promising results for implementing technique [81]. Another option would be to design the initial states ourselves by using information on the correlation clustering problem, as for example the ratio of positive to all weights. This also has the potential to serve as a substitute to the looping over clusters.

Our results showed the importance of having good initial points when using QAOA. A big open question is whether there are 'shortcuts' to come up with these initial points. For example, can we use relationships between optimal points for small instances to predict the locations of good initial points for larger instances? Or on a more fundamental level, can we analytically predict what makes a point optimal in the first place?

In our optimiser study we found that, when you don't have access to good initial points, the classical optimiser is a crucial element of QAOA: the difference between the best and worst optimiser could result in as much as a 0.1 absolute difference in the average approximation ratio. However, we observed that a considerable gap between the best-found approximation ratio and the best optimiser still exists: hyper-parameter optimisation as well as other strategies (besides the improved initial points) to improve the classical optimisers' performance could still be very useful in improving the obtained results.

So far, we were only able to derive a performance guarantee for $p = 1$ on 3-regular graphs. We saw that it would be computationally extremely expensive to use the same method to determine bounds for $p > 1$. This also holds for increasing the degree: the amount of possible sub-graphs increases considerably as well as the simulation times for these sub-

graphs. More research is needed to find better methods for deriving these performance bounds.

Alternatively, we can also do more at the numerical end: recent work by Sato, Yamada and Kashima provides us with an algorithm that uses machine learning to model the hard instance distribution of graph algorithm [127]. This would allow us to strengthen the results from numerical bench-marking, but we would still be restricted by only being able to study very small instances.

The biggest point that is not addressed in our research is the fact that real quantum hardware, at least until we have a general-purpose fault-tolerant quantum computer (if this will even be possible), will be noisy. Our current study only includes sampling noise but does assume perfect gate behaviour. Experiments on real quantum hardware, for example via the IBM cloud, or quantum simulations with introduced noise are needed to see the performance under more realistic conditions.

8.3 Outlook

The contents of this work started the efforts of creating a full-stack quantum computing story about implementing QAOA-CC-improved on a Rydberg quantum computer (see also Appendix D and Appendix E). Together with other researchers from QuSoft and the University of Amsterdam, we already started working on the development of realistic noise models and full implementation schemes, of which the results can hopefully be published soon. Most importantly, we hope to derive criteria on the noise levels under which the quantum computer is able to show performances that are competitive with the best classical algorithms. Whilst the actual Rydberg quantum computer that is being build is still too small (50 qudits with 10 levels) to solve practical problems—as for example realistic instances of pose estimation as we saw in Chapter 3—being able to experimentally show similar (or better) approximations than the best classical algorithms on small instances beyond simulation capabilities would be a large contribution to the field of quantum computing, and in particular QAOA.

Appendices

APPENDIX A

Alternative Hamiltonian formalisms

One-hot with a penalty function

The domain is equivalent to what we defined for the OH-formulation. We want to create a Hamiltonian that is above zero on any state that has variables placed in more than one cluster—this corresponds to violating the constraint. Using the approach by Wang et al. [74], we consider the two-body penalty term

$$f = \sum_{v}^{N} \left(1 - \sum_{i}^{\kappa} x_{v,i}\right)^2.$$

We convert this to a Hamiltonian formulation by letting $x_{v,i} \to (I - Z_{v,i})/2$ and expand the sum to obtain

$$\begin{aligned} H_{\text{constraints}} &= \sum_{v}^{N} \left(1 - \sum_{i}^{\kappa} \frac{I - Z_{v,i}}{2}\right)^2 \\ &= \frac{1}{4} \sum_{v}^{N} \left(2 - \kappa + \sum_{i}^{\kappa} Z_{v,i}\right)^2 \\ &= \frac{N}{4}(2-\kappa)^2 + \frac{1}{2} \sum_{v}^{N} \left((2-\kappa) \sum_{i}^{\kappa} Z_{v,i} + \frac{1}{2}(\sum_{i}^{\kappa} Z_{v,i})^2\right) \\ &= n\left(1 - \frac{\kappa}{2}\right)^2 + \frac{N\kappa}{4} + \frac{1}{2} \sum_{v}^{N} \left((2-\kappa) \sum_{i}^{\kappa} Z_{v,i} + \sum_{i<j}^{\kappa} Z_{v,i} Z_{v,j}\right) \end{aligned} \tag{A.1}$$

Just as we had for the EB-formulation, the cost Hamiltonian is given by:

$$H_C = H_{\text{problem}} + \lambda H_{\text{constraints}}, \tag{A.2}$$

where H_{problem} is given by Equation (6.13). The mixing operation is again the Pauli X-mixer:

$$H_M = \sum_{v}^{N} \sum_{v}^{\kappa} X_{v,i}.$$

Our initial state is the uniform superposition over all possible string combinations:

$$|s\rangle = |+\rangle^{\bigotimes \kappa N} = \frac{1}{\sqrt{2^{\kappa N}}} \sum_{x \in \{0,1\}^{\kappa N}} |x\rangle \,. \tag{A.3}$$

Multi-level using generalised Pauli operators

Define the generalised Pauli Z-operator as

$$\Sigma^z = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & \omega & 0 & \dots & 0 \\ 0 & 0 & \omega & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \omega^{\kappa-1} \end{bmatrix}, \tag{A.4}$$

where $\omega = \exp\{(2i\pi/\kappa)\}$. Following the approach by Pramanik et al. [49], our cost Hamiltonian is

$$H_C = -\sum_{u,v} \frac{w_{u,v}}{2} \left(\Sigma_u^z \Sigma_v^{z\dagger} + \Sigma_u^{z\dagger} \Sigma_v^z\right). \tag{A.5}$$

The complex conjugate is needed to make H_C Hermitian—all its eigenvalues have to be real in order to be observable. Since H_C is a diagonal matrix, its eigenvalues are the entries on this diagonal. Hence, using (A.5) we see that its eigenvalues are equal to

$$\begin{aligned} \mathrm{eig}(H_C, u, v) &= -\frac{w_{u,v}}{2} \left(e^{i2\pi u/\kappa} e^{-i2\pi v/\kappa} + e^{-i2\pi u/\kappa} e^{i2\pi v/\kappa}\right) \\ &= -w_{u,v} \cos\left(\frac{2\pi}{\kappa}(u-v)\right), \end{aligned} \tag{A.6}$$

where $u, v \in \{0, 1, \dots, \kappa - 1\}$ [1]. Hence, when $u = v$ the resulting eigenvalue is $-w_{u,v}$ and when $u \neq v$ we have that its eigenvalue is an element of $[-w_{u,v}, w_{u,v})$. This formulation comes with a potential disadvantage for higher values of κ: two different state combinations, which are, considering their contribution to the objective value, equivalent, might have two different energy values. Therefore, we will not use it in our numerical simulations, unless it proves to be easier to implement on actual hardware than ML.

This formulation uses the same mixer and initial state as ML given by Equations (6.28) and (6.29).

[1] This is similar to the Potts model known from physics.

APPENDIX B

Increased mixing in the multi-level formulation

In Chapter 6, we defined the mixing Hamiltonian that operates on a single qudit with κ levels as

$$h_M(r) = \sum_{i=1}^{r} \left((\Sigma^x)^i + (\Sigma^{x\dagger})^i \right).$$

We consider $\kappa = N = 5$, such that $r \in \{1, 2, 3, 4\}$. For these values of r, the matrix representation of the mixing Hamiltonian is given by

$$h_M(r=1) = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}, \qquad h_M(r=2) = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix},$$

$$h_M(r=3) = \begin{bmatrix} 0 & 2 & 1 & 1 & 2 \\ 2 & 0 & 2 & 1 & 1 \\ 1 & 2 & 0 & 2 & 1 \\ 1 & 1 & 2 & 0 & 2 \\ 2 & 1 & 1 & 2 & 0 \end{bmatrix}, \qquad h_M(r=4) = \begin{bmatrix} 0 & 2 & 2 & 2 & 2 \\ 2 & 0 & 2 & 2 & 2 \\ 2 & 2 & 0 & 2 & 2 \\ 2 & 2 & 2 & 0 & 2 \\ 2 & 2 & 2 & 2 & 0 \end{bmatrix}.$$

We see that for any $r > 1$, the mixing Hamiltonian becomes fully connected: it can transition to any other level at all levels. Wang et al. [74] already showed how increased mixing increases performance, and this is confirmed by our results as shown in Figure B.1 for the $N = 5$ complete graph data-set. Interestingly, we observe in Figure B.1(top) that for $p = 3$ that even though we have a complete mixer, the fact that the transition factors are different for different levels makes the performance only as good as the nearest-level mixer ($r = 1$). Figure B.1(bottom) shows that for the different mixers the same types of graphs are difficult to solve. However, the relative performance is somewhat better on the difficult graphs compared to the easier-to-solve graphs.

