Quantum-inspired and Quantum Optimization on a Superconducting Quantum Processor

by

William P. Banner

B.S., Virginia Polytechnic Institute and State University (2020)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2023

Authored by: William P. Banner
Department of Electrical Engineering and Computer Science
May 19, 2023

Certified by: William D. Oliver
Professor of Electrical Engineering and Computer Science
Professor of Physics
Thesis Supervisor

Accepted by: Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Quantum-inspired and Quantum Optimization on a Superconducting Quantum Processor

by

William P. Banner

Submitted to the Department of Electrical Engineering and Computer Science
on May 19, 2023, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

Quantum and Quantum-inspired optimization represent rapidly growing fields that combine classical optimization techniques with either quantum-inspired ideas or quantum hardware to address complex optimization problems. This thesis provides an overview of quantum-inspired optimization as well as quantum optimization, including the theoretical underpinnings of both processes on hardware and in software. In particular, this thesis considers a specific, practically relevant problem, a BMW production planning problem, and evaluates the performance of quantum-inspired optimizers. This evaluation is implemented by comparing the performance of a family of quantum-inspired optimizers with that of several common black-box combinatorial methods. We find that the use of important operations research techniques including the incorporation of domain-specific information as well as state-space pruning improves the performance of all solvers. In addition, we find that in a majority of tested cases, quantum-inspired methods tie or improve upon the results of their conventional counterparts, albeit by small margins, particularly in regimes of moderate state-space size. This thesis demonstrates that quantum-inspired optimization can outperform many conventional optimization methods in some cases, motivating future use and study of quantum-inspired protocals as well as implementation of fully-quantum optimization techniques.

Thesis Supervisor: William D. Oliver
Professor of Electrical Engineering and Computer Science
Professor of Physics
Thesis Supervisor

Acknowledgments

I would first like to thank my mentors within the EQuS Group. I cannot express enough my gratitude to my mentor, Dr. Terry P. Orlando, who patiently gave invaluable and insightful feedback on this thesis. I would also like to thank my friend and mentor Dr. Tim Menke, who first introduced me to this project and worked with me to develop key concepts and understanding. I appreciate the helpful advice, consistent encouragement, and always-open-door of my mentor Dr. Jeffrey A. Grover. His presence has been an enormous help across multiple projects. I would also like to thank my thesis advisor, Professor William D. Oliver, who helped to onboard me to this program and whose incisive comments and feedback greatly clarified and honed this project.

I would also like to thank my industry collaborators on this project from both Zapata Computing and BMW Inc. In particular, Dr. Shima Bab Hadiashar, Professor Grzegorz Mazur, and Dr. Marcin Ziolkowski. All three met regularly throughout this work providing consistent input and feedback. This is in addition to their unique and inspired technical contributions such as the novel production-guided binary encoding as well as significant data analysis and interpretation. Without this group, this project would not have been possible. Other mentors from Zapata Computing also played significant roles in this work including Dr. Yudong Cao and Dr. Jhonathan Romero, who provided direction at intermediate junctures and feedback on our final results, as well as Dr. Alejandro Perdomo-Ortiz. Dr. Marta Mauri, and the Zapata QML team for helpful discussions throughout this work.

Finally, I would like to thank my friends and family. My first-year roommates and friends outside of lab including Duncan Lee, Michael Fernandez, Nico Studen, and Andrew Wong made my time outside of lab memorable and fun. My friends in lab, in particular my incoming classmates Aziza Almanakly, Junyoung An, and Andy Ding, my office mates Lamia Ateshian and Tim Menke, as well as my other friends within

the EQuS group made coming to lab one of the best parts of my day. I also could not have completed this work without the continuous support of my family. These were the relationships that I relied on the most during the past few years, especially during the COVID pandemic. I am more grateful than I can convey. Finally, I would like to express my love and gratitude to my partner, Tishara Garg. I wish I had known her sooner and asked for her help more. Just the opportunity to meet her made my entire time at MIT worth it and I am excited to continue working on my PhD alongside her.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Quantum mechanics is a postulated fundamental theory in physics that describes the behavior of matter and energy on the smallest scales, such as atoms and subatomic particles [5, 6]. It is a field that has revolutionized our understanding of the natural world, challenging many of our traditional intuitions. Despite its abstract nature, quantum mechanics has proven to be an extremely accurate theory, with predictions that have been confirmed by numerous experiments. This in turn has led to the development of many groundbreaking technologies, including transistors, lasers, and, more recently, quantum computers [7, 8, 9].