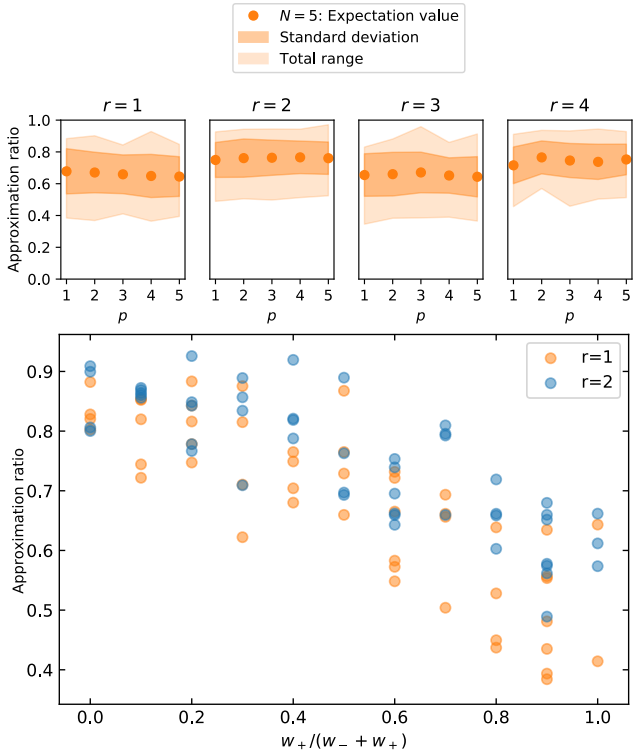


Figure B.1: Performance as a function of the mixing parameter r. The results are for the $N = 5$ complete data-set and were obtained by using COBYLA as an optimiser.

APPENDIX C

OH versus OHr

The reduction in the amount of qubits we used for our one-hot encoding changes the Ansatz, and as a result one would expect that the optimisation landscape changes accordingly. In order to verify whether this does not considerably change the performance, we compare the results for both formalisms on our $N = 4$ complete graph data set using QAOA-CC.

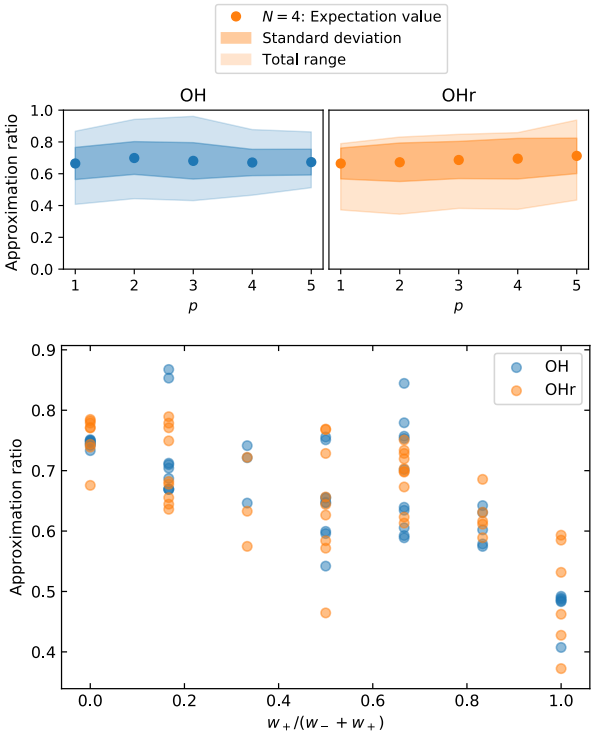


Figure C.1: Performance for QAOA-CC the OH and OHr formulation. The results are for the $N = 4$ complete data set and were obtained by using COBYLA as an optimiser.

Figure C.1 (top) shows the average approximation ratio, standard deviation and total range for both the OH and OHr formulation. We see that on average both formalisms have similar performance, but when it comes to the worst and best case instances OH slightly

outperforms OHr. Figure C.1 (bottom) shows the performance on individual instances when $p = 1$. This shows that only the OH formulation is able to obtain approximation ratios larger than 0.8.

APPENDIX D

An introduction to Rydberg quantum computers

Many different types of physical systems are pursued as candidate hardware for quantum computers, of which some of the most common ones are superconducting [128, 129], trapped ion [130], quantum dots [131] and neutral atoms [132, 133]. For neutral atoms, *Rydberg coupling* can be utilised to perform quantum computations. A *Rydberg state* is a state of an atom (or molecule) in which one of the electrons has been excited to a large *principal quantum number* orbital. Rydberg atoms are relatively long-lived, and the large number of available energy levels allow for coupling to electromagnetic fields spanning over a large order of frequencies. In addition, they offer strong and controllable atomic interactions that can be tuned through the states principal quantum number and orbital angular momentum. These properties make Rydberg atoms attractive candidates for quantum computing.

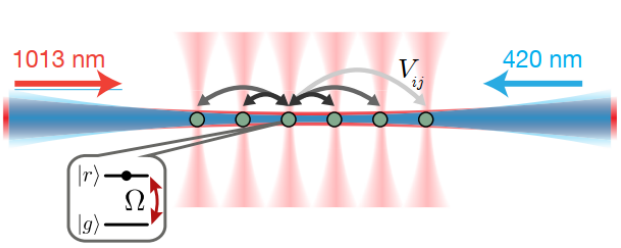


Figure D.1: Rydberg simulator array setup for ^{87}Rb-atoms, trapped using optical tweezers (vertical red beams). Interactions V_{ij} between the atoms (arrows) are enabled by exciting them (horizontal blue and red beams) to a Rydberg state, with strength Ω and detuning Δ (inset). Picture taken from Ref. [134].

Rydberg coupling

We consider individual atoms that are held in an array of arbitrary geometry using so-called *optical tweezers*. These tweezer traps are formed using micron-scale focused beams to create a tight trapping volume that enhances atom-light interactions. Using techniques as dynamic trap reconfiguration [135], spatial light modulating [136] or real-time sorting [137] it is possible to create large defect-free qubit arrays. The quantum information is encoded in hyper-fine ground states with a microwave-scale energy separation. The individual neutral atoms experience extremely weak interactions in the ground state, providing excellent ground

state coherence times. However, this makes two-qubit gate entanglement challenging unless traps are merged to exploit collisional interactions. Coupling of the qubits to Rydberg atoms overcomes this limitation. For atoms in ground state $|g\rangle$ coupled to a Rydberg level $|r\rangle$, the Rydberg Hamiltonian is given by

$$\mathcal{H}_{\text{Ryd}} = \frac{\hbar\Omega}{2}\sum_i \sigma_x^i - \hbar\Delta\sum_i n_i + \sum_{i\neq j} V_{i,j} n_i n_j, \tag{D.1}$$

where $\sigma_x^i = |g_i\rangle\langle r_i| + |r_i\rangle\langle g_i|$ describes the coupling between ground and Rydberg levels, Ω is the driving Rabi frequency, Δ the laser detuning, n_i the number operator on site i and V_{ij} is the Rydberg energy shift [138]. The Rydberg energy shift scales as

$$V_{ij} \propto \frac{1}{r_{ij}^6}, \tag{D.2}$$

where r_{ij} is the distance between sites i and j. If we look at the different terms in (D.1), we see that the first term describes the laser driving between states. The second term describes the extent to which the laser is off-resonance with the transition. Finally, the last term represents the *Rydberg blockade effect*: when an electron is excited to a Rydberg state it creates a potential that shifts the energy levels of nearby atoms. Consequently, the laser cannot drive these atoms to the Rydberg state due to the mismatch between the laser and transition frequencies. This property can be used to create a CNOT operation in a very natural way, as we will see in the next section.

Rydberg quantum gates

In general, the time evolution of a Hamiltonian $\mathcal{H}$ is given by (setting $\hbar$ to unity)

$$U(t) = e^{-i\mathcal{H}t}. \tag{D.3}$$

However, if n-qubit Hamiltonian $\mathcal{H} = \mathcal{H}_1 + \cdots + \mathcal{H}_n$ contains terms that do not commute we have to use Trotterisation to be able to decompose the total Hamiltonian:

$$e^{i\mathcal{H}t} = (e^{i\mathcal{H}_1 t/r}\ldots e^{i\mathcal{H}_n t/r} + E)^r, \tag{D.4}$$

which introduces an error E with $\|E\| = O(t^2/r^2)$.

Defining our Hamiltonian as $\mathcal{H}_{\text{Ryd}}$, we have all ingredients needed to perform quantum gate operations in a Rydberg quantum simulator. Let us give an example on how to construct such a gate: the controlled Z-gate. First, we define two important single-qubit operations in Rydberg quantum computing, the so-called π- and 2π-pulses, defined as

$$U_\pi = \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix}, \quad U_{2\pi} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix},$$

which both can be constructed from the time evolution of the Rydberg Hamiltonian (D.1) using only the first term with $\Omega t = \pi$ or $\Omega t = 2\pi$, respectively. We define control and target basis states $|0_c\rangle$,$|1_c\rangle$,$|0_t\rangle$ and $|1_t\rangle$ on which the unitary acts, as well as Rydberg states $|r_c\rangle$ and $|r_t\rangle$. Our controlled Z gate is applied through a three pulse sequence: 1) a π pulse on the control atom, 2) a 2π pulse on the target atom and 3) a π pulse on the control atom again.

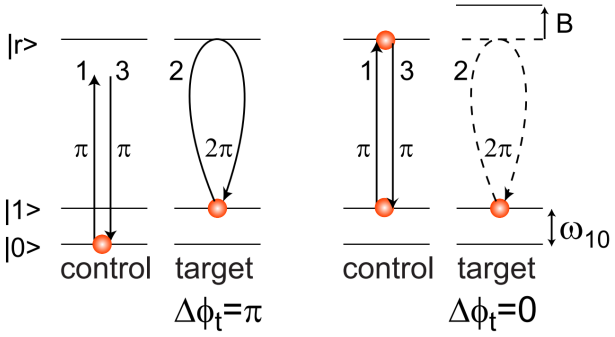


Figure D.2: Left: the control atom is in $|0_c\rangle$, no Rydberg blocking takes place and the 2π pulse gives the target atom a phase shift. Right: the control qubit is in $|1_c\rangle$, the Rydberg blocking prevents the target atom to pick up a phase shift. Picture taken from Ref. [139].

We consider the case when the target atom is in $|1_t\rangle$. If the control atom is in $|0_c\rangle$, it is not Rydberg excited and there is no blockade. This results in a phase shift of π for the target atom. However, when the control atom is in $|1_c\rangle$, both atoms are coupled to the Rydberg level. When the two-atom blockade shift B due to the Rydberg interaction is large compared to Ω, the excitation of the target atom is blocked and it picks up no phase shift. This way we have used the Rydberg blockade effect to create a C_Z operation. Using different laser schemes, whilst exploiting the Rydberg blockade properties, other gates can be constructed as well (see also Appendix E).

Qudit quantum computing

So far we just considered qubit operations, but the formalism can in principle also be extended to qudit systems. In September 2019, Stellmer et al. [140] created the first strontium Bose-Einstein condensate. Since the fermionic isotope of Sr^{87} has a nuclear spin of $I = 9/2$, it allows for 10 spin states to be used to store quantum information (Figure D.3). Therefore, it would in theory be able to physically encode a 10-level qudit. This makes these systems potentially suitable for our ML-formulation. An example of an implementation scheme for this formulation, considering only qubits but with a formalism that can be extended to qudits, is given in Appendix E.

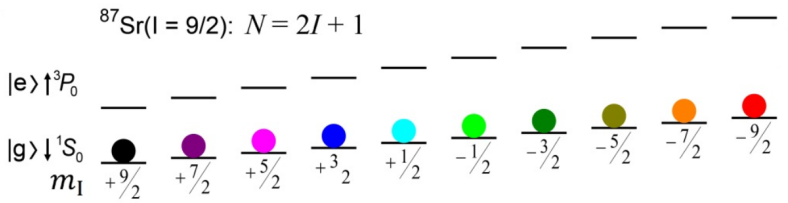


Figure D.3: Energy levels for the two lowest electronic states of ^{87}Sr in a magnetic field, each with ten nuclear spin states, depicted by colours. Adapted picture taken from Ref. [141].

APPENDIX E

Rydberg implementation scheme for two clusters

This appendix is created from notes of work-in-process by prof. dr. C.J.M. Schoutens and J. Minar. It proposes an implementation scheme for the ML-formulation on a Rydberg quantum computer. Currently, we are working on creating one full-stack quantum computing story that includes the algorithmic results of this thesis, the implementation scheme for such a Rydberg quantum computer as well as some simulations using a realistic noise model.

The Rydberg Hamiltonian

We consider d states from a ground state manifold $|0\rangle, \ldots, |d-1\rangle$, coupled to an excited Rydberg state $|r\rangle$ by bosonic fields a_j described by a Hamiltonian

$$H_1 = \sum_{j=1}^{d-1} \omega_j^L a_j^\dagger a_j + g_j(a_j \sigma_j^+ + a_j^\dagger \sigma_j^-) + \omega_j |j\rangle\langle j| + \omega_r |r\rangle\langle r| . \tag{E.1}$$

Transforming this to the frame where each bosonic field rotates with its respective frequency, $H \to U_L H U_L^\dagger - i\dot{U}_L U_L^\dagger$, $U_L = \exp\left[-it\sum_j \omega_j^L a_j^\dagger a_j\right]$, we obtain

$$H_2 = \sum_{j=1}^{d-1} g_j \left(e^{i\omega_j^L t} a_j \sigma_j^+ + \text{H.c.}\right) + \omega_j |j\rangle\langle j| + \omega_r |r\rangle\langle r| . \tag{E.2}$$

This is followed by transforming to a new frame where the atomic states rotate with their respective frequencies, $H \to U_a H U_a^\dagger - i\dot{U}_a U_a^\dagger, U_a = \exp\left[-it\sum_j \omega_j |j\rangle\langle j| + \omega_r |r\rangle\langle r|\right]$

$$H_3 = \sum_{j=1}^{d-1} g_j \left(e^{i\Delta_j^L t} a_j \sigma_j^+ + \text{H.c.}\right), \tag{E.3}$$

where $\Delta_j = \omega_r - \omega_j^L - \omega_j$. Alternatively, one can remove the explicit time dependence in the spin flip operators by rotating to a frame defined by $U = \exp\left[it(\omega_j^L |j\rangle\langle j|)\right]$. In this case, H_2 becomes

$$H_2 = \sum_{j=1}^{d-1} w_j' |j\rangle\langle j| + g_j(a_j \sigma_j^+ + \text{H.c.}) + \omega_r |r\rangle\langle r| , \tag{E.4}$$

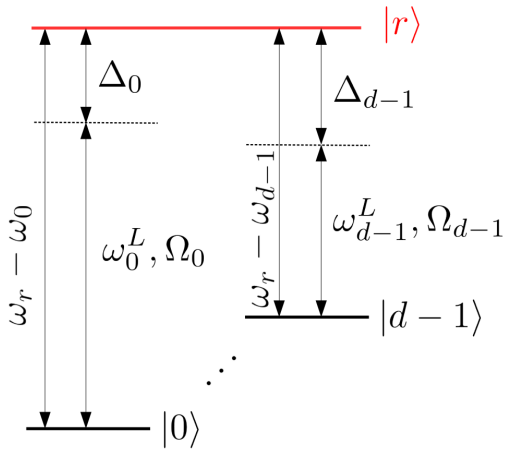


Figure E.1: Level scheme for d-level quantum system with a Rydberg state $|r\rangle$.

where $\omega_j' = \omega_j^L + \omega_j = \omega_r + \Delta_j$. Using $\mathbb{I} = \sum_{j=0}^{d-1} |j\rangle\langle j| + |r\rangle\langle r|$ (which is subsequently dropped), we can rewrite H_2' as

$$H = \sum_{j=0}^{d-1} -\Delta_j |j\rangle\langle j| + g_j(a_j\sigma_j^\dagger + \text{H.c.}) + V|r\rangle\langle r|, \tag{E.5}$$

where we introduced some interaction energy V of the excited state $|r\rangle$.

Qubits representing two possible clusters

In the following section we will design an implementation scheme for the case when we have two clusters ($\kappa = 2$), such that the scheme can be implemented on a qubit system. It is also possible to extend this scheme to a general d-level system using similar operations.