This most recent resulting technology has so far progressed as many of those quantum technologies that came before it- it started as an academic endeavor. When faced with the intractable nature of many large quantum mechanical problems, Richard Feynman, Yuri Manin, and others who came after, wondered if maybe a fundamentally quantum simulator could be coded to provide the exact results directly, circumventing expensive classical calculations. The application of interest was therefore many-body physics and to gain a better understanding of the universe [10].

Over the proceeding four decades, this goal changed substantially. Beginning with Shor's algorithm in 1994 and with Grover's algorithm in 1996 the community and the wider world with it, realized that quantum computers may act as far more useful

tools than academic curiosities [11, 12]. Since then, a number of possible "practical" (or "everyday-impact") algorithms have been proposed. In addition, many of these algorithms fall under the umbrella of near-term intermediate-scale quantum (NISQ) algorithms [13]. In other words, they could theoretically be performed on the rather far-from-ideal quantum computers available today. The most famous and successful of these include the variational quantum eigensolver (VQE), the quantum approximate optimization algorithm (QAOA), quantum annealing, and quantum machine learning methods also called quantum Born machines [14, 15, 16, 17].

There have been some claims of quantum advantage shown using these algorithms, although theoretical guarantees are unclear. Under certain assumptions, including exact computation of expectation values, VQE's scale polynomially as compared with exponential classical counterparts [18]. QAOA and quantum annealing, on the other hand, have no such guarantee and instead fall under the label of heuristic solvers: they may perform well for some specialized problems. Some of these problems have already been identified for quantum annealing [19, 20], while realistic estimates for QAOA advantage requirements are still typically beyond the capabilities of modern processors [21, 22]. Born Machines may display some quantum advantage, although the notion for advantage with these methods are often less well-defined as defining machine learning success depends heavily on the types of distance metrics used as well as the types of classical optimization methods used to augment it. At the very least however, it has been shown that certain machine-learning tasks using born machines have achieved better performance than conventional methods while exploiting exponentially-scaling models [23].

In addition to the scientific advancements generated via development of quantum hardware, several mathematical developments occurred which fundamentally restructured our understanding of information manipulation and storage. The culmination of these was the idea of tensor networks [24]. These networks are compact data struc-

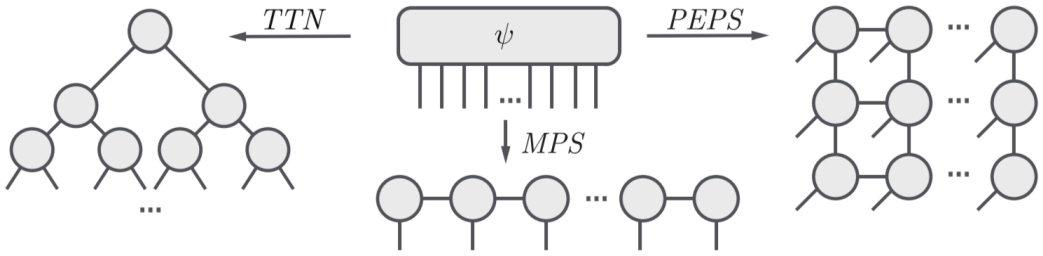


Figure 1-1: Examples of tensor networks commonly found in literature. On the left is an example of a tree tensor network. In the center is an example of a matrix product state (relevant for this work), and on the right is a projected entangled pairs state [1].

tures which originally found wide use in the classic Density Matrix Renormalization Group (DMRG) algorithm, useful for exactly solving linear quantum models, as shown in Fig. 1-1. These tensor networks are often referred to as "quantum-inspired" techniques as, although they are entirely classical objects, they draw their intuition in part from quantum mechanics. It was from these objects that "quantum-inspired" optimization methods were developed [25, 26]. The focus of this thesis is to expand upon quantum-inspired and, to a lesser extent quantum opimization, and to describe the results of quantum-inspired optimization techniques to a particular problem.