We are interested in classical driving of the qubits. Therefore, we take the bosonic fields to be in a coherent state $|a_j\rangle, \mathbf{a}_j \in \mathbb{C}, |a_j| \gg 1$, such that $g_j\langle a_j\rangle = g_j\alpha_j = \Omega_j, g_j\langle a_j^\dagger\rangle = g_j\alpha_j^* = \Omega_j^*$. We can now write the Hamiltonian (E.5) for a single qubit in matrix notation as

$$H = \begin{pmatrix} -\Delta_0 & 0 & \Omega_0 \\ 0 & -\Delta_1 & \Omega_1 \\ \Delta_0^* & \Delta_1^* & V \end{pmatrix}. \tag{E.6}$$

The associated unitary operator can in principle be obtained by diagonalisation of (E.6), which leads to a cubic equation for the eigenvalues. If we consider two-photon resonance for both qubit levels, i.e. $\Delta_0 = \Delta_1 = 0.$, the situation simplifies. In the limit of large detuning, $V \to \infty$, we obtain

$$U = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & e^{-iVt} \end{pmatrix}. \tag{E.7}$$

For a general V, the $|r\rangle\langle r|$ element of $U = e^{-itH}$ reads

$$U_{rr} = e^{-\frac{1}{2}itV}\left[\cos\left(\frac{1}{2}t\sqrt{V^2+4\Omega^2}\right) - \frac{iV\sin()\frac{1}{2}t\sqrt{V^2+4\Omega^2}}{\sqrt{V^2+4\Omega^2}}\right], \tag{E.8}$$

where $\Omega = \sqrt{|\Omega_0|^2+|\Omega_1|^2}$. Since we require that the Rydberg population is zero at the end of the evolution, when there is initially no excitation in the Rydberg state, we have the following condition on the duration

$$t = \frac{2n\pi}{\sqrt{V^2+4\Omega^2}}, n \in \mathbb{N}. \tag{E.9}$$

Specifically, at resonance $V = 0$,

$$U(\theta,\varphi)) = \begin{pmatrix} \cos\theta & \sin\theta e^{i\varphi} & 0 \\ \sin\theta e^{-i\varphi} & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ for } n \text{ odd}, \tag{E.10}$$

and

$$U = \mathbb{I} \text{ for } n \text{ even}, \tag{E.11}$$

where

$$\cos\theta = \frac{|\Omega_0|^2-|\Omega_1|^2}{\Omega^2}, \sin\theta = \frac{2\Omega_0\Omega_1^*}{\Omega^2}, \tag{E.12}$$

and $\varphi = \phi_0 - \phi_1$ with $\Omega_{0,1} = |\Omega_{0,1}|e^{i\phi_{0,1}}$. Coupling only one of the qubit levels $|j\rangle$ resonantly to the Rydberg state would be described by

$$H = \begin{pmatrix} 0 & \Omega \\ \Omega^* & 0 \end{pmatrix} \to U(\theta,\varphi) = e^{-iHt}\begin{pmatrix} \cos\theta & -i\sin\theta e^{i\varphi} \\ -i\sin\theta e^{-i\varphi} & \cos\theta \end{pmatrix}. \tag{E.13}$$

We will use all the established operations to create the operations needed to perform our QAOA algorithm in the ML formulation with $\kappa = 2$:

The initial state The uniform superposition can be created by applying Hadamard operations on every qubit. The Hadamard operation, within a global phase factor, can be achieved in the following way

$$H = U(\pi/2,0)U(\pi/4,-\pi/4), \tag{E.14}$$

since

$$\begin{pmatrix} \cos\frac{\pi}{2} & -i\sin\frac{\pi}{2} \\ -i\sin\frac{\pi}{2} & \cos\frac{\pi}{2} \end{pmatrix}\begin{pmatrix} \cos\frac{\pi}{4} & -i\sin\frac{\pi}{4}e^{-i\frac{\pi}{2}} \\ -i\sin\frac{\pi}{4}e^{-i\frac{\pi}{2}} & \cos\frac{\pi}{4} \end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix} -i & -i \\ -i & i \end{pmatrix}, \tag{E.15}$$

which after adding a global phase of $\pi/2$ results in

$$\frac{1}{\sqrt{2}}\begin{pmatrix} e^{i\frac{\pi}{2}} & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{pmatrix}\begin{pmatrix} -i & -i \\ -i & i \end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{E.16}$$

which is the Hadamard gate.

The cost unitary The two-qubit cost operation is slightly more involved, but eventually we want to design a two-body interaction $U_V(\varphi)$ on qubits representing nodes u and v that achieves (up to a global phase):

$$U_V(\varphi) = \begin{cases} |ij\rangle \to e^{i\varphi} |ij\rangle \text{ if } i = j \\ |ij\rangle \to e^{-i\varphi} |ij\rangle \text{ if } i \neq j, \end{cases} \tag{E.17}$$

where $\varphi = w_{u,v}\beta, w_{u,v} \in \{-1, +1\}, \beta \in [0, 2\pi)$. First, we define the 1-qubit phase gate $P_{\varphi=\varphi_1-\varphi_2} = U(\frac{\pi}{2}, \varphi_1)U(\frac{\pi}{2}, \varphi_2)$, written explicitly for two levels (basis $|j\rangle, |r\rangle$) as

$$P_\varphi = \begin{pmatrix} e^{i\varphi} & 0 \\ 0 & e^{-i\varphi} \end{pmatrix}. \tag{E.18}$$

We define the unitary of a $\pi-$pulse as $U_\pi \equiv U(\pi/2, 0)$. Using just $\pi-$pulses and 1-qubit phase gates, we can achieve our desired interaction through the following sequence of operations (written in the $\{|uu\rangle, |ud\rangle, |du\rangle, |dd\rangle\}$-basis)

$$U^{(1)}_{\pi,ur} P^{(2)}_{\varphi,dr} U^{(1)}_{\pi,ur} U^{(2)}_{\pi,dr} P^{(1)}_{\varphi,ur} U^{(2)}_{\pi,dr} = \text{diag}(e^{i\varphi}, 1, 1, e^{i\varphi}), \tag{E.19}$$

where $G^{(n)}_{\alpha,ij}$ is the gate acting between levels $|i\rangle, |j\rangle$ of the $n-$th qubit. Note that this implements our desired interaction up to a global phase of $-\varphi/2$, since

$$\text{diag}(e^{i\varphi'}, 1, 1, e^{i\varphi'})e^{-i\varphi'/2} = \text{diag}(e^{i\varphi'/2}, e^{-i\varphi'/2}, e^{-i\varphi'/2}, e^{i\varphi'/2}) = U_V(\varphi'), \tag{E.20}$$

where $\varphi' = 2\varphi/2 = 2w_{u,v}\beta$.

The mixing unitary For the mixing operation, we simply require a rotation around the $x-$axis, defined as

$$R_x(\gamma/2) = U(\gamma/2, 0) = \begin{pmatrix} \cos\gamma/2 & -i\sin\gamma/2 \\ -i\sin\gamma/2 & \cos\gamma/2 \end{pmatrix}. \tag{E.21}$$

This defines all necessary operations to implement our QAOA algorithm for $\kappa = 2$ on a Rydberg quantum computer with Hamiltonian (E.5).

APPENDIX F

Quantum mechanics and Markov theory

The following appendix on the link between quantum mechanics and stochastic theory was requested by one of my supervisors, prof. dr. R.J. Boucherie. It should be viewed as a separate appendix to the rest of the work, and mainly covers the work by Godart who has over the years attempted to develop a stochastic theory of quantum mechanics [142, 143, 144].

Stochastic mechanics is an attempt to formulate a stochastic theory that describes quantum mechanics. In 1952, Fenyes suggested a theory in which quantum mechanics is fundamentally described by particles that move along trajectories that are random sample functions of a Markov process [145]. This theory never really took off, and in 1966 Nelson showed a derivation of quantum mechanics from classical mechanics and Brownian motion, essentially rediscovering Fenyes' idea [146]. In this work, Nelson uses that any wave function $\Psi(\vec{x}, t)$ that is a solution to the Schrödinger equation is of the form

$$\Psi(\vec{x}, t) = \exp[R(\vec{x}, t) + iS(\vec{x}, t]. \tag{F.1}$$

Subsequently, he defines two vectors $\vec{u}(\vec{x}, t)$ and $\vec{v}(\vec{x}, t)$ such that

$$\begin{aligned} m\vec{u}(\vec{x}, t) &= \frac{h}{2\pi}\nabla R(x, t) \\ m\vec{v}(\vec{x}, t) &= \frac{h}{2\pi}\nabla S(x, t) - e\vec{A}(\vec{x}, t) \end{aligned} \tag{F.2}$$

and remarks that the vectors $\vec{v}_{\pm}(\vec{x}, t) = \vec{v}(\vec{x}, t) \pm \vec{u}(\vec{x}, t)$ obey the classical equations of a Markov process. Finally, he claims that these vectors coincide with the forward and backward drift vectors that characterise such stochastic processes.

However, several authors raised objectives to this theory. The main argument is that these vectors are not well-defined at points where the wave function $\Psi(\vec{x}, t)$ is equal to zero. In a more recent work by Godart, he tries to counter these arguments by working in the reverse direction compared to what Nelson did: instead of staring out from the wave function he shows that the stochastic theory of quantum mechanics allows him to recover the solutions to the Schrödinger equation in a great number of particular cases. For this, he uses the following three principles to start with:

1. Trajectories of particles are sample functions of a Markov stochastic process.
2. The diffusion tensor in non-relativistic stochastic theory is given by

$$w^{ij} = \kappa g^{ij}, \tag{F.3}$$

with $\kappa = h/4\pi m$, h is Planck's constant and g^{ij} the metric tensor, which is in a Cartesian coordinate system equal to

$$g^{ij} = \begin{cases} 1 \text{ if } i = j, \\ 0 \text{ if } i \neq j. \end{cases} \tag{F.4}$$

3. The stochastic mechanics will depend on a stochastic variational principle, which states that the evolution of a system that starts at point $\vec{x_0}$ at time t_0 that arrives at $\vec{x_1}$ at time t_1 is described by a Markov process that makes the variation of the functional zero.

By assuming that the electric field $\vec{E}(\vec{x})$ and magnetic field $\vec{B}(\vec{x})$ that act on the particle do not depend on time, he shows that from the solutions of the Kolmogorov and Fokker-Planck equations he is able to retrieve the time-independent Schrödinger equation:

$$\frac{h^2}{8\pi^2 m}\Delta\Psi = (E - V)\Psi. \tag{F.5}$$

However, this is only possible because of another condition set along the way: the magnetic field must be equal to zero. In the conclusion of the work, it is noted that some other equations have to be adopted in order to use stochastic theory to explain quantum mechanics with non-zero magnetic fields.

In the end, stochastic mechanics is able to formally recover the elementary solutions of the Schrödinger in a great number of particular cases (including ones that include relativity), but has so far not been able to show equivalence to the orthodox Copenhagen interpretation of quantum mechanics. However, it also remains to be shown whether other, alternative stochastic approaches might not be able to show equivalence.

Bibliography

[1] P. W. Shor. 'Algorithms for Quantum Computation: Discrete Logarithms and Factoring'. In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134.

[2] A. W. Harrow, A. Hassidim and S. Lloyd. 'Quantum Algorithm for Linear Systems of Equations'. In: *Physical Review Letters* 103.15 (Oct. 2009).

[3] L. K. Grover. 'A Fast Quantum Mechanical Algorithm for Database Search'. In: *arXiv e-prints* (May 1996). arXiv: quant-ph/9605043.

[4] I. Kassal, S. P. Jordan, P. J. Love, M. Mohseni and A. Aspuru-Guzik. 'Polynomial-time Quantum Algorithm for the Simulation of Chemical Dynamics'. In: *Proceedings of the National Academy of Science* 105.48 (Dec. 2008), pp. 18681–18686. arXiv: 0801.2986.

[5] J. Kelly, Z. Chen, B. Chiaro, B. Foxen, J. Martinis and Google Quantum Hardware Team. 'Operating and Characterizing of a 72 Superconducting Qubit Processor "Bristlecone": Part 1'. In: *APS March Meeting Abstracts*. Vol. 2019. APS Meeting Abstracts. Jan. 2019, A42.002.

[6] E. Farhi, J. Goldstone and S. Gutmann. 'A Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Nov. 2014). arXiv: 1411.4028.

[7] M. Fingerhuth, T. Babej and C. Ing. 'A Quantum Alternating Operator Ansatz with Hard and Soft Constraints for Lattice Protein Folding'. In: *arXiv e-prints* (Oct. 2018). arXiv: 1810.13411.

[8] L. Tse, P. Mountney, P. Klein and S. Severini. 'Graph Cut Segmentation Methods Revisited with a Quantum Algorithm'. In: *arXiv e-prints* (Dec. 2018). arXiv: 1812.03050.

[9] M. M. Wauters, G. Bigan Mbeng and G. E. Santoro. 'Polynomial Scaling of QAOA for Ground-state Preparation of the Fully-connected p-spin Ferromagnet'. In: *arXiv e-prints* (Mar. 2020). arXiv: 2003.07419.

[10] E. Farhi, J. Goldstone, S. Gutmann and L. Zhou. 'The Quantum Approximate Optimization Algorithm and the Sherrington-Kirkpatrick Model at Infinite Size'. In: *arXiv e-prints* (Oct. 2019). arXiv: 1910.08187.

[11] J. Cook, S. Eidenbenz and A. Bärtschi. 'The Quantum Alternating Operator Ansatz on Maximum k-Vertex Cover'. In: *arXiv e-prints* (Oct. 2019). arXiv: 1910.13483.

[12] Y.-H. Oh, H. Mohammadbagherpoor, P. Dreher, A. Singh, X. Yu and A. J. Rindos. 'Solving Multi-Coloring Combinatorial Optimization Problems Using Hybrid Quantum Algorithms'. In: *arXiv e-prints* (Nov. 2019). arXiv: 1911.00595.

[13] F. G. Fuchs, H. Oie Kolden, N. H. Aase and G. Sartor. 'Efficient Encoding of the Weighted MAX k-CUT on a Quantum Computer using QAOA'. In: *arXiv e-prints* (Sept. 2020). arXiv: 2009.01095.

[14] E. R. Anschuetz, J. P. Olson, A. Aspuru-Guzik and Y. Cao. 'Variational Quantum Factoring'. In: *arXiv e-prints* (Aug. 2018). arXiv: 1808.08927.

[15] D. An and L. Lin. 'Quantum Linear System Solver based on Time-optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Sept. 2019). arXiv: 1909.05500.

[16] G. Verdon, J. Pye and M. Broughton. 'A Universal Training Algorithm for Quantum Deep Learning'. In: *arXiv e-prints* (June 2018). arXiv: 1806.09729.

[17] T. Matsumine, T. Koike-Akino and Y. Wang. 'Channel Decoding with Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Mar. 2019). arXiv: 1903.02537.

[18] A. Borle, V. E. Elfving and S. J. Lomonaco. 'Quantum Approximate Optimization for Hard Problems in Linear Algebra'. In: *arXiv e-prints* (June 2020). arXiv: 2006.15438.

[19] M. Szegedy. 'What Do QAOA Energies Reveal about Graphs?' In: *arXiv e-prints* (Dec. 2019). arXiv: 1912.12277.

[20] M. Hodson, B. Ruck, H. Ong, D. Garvin and S. Dulman. 'Portfolio Rebalancing Experiments Using the Quantum Alternating Operator Ansatz'. In: *arXiv e-prints* (Nov. 2019). arXiv: 1911.05296.

[21] N. Bansal, A. Blum and S. Chawla. 'Correlation Clustering.' In: *Mach. Learn.* 56.1-3 (July 2004), pp. 89–113.

[22] C. Swamy. 'Correlation Clustering: Maximizing Agreements via Semidefinite Programming'. In: *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*. Vol. 15. Jan. 2004, pp. 526–527.

[23] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. 10th. USA: Cambridge University Press, 2011.

[24] M. Kues et al. 'On-chip Generation of High-dimensional Entangled Quantum States and Their Coherent Control'. In: *Nature* 546 (June 2017), pp. 622–626.

[25] S. Stellmer, F. Schreck and T. C. Killian. 'Degenerate Quantum Gases of Strontium'. In: *Annual Review of Cold Atoms and Molecules - Volume 2*. Ed. by K. W. Madison and et al. Vol. 2. 2014, pp. 1–80.

[26] E. Bernstein and U. Vazirani. 'Quantum complexity theory'. In: *Proc. 25th Annual ACM Symposium on Theory of Computing, ACM*. 1993, pp. 11–20.

[27] S. Aaronson. *Introduction to Quantum Information Science Lecture Notes*. UT Austin, 2018.

[28] R. de Wolf. 'Quantum Computing: Lecture Notes'. In: *arXiv e-prints* (July 2019). arXiv: 1907.09415.

[29] F. Harary. 'On the Notion of Balance of a Signed Graph.' In: *Michigan Math. J.* 2.2 (1953), pp. 143–146.

[30] A. Ben-Dor and Z. Yakhini. 'Clustering Gene Expression Patterns'. In: *Proceedings of the Third Annual International Conference on Computational Molecular Biology*. RECOMB '99. Lyon, France: Association for Computing Machinery, 1999, pp. 33–42.

[31] S. Böcker and J. Baumbach. 'Cluster Editing'. In: *The Nature of Computation. Logic, Algorithms, Applications*. Ed. by P. Bonizzoni, V. Brattka and B. Löwe. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 33–44.

[32] R. Shamir, R. Sharan and D. Tsur. 'Cluster Graph Modification Problems'. In: *Discrete Applied Mathematics* 144.1 (2004). Discrete Mathematics and Data Mining, pp. 173–182.

[33] S. Rahmann, T. Wittkop, J. Baumbach, M. Martin, A. Truss and S. Böcker. 'Exact and Heuristic Algorithms for Weighted Cluster Editing'. In: *Computational systems bioinformatics / Life Sciences Society. Computational Systems Bioinformatics Conference* 6 (Feb. 2007), pp. 391–401.