1.1 Optimization Using Generative Models

The fundamental idea behind the black-box quantum and quantum-inspired optimization methods described in this thesis work is to utilize generative models. By black-box optimization, it is meant optimization of an objective function or problem with unknown structure. Generative models are mathematical models of conditional probabilities $Q(X|Y = y)$.Output pairs (x,y) can be generated from the model via choosing a random y and generating a random x from the distribution provided by the model and y. Typically, generative models are parameterized, or are modifiable by some external parameters θ and can therefore be represented as $Q(X|Y, \theta)$. The parameters θ are then trained such that the distribution Q matches some desired

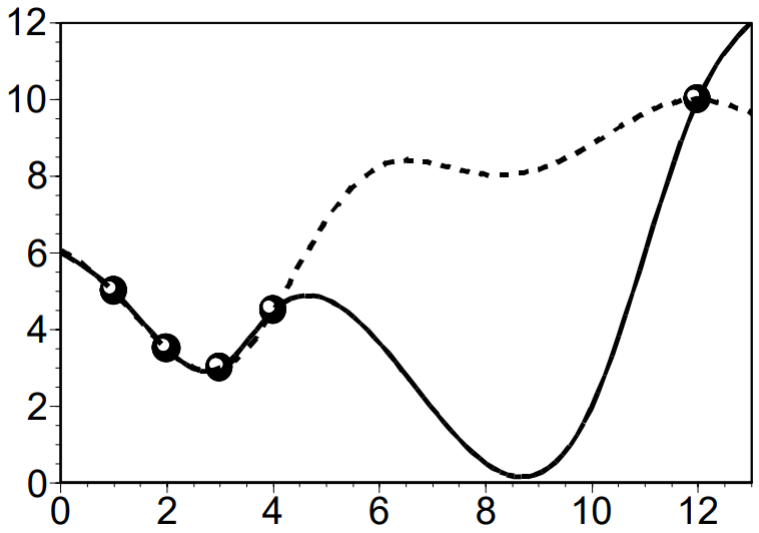


Figure 1-2: An illustration of Bayesian optimization, which uses a stochastic model of assumed correlations in a problem to predict local minima. The solid line represents the true objective function while the dotted line represents a trained model.

distribution $P(x|y)$. For a brief review of this topic, see [27].

Since the late 1990s, researchers have attempted to use generative models to learn the structure of black-box models, making them no-longer black-box and therefore easier to solve. As an example, see Fig. 1-2 [28]. This idea has become more popular, however, now that significant helpful advances have been made in computing power, machine learning algorithms, and, for the quantum case, quantum hardware. Many of these recent uses of generative models have fallen under the label of "Generator-Enhanced Optimization" methods, or GEO, which specifically combines conventional black-box optimization methods with quantum-inspired and quantum optimization routines [26]. In the case of quantum-inspired GEO routines, tensor networks are used as the back-end of the model. In the case of quantum GEO routines, parameterized quantum circuits, in the style of variational quantum algorithms like VQE, form the back-end of the generative model.

For the above approaches, the type of hardware used matters little and in theory they can be implemented on any quantum computing modality. In practice, varia-

tional algorithms have only been implemented on superconducting qubits and trapped ions to the best of our knowledge [29, 30]. For the purposes of this work and the proposed quantum implementation in Chaper 5, we will only consider superconducting quantum circuits. These circuits were the first modality to achieve verifiable quantum advantage, can often be analyzed using classical circuit techniques and are fabricated using well-established lithographic methods [31, 32]. Additional details about the fundamentals of superconducting circuits can be found in Chapter 2.

1.2 Optimization of the BMW Production Planning Problem

Although (quantum and quantum-inspired) model-based optimization methods have become popular in the literature, it is still unclear precisely the advantage provided as compared to conventional techniques. This is in large part due to the lack of theoretical guarantees for these types of optimization routines. Characterization of optimizer performance therefore requires a careful problem-by-problem evaluation.

The purpose of this thesis is to perform a characterization of quantum-inspired model-based optimization techniques and to propose characterization of quantum model-based techniques for a particular problem as part of this problem-by-problem line of work. The particular problem that we investigate is a production planning problem based on BMW's current manufacturing facilities. In this problem, a series of shops are set up in an assembly line, and the goal is to choose the individual working hours or shifts and rates of production of these shops throughout a week. These shifts and rates should be chosen such that the assembly line meets a series of monthly production targets and minimizes the amount of time that shops are left idle (due to production backlogs) when they are supposed to be working. The distance from production targets and number of idle hours are calculated based on costly year-long simulations of the assembly line operation, as showin in Fig. 1-3. Additional details