[34] S. Schaeffer. 'Graph Clustering'. In: *Computer Science Review* 1 (Aug. 2007), pp. 27–64.

[35] A. Wirth. 'Correlation Clustering'. In: *Encyclopedia of Machine Learning*. Ed. by C. Sammut and G. I. Webb. Boston, MA: Springer US, 2010, pp. 227–231.

[36] T. Leighton and S. Rao. 'Multicommodity Max-Flow Min-Cut Theorems and Their Use in Designing Approximation Algorithms'. In: *J. ACM* 46.6 (Nov. 1999), pp. 787–832.

[37] J. Tan. 'A Note on the Inapproximability of Correlation Clustering'. In: *Information Processing Letters* 108.5 (2008), pp. 331–335.

[38] M. Charikar, V. Guruswami and A. Wirth. 'Clustering with Qualitative Information'. In: *J. Comput. Syst. Sci.* 71 (Oct. 2005), pp. 360–383.

[39] M. Karpinski and W. Schudy. 'Linear Time Approximation Schemes for the Gale-Berlekamp Game and Related Minimization Problems'. In: STOC '09. Bethesda, MD, USA: Association for Computing Machinery, 2009, pp. 313–322.

[40] E. D. Demaine and N. Immorlica. 'Correlation Clustering with Partial Information'. In: *Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems and 7th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM-APPROX 2003)*. Princeton, New Jersey, Aug. 2003, pp. 1–13.

[41] D. Emanuel and A. Fiat. 'Correlation clustering – minimizing disagreements on arbitrary weighted graphs'. In: *Proceedings of the 11th Annual European Symposium on Algorithms*. Springer, 2003, pp. 208–220.

[42] M. Bertolacci and A. Wirth. 'Are Approximation Algorithms for Consensus Clustering Worthwhile?' In: *Proceedings of the 7th SIAM International Conference on Data Mining*. Apr. 2007.

[43] S. Chawla, K. Makarychev, T. Schramm and G. Yaroslavtsev. 'Near Optimal LP Rounding Algorithm for Correlation Clustering on Complete and Complete k-partite Graphs'. In: *arXiv e-prints* (Dec. 2014). arXiv: 1412.0681.

[44] C. Bauckhage, E. Brito, K. Cvejoski, C. Ojeda, R. Sifa and S. Wrobel. 'Adiabatic Quantum Computing for Binary Clustering'. In: *arXiv e-prints* (June 2017). arXiv: 1706.05528.

[45] H. Ushijima-Mwesigwa, C. F. A. Negre and S. M. Mniszewski. 'Graph Partitioning Using Quantum Annealing on the D-Wave System'. In: *Proceedings of the Second International Workshop on Post Moores Era Supercomputing*. PMES'17. Denver, CO, USA: Association for Computing Machinery, 2017, pp. 22–29.

[46] J. S. Otterbach et al. 'Unsupervised Machine Learning on a Hybrid Quantum Computer'. In: *arXiv e-prints* (Dec. 2017). arXiv: 1712.05771.

[47] R. Shaydulin, I. Safro and J. Larson. 'Multistart Methods for Quantum Approximate Optimization'. In: *arXiv e-prints* (May 2019). arXiv: 1905.08768.

[48] E. Aimeur, G. Brassard and S. Gambs. 'Quantum Speed-up for Unsupervised Learning'. In: *Machine Learning* 90 (Feb. 2013), pp. 261–287.

[49] S. Pramanik and M. Girish Chandra. 'Quantum-Assisted Graph Clustering and Quadratic Unconstrained D-ary Optimisation'. In: *arXiv e-prints* (Apr. 2020). arXiv: 2004.02608.

[50] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler and B. Schiele. 'DeepCut: Joint Subset Partition and Labeling for Multi Person Pose Estimation'. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[51] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka and B. Schiele. 'DeeperCut: A Deeper, Stronger, and Faster Multi-Person Pose Estimation Model'. In: *European Conference on Computer Vision (ECCV)*. May 2016.

[52] E. Farhi, J. Goldstone and S. Gutmann. 'A Quantum Approximate Optimization Algorithm Applied to a Bounded Occurrence Constraint Problem'. In: *arXiv e-prints* (Dec. 2014). arXiv: 1412.6062.

[53] B. Barak, A. Moitra, R. O'Donnell, P. Raghavendra, O. Regev, D. Steurer, L. Trevisan, A. Vijayaraghavan, D. Witmer and J. Wright. 'Beating the Random Assignment on Constraint Satisfaction Problems of Bounded Degree'. In: *arXiv e-prints* (May 2015). arXiv: 1505.03424.

[54] 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'. In: *arXiv e-prints* (Sept. 2017). arXiv: 1709.03489.

[55] E. Farhi, J. Goldstone, S. Gutmann and M. Sipser. 'Quantum Computation by Adiabatic Evolution'. In: *arXiv e-prints* (Jan. 2000). arXiv: quant-ph/0001106.

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

[57] E. Haber. *The Time Evolution Operator as a Time-ordered Exponential*. scipp.ucsc.edu/~haber/ph215/TimeOrderedExp.pdf. 2018.

[58] G. Verdon, M. Broughton and J. Biamonte. 'A Quantum Algorithm to Train Neural Networks Using Low-depth Circuits'. In: *arXiv e-prints* (Dec. 2017). arXiv: 1712.05304.

[59] M. Streif and M. Leib. 'Comparison of QAOA with Quantum and Simulated Annealing'. In: *arXiv e-prints* (Jan. 2019). arXiv: 1901.01903.

[60] L. T. Brady, C. L. Baldwin, A. Bapat, Y. Kharkov and A. V. Gorshkov. 'Optimal Protocols in Quantum Annealing and QAOA Problems'. In: *arXiv e-prints* (Mar. 2020). arXiv: 2003.08952.

[61] 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'. In: *arXiv e-prints* (Dec. 2018). arXiv: 1812.01041.

[62] E. Farhi, J. Goldstone, S. Gutmann and D. Nagaj. 'How to Make the Quantum Adiabatic Algorithm Fail'. In: *arXiv e-prints* (Dec. 2005). arXiv: quant-ph/0512159.

[63] Z. Wang, S. Hadfield, Z. Jiang and E. G. Rieffel. 'Quantum Approximate Optimization Algorithm for MaxCut: A Fermionic View'. In: *Phys. Rev. A* 97 (2 Feb. 2018), p. 022304.

[64] M. Yuezhen Niu, S. Lu and I. L. Chuang. 'Optimizing QAOA: Success Probability and Runtime Dependence on Circuit Depth'. In: *arXiv e-prints* (May 2019). arXiv: 1905.12134.

[65] R. Shaydulin and Y. Alexeev. 'Evaluating Quantum Approximate Optimization Algorithm: A Case Study'. In: *arXiv e-prints* (Oct. 2019). arXiv: 1910.04881.

[66] M. Willsch, D. Willsch, F. Jin, H. De Raedt and K. Michielsen. 'Benchmarking the Quantum Approximate Optimization Algorithm'. In: *Quantum Information Processing* 19.7, 197 (June 2020), p. 197. arXiv: 1907.02359.

[67] M. Streif and M. Leib. 'Forbidden Subspaces for Level-1 Quantum Approximate Optimization Algorithm and Instantaneous Quantum Polynomial Circuits'. In: *Phys. Rev. A* 102 (4 Oct. 2020), p. 042416.

[68] C. Moussa, H. Calandra and V. Dunjko. 'To Quantum or not to Quantum: Towards Algorithm Selection in Near-term Quantum Optimization'. In: *Quantum Science and Technology* 5.4, 044009 (Oct. 2020), p. 044009. arXiv: 2001.08271.

[69] Z.-C. Yang, A. Rahmani, A. Shabani, H. Neven and C. Chamon. 'Optimizing Variational Quantum Algorithms Using Pontryagin's Minimum Principle'. In: *Phys. Rev. X* 7 (2 May 2017), p. 021027.

[70] S. Lloyd. 'Quantum Approximate Optimization is Computationally Universal'. In: *arXiv e-prints*, arXiv:1812.11075 (Dec. 2018), arXiv:1812.11075. arXiv: 1812.11075.

[71] M. E. S. Morales, J. D. Biamonte and Z. Zimborás. 'On the Universality of the Quantum Approximate Optimization Algorithm'. In: *Quantum Information Processing* 19.9, 291 (Aug. 2020), p. 291. arXiv: 1909.03123.

[72] S. Hadfield, Z. Wang, E. G. Rieffel, B. O'Gorman, D. Venturelli and R. Biswas. 'Quantum Approximate Optimization with Hard and Soft Constraints'. In: *Proceedings of the Second International Workshop on Post Moores Era Supercomputing*. PMES'17. Denver, CO, USA: Association for Computing Machinery, 2017, pp. 15–21.

[73] Y. Ruan, S. Marsh, X. Xue, X. Li, Z. Liu and J. Wang. 'Quantum Approximate Algorithm for NP Optimization Problems with Constraints'. In: *arXiv e-prints* (Jan. 2020). arXiv: 2002.00943.

[74] Z. Wang, N. C. Rubin, J. M. Dominy and E. G. Rieffel. 'XY Mixers: Analytical and Numerical Results for the Quantum Alternating Operator Ansatz'. In: *Phys. Rev. A* 101.1 (Jan. 2020). arXiv: 1904.09314.

[75] W. W. Ho and T. H. Hsieh. 'Efficient Variational Simulation of Non-trivial Quantum States'. In: *SciPost Physics* 6.3, 029 (Mar. 2019), p. 029. arXiv: 1803.00026.

[76] S. Marsh and J. B. Wang. 'A Quantum Walk-assisted Approximate Algorithm for Bounded NP Optimisation Problems'. In: *Quantum Information Processing* 18.3, 61 (Mar. 2019), p. 61.

[77] A. Bapat and S. P. Jordan. 'Bang-bang Control as a Design Principle for Classical and Quantum Optimization Algorithms'. In: *Quantum Information & Computation* 19 (2018), pp. 424–446.

[78] L. Zhu, H. Lun Tang, G. S. Barron, N. J. Mayhall, E. Barnes and S. E. Economou. 'An Adaptive Quantum Approximate Optimization Algorithm for Solving Combinatorial Problems on a Quantum Computer'. In: *arXiv e-prints* (May 2020). arXiv: 2005.10258.

[79] A. Bärtschi and S. Eidenbenz. 'Grover Mixers for QAOA: Shifting Complexity from Mixer Design to State Preparation'. In: *arXiv e-prints* (May 2020). arXiv: 2006.00354.

[80] Y. Zhang, R. Zhang and A. C. Potter. 'QED Driven QAOA for Network-flow Optimization'. In: *arXiv e-prints* (June 2020). arXiv: 2006.09418.

[81] D. J. Egger, J. Marecek and S. Woerner. 'Warm-starting Quantum Optimization'. In: *arXiv e-prints* (Sept. 2020).

[82] L. Li, M. Fan, M. Coram, P. Riley and S. Leichenauer. 'Quantum Optimization with a Novel Gibbs Objective Function and Ansatz Architecture Search'. In: *Physical Review Research* 2.2, 023074 (Apr. 2020), p. 023074. arXiv: 1909.07621.

[83] G. Giacomo Guerreschi and M. Smelyanskiy. 'Practical Optimization for Hybrid Quantum-classical Algorithms'. In: *arXiv e-prints* (Jan. 2017). arXiv: 1701.01450.

[84] D. Wecker, M. B. Hastings and M. Troyer. 'Training a Quantum Optimizer'. In: *Phys. Rev. A* 94.2, 022309 (Aug. 2016), p. 022309. arXiv: 1605.05370.

[85] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev and P. Balaprakash. 'Reinforcement-Learning-Based Variational Quantum Circuits Optimization for Combinatorial Problems'. In: *arXiv e-prints* (Nov. 2019). arXiv: 1911.04574.

[86] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev and P. Balaprakash. 'Learning to Optimize Variational Quantum Circuits to Solve Combinatorial Problems'. In: *arXiv e-prints* (Nov. 2019). arXiv: 1911.11071.

[87] A. Garcia-Saez and J. Riu. 'Quantum Observables for Continuous Control of the Quantum Approximate Optimization Algorithm via Reinforcement Learning'. In: *arXiv e-prints* (Nov. 2019). arXiv: 1911.09682.

[88] J. Yao, M. Bukov and L. Lin. 'Policy Gradient Based Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Feb. 2020). arXiv: 2002.01068.

[89] R. Shaydulin, I. Safro and J. Larson. 'Multistart Methods for Quantum Approximate Optimization'. In: *arXiv e-prints* (May 2019). arXiv: 1905.08768.

[90] C. Roch, A. Impertro, T. Phan, T. Gabor, S. Feld and C. Linnhoff-Popien. 'Cross Entropy Hyperparameter Optimization for Constrained Problem Hamiltonians Applied to QAOA'. In: *arXiv e-prints* (Mar. 2020). arXiv: 2003.05292.

[91] *Scipy Documentation*. https://www.scipy.org.

[92] W. Lavrijsen, A. Tudor, J. Müller, C. Iancu and W. de Jong. 'Classical Optimizers for Noisy Intermediate-Scale Quantum Devices'. In: *arXiv e-prints* (Apr. 2020). arXiv: 2004.03004.

[93] F. G. S. L. Brandao, M. Broughton, E. Farhi, S. Gutmann and H. Neven. 'For Fixed Control Parameters the Quantum Approximate Optimization Algorithm's Objective Function Value Concentrates for Typical Instances'. In: *arXiv e-prints* (Dec. 2018). arXiv: 1812.04170.

[94] M. Streif and M. Leib. 'Training the Quantum Approximate Optimization Algorithm Without Access to a Quantum Processing Unit'. In: *Quantum Science and Technology* 5.3, 034008 (July 2020), p. 034008. arXiv: 1908.08862.

[95] G. Verdon, J. M. Arrazola, K. Brádler and N. Killoran. 'A Quantum Approximate Optimization Algorithm for Continuous Problems'. In: *arXiv e-prints* (Feb. 2019). arXiv: 1902.00409.

[96] P. Vikstâl, M. Grönkvist, M. Svensson, M. Andersson, G. Johansson and G. Ferrini. 'Applying the Quantum Approximate Optimization Algorithm to the Tail-Assignment Problem'. In: *Physical Review Applied* 14.3, 034009 (Sept. 2020), p. 034009. arXiv: 1912.10499.

[97] Utkarsh, B. K. Behera and P. K. Panigrahi. 'Solving Vehicle Routing Problem Using Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Feb. 2020). arXiv: 2002.01351.

[98] J. Ostrowski, R. Herrman, T. S. Humble and G. Siopsis. 'Lower Bounds on Circuit Depth of the Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Aug. 2020). arXiv: 2008.01820.

[99] D. Venturelli, M. Do, E. Rieffel and J. Frank. 'Compiling Quantum Circuits to Realistic Hardware Architectures Using Temporal Planners'. In: *Quantum Science and Technology* 3.2 (Apr. 2018), p. 025004. arXiv: 1705.08927.

[100] A. Oddi and R. Rasconi. 'Greedy Randomized Search for Scalable Compilation of Quantum Circuits'. In: *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. Ed. by W.-J. van Hoeve. Cham: Springer International Publishing, 2018, pp. 446–461.

[101] D. M. Abrams, N. Didier, B. R. Johnson, M. P. da Silva and C. A. Ryan. 'Implementation of the XY Interaction Family with Calibration of a Single Pulse'. In: *arXiv e-prints* (Dec. 2019). arXiv: 1912.04424.

[102] H. Pichler, S.-T. Wang, L. Zhou, S. Choi and M. D. Lukin. 'Quantum Optimization for Maximum Independent Set Using Rydberg Atom Arrays'. In: *arXiv e-prints* (Aug. 2018). arXiv: 1808.10816.

[103] M. Alam, A. Ash-Saki and S. Ghosh. 'Analysis of Quantum Approximate Optimization Algorithm under Realistic Noise in Superconducting Qubits'. In: *arXiv e-prints* (July 2019). arXiv: 1907.09631.

[104] C. Xue, Z.-Y. Chen, Y.-C. Wu and G.-P. Guo. 'Effects of Quantum Noise on Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Sept. 2019). arXiv: 1909.02196.

[105] Y. Dong, X. Meng, L. Lin, R. Kosut and K. B. Whaley. 'Robust Control Optimization for Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Nov. 2019). arXiv: 1911.00789.

[106] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio and P. J. Coles. 'Noise-Induced Barren Plateaus in Variational Quantum Algorithms'. In: *arXiv e-prints* (July 2020). arXiv: 2007.14384.

[107] F. Arute et al. 'Quantum Approximate Optimization of Non-Planar Graph Problems on a Planar Superconducting Processor'. In: *arXiv e-prints* (Apr. 2020). arXiv: 2004.04197.

[108] G. Pagano et al. 'Quantum Approximate Optimization of the Long-range Ising Model with a Trapped-ion Quantum Simulator'. In: *Proceedings of the National Academy of Science* 117.41 (Oct. 2020), pp. 25396–25401. arXiv: 1906.02700.

[109] E. Farhi and A. W. Harrow. 'Quantum Supremacy through the Quantum Approximate Optimization Algorithm'. In: *arXiv e-prints* (Feb. 2016). arXiv: 1602.07674.

[110] G. Crooks and N. Rubin. 'Performance of the Quantum Approximate Optimization Algorithm on the Maximum Cut Problem'. In: *APS March Meeting Abstracts*. Vol. 2019. APS Meeting Abstracts. Jan. 2019, K27.002.

[111] M. X. Goemans and D. P. Williamson. 'Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming'. In: *J. ACM* 42.6 (Nov. 1995), pp. 1115–1145.

[112] G. G. Guerreschi and A. Y. Matsuura. 'QAOA for Max-Cut Requires Hundreds of Qubits for Quantum Speed-up'. In: *Scientific Reports* 9, 6903 (May 2019), p. 6903. arXiv: 1812.07589.

[113] S. Shankland. *IBM's new 53-qubit quantum computer is its biggest yet*. https://www.cnet.com/news/ibm-new-53-qubit-quantum-computer-is-its-biggest-yet/. Sept. 2019.

[114] A. Dalzell, A. Harrow, D. Koh and R. La Placa. 'How Many Qubits Are Needed for Quantum Computational Supremacy?' In: *APS March Meeting Abstracts*. Vol. 2019. APS Meeting Abstracts. Jan. 2019, K42.002.

[115] M. B. Hastings. 'Classical and Quantum Bounded Depth Approximation Algorithms'. In: *Quantum Information & Computation* 19 (Aug. 2019), pp. 1116–1140.

[116] V. Akshay, H. Philathong, M. E. S. Morales and J. D. Biamonte. 'Reachability Deficits in Quantum Approximate Optimization'. In: *Physical Review Letters* 124.9 (Mar. 2020). arXiv: 1906.11259.

[117] S. Hadfield. 'On the Representation of Boolean and Real Functions as Hamiltonians for Quantum Computing'. In: *arXiv e-prints* (Apr. 2018). arXiv: 1804.09130.

[118] H. Leipold and F. M. Spedalieri. 'Constructing Driver Hamiltonians for Several Linear Constraints'. In: *arXiv e-prints* (June 2020). arXiv: 2006.12028.

[119] F. Vatan and C. P. Williams. 'Realization of a General Three-Qubit Quantum Gate'. In: *arXiv e-prints* (Jan. 2004). arXiv: quant-ph/0401178.

[120] M. J. D. Powell. 'Direct Search Algorithms for Optimization Calculations'. In: *Acta Numerica* 7 (Jan. 1998), pp. 287–336.

[121] D. J. Wales and J. P. K. Doye. 'Global Optimization by Basin-Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms'. In: *The Journal of Physical Chemistry A* 101.28 (July 1997), pp. 5111–5116.

[122] C. T. Kelley. *Implicit filtering*. Software, environments, and tools 23. Society for Industrial and Applied Mathematics SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104, 2011.

[123] M. Powell. 'The BOBYQA Algorithm for Bound Constrained Optimization without Derivatives'. In: *Technical Report, Department of Applied Mathematics and Theoretical Physics* (Jan. 2009).

[124] W. Huyer and A. Neumaier. 'SNOBFIT – Stable Noisy Optimization by Branch and Fit'. In: *ACM Trans. Math. Softw.* 35.2 (July 2008).

[125] J. Wurtz and P. J. Love. 'Bounds on MAXCUT QAOA Performance for p>1'. In: *arXiv e-prints* (Oct. 2020). arXiv: 2010.11209.

[126] H.-W. Jiao, F.-H. Wang and Y.-Q. Chen. 'An Effective Branch and Bound Algorithm for Minimax Linear Fractional Programming'. In: *Journal of Applied Mathematics* 2014 (2014), pp. 1–8.

[127] R. Sato, M. Yamada and H. Kashima. 'Learning to Sample Hard Instances for Graph Algorithms'. In: *arXiv e-prints* (Feb. 2019). arXiv: 1902.09700.

[128] J. Clarke and F. K. Wilhelm. 'Superconducting Quantum Bits'. In: *Nature* 453.7198 (June 2008), pp. 1031–1042.

[129] W. M. Kaminsky, S. Lloyd and T. P. Orlando. 'Scalable Superconducting Architecture for Adiabatic Quantum Computation'. In: *arXiv e-prints* (Mar. 2004). arXiv: quant-ph/0403090.

[130] J. I. Cirac and P. Zoller. 'Quantum Computations with Cold Trapped Ions'. In: *Physical Review Letters* 74.20 (May 1995), pp. 4091–4094.

[131] A. Imamoğlu, D. D. Awschalom, G. Burkard, D. P. Divincenzo, D. Loss, M. Sherwin and A. Small. 'Quantum Information Processing Using Quantum Dot Spins and Cavity QED'. In: *Physical Review Letters* 83.20 (Nov. 1999), pp. 4204–4207.

[132] M. Khazali and K. Mølmer. 'Fast Multiqubit Gates by Adiabatic Evolution in Interacting Excited-State Manifolds of Rydberg Atoms and Superconducting Circuits'. In: *Physical Review X* 10.2, 021054 (Apr. 2020), p. 021054.

[133] M. Saffman. 'Quantum Computing with Neutral Atoms'. In: *National Science Review* 6.1 (Sept. 2018), pp. 24–25. eprint: https://academic.oup.com/nsr/article-pdf/6/1/24/30336094/nwy088.pdf.

[134] H. Bernien et al. 'Probing Many-body Dynamics on a 51-atom Quantum Simulator'. In: *Nature* 551.7682 (Nov. 2017), pp. 579–584.

[135] M. Endres, H. Bernien, A. Keesling, H. Levine, E. R. Anschuetz, A. Krajenbrink, C. Senko, V. Vuletic, M. Greiner and M. D. Lukin. 'Cold Matter Assembled Atom-by-Atom'. In: *arXiv e-prints* (July 2016). arXiv: 1607.03044.

[136] H. Kim, W. Lee, H.-G. Lee, H. Jo, Y. Song and J. Ahn. 'In Situ Single-atom Array Synthesis Using Dynamic Holographic Optical Tweezers'. In: *Nature Communications* 7, 13317 (Oct. 2016), p. 13317. arXiv: 1601.03833.

[137] D. Barredo, S. de Léséleuc, V. Lienhard, T. Lahaye and A. Browaeys. 'An Atom-by-atom Assembler of Defect-free Arbitrary Two-dimensional Atomic Arrays'. In: *Science* 354.6315 (2016), pp. 1021–1023. eprint: https://science.sciencemag.org/content/354/6315/1021.full.pdf.

[138] A. Browaeys and T. Lahaye. 'Many-body Physics with Individually Controlled Rydberg Atoms'. In: *Nature Physics* 16.2 (Jan. 2020), pp. 132–142. arXiv: 2002.07413.

[139] M. Saffman, T. G. Walker and K. Mølmer. 'Quantum Information with Rydberg Atoms'. In: *Reviews of Modern Physics* 82.3 (July 2010), pp. 2313–2363. arXiv: 0909.4777.

[140] S. Stellmer, B. Pasquiou, R. Grimm and F. Schreck. 'Laser Cooling to Quantum Degeneracy'. In: *Physical Review Letters* 110.26, 263003 (June 2013), p. 263003. arXiv: 1301.4776.

[141] X. Zhang, M. Bishof, S. L. Bromley, C. V. Kraus, M. S. Safronova, P. Zoller, A. M. Rey and J. Ye. 'Spectroscopic Observation of SU(N)-symmetric Interactions in Sr Orbital Magnetism'. In: *Science* 345.6203 (Aug. 2014), pp. 1467–1473.

[142] M. J. M. L. O. Godart. 'Stochastic Theory of Quantum Mechanics'. In: *arXiv e-prints* (June 2012). arXiv: 1206.2917.

[143] M. Godart. 'Stochastic Theory of Quantum mechanics and the Schrödinger Equation'. In: *arXiv e-prints* (Mar. 2016). arXiv: 1603.08966.

[144] M. Godart. 'Stochastic Theory of Relativistic Quantum Mechanics'. In: *arXiv e-prints* (Dec. 2014). arXiv: 1412.1508.

[145] I. Fényes. 'Eine Wahrscheinlichkeitstheoretische Begründung und Interpretation der Quantenmechanik'. In: *Zeitschrift fur Physik* 132.1 (Feb. 1952), pp. 81–106.

[146] E. Nelson. 'Derivation of the Schrödinger Equation from Newtonian Mechanics'. In: *Phys. Rev.* 150 (4 Oct. 1966), pp. 1079–1085.

Solving Correlation Clustering with the Quantum Approximate Optimisation Algorithm

J.R.Weggemans, 2020