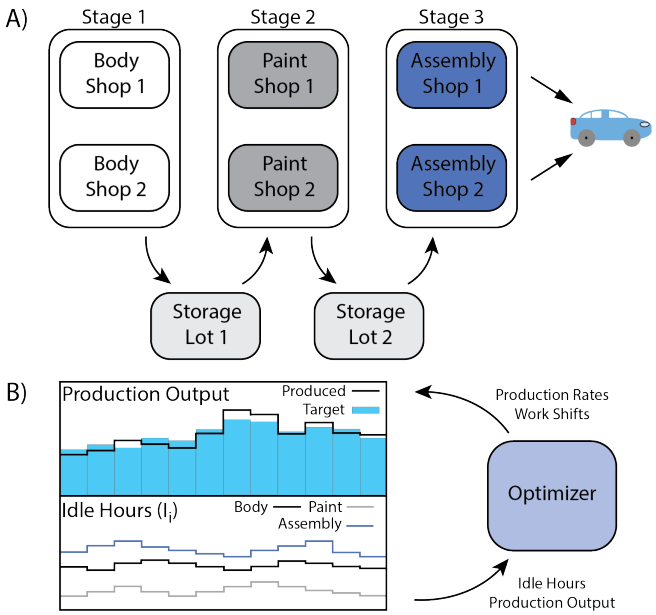


Figure 1-3: A) A high-level view of the production planning problem with shops and storage labeled. B) A stylized diagram of the optimization procedure. First, the parameters of the optimization, in the form of production rates and work shifts, are used to perform a time-domain simulation of the assembly line. From this simulation, the number of idle hours for each month and cars produced each month is retrieved. These values are used to generate a cost using the cost function.

of this problem are provided in Chapter 4.2.

Taking this assembly line problem, we restrict the solution space to varying degrees using domain knowledge, then apply a binary encoding to the resulting state space to encode each shop's production rates and shift schedules. We then go on to compare the performance of several quantum-inspired generative model solvers to that of conventional black-box methods such as genetic algorithms, simulated annealing and parallel tempering. Through this study, we find that performance of all solvers improves using specific preprocessing techniques. In addition, we find that relative solver performance depends heavily on the state-space size of the problem with intermediate sizes being best for model-based optimization. In all, we find that quantum-inspired generative models can improve on the solutions found by conventional solvers in a majority of tested cases, with heavy dependence on the type of solver used for pre-training.

In this thesis work, we first provide background on the physics of superconducting hardware with superconducting quantum circuits in Chapter 2. We then provide additional background information on the GEO framework as well as tensor networks and quantum optimization in Chapter 3. We then move on to characterize the performance of quantum-inspired black-box optimization methods when applied to a particular practically-relevant problem, the BMW production planning problem in Chapter 4. Finally, we conclude with an outlook and proposal for future work with a quantum generative optimizer in Chapter 5 based on Chapter 2.

Chapter 2

Fundamentals of Superconducting Quantum Hardware

2.1 The Quantum Mechanical LC Oscillator

The most basic circuit model utilized in quantum information processing is the LC-circuit. This model always reduces to a parallel form and is represented in Fig. 2-1. To analyze the behavior of such a circuit following an electrical engineering approach, we would start with Kirchoff's voltage law (KVL). This law states that the sum of the potentials around a circuit loop must be equal to the external flux through the loop (in this case zero). The voltage across the inductor is $V_L = L\frac{dI}{dt}$ where I is the current through the inductor. The voltage across the capacitor is $V_c = \frac{q}{C}$, where q is

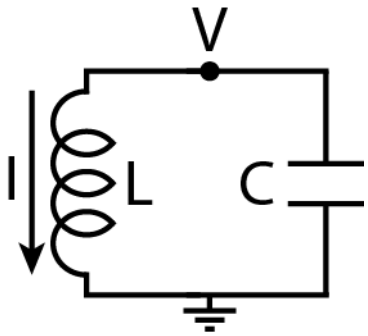


Figure 2-1: A hanging parallel LC Oscillator.

the charge across the capacitor. The sum of (signed) voltages is therefore $\frac{q}{C} - L\frac{\partial I}{\partial t} = 0$. We can now note that, given there is only one node in this circuit, charge is conserved, so $I = -\frac{dq}{dt}$ where I and q are the same as previously defined. Then:

$$\frac{q}{C} + L\frac{d^2q}{dt^2} = 0 \tag{2.1}$$

Solving 2.1 gives us the classical behavior of the charge on the capacitor in time $q = q_0 \cos(\Omega t + c)$ with c an arbitrary phase and $\Omega = \frac{1}{\sqrt{LC}}$ - hence to the name "LC-oscillator".

While the classical behavior is useful from an intuitive perspective, it is helpful to borrow some additional tools from physics before this system can be understood in the context of quantum computation. First, consider the energy stored in the components of this system. In the capacitor the energy can be written as $\frac{1}{2}CV^2 = \frac{q^2}{2C}$ and in the inductor as $\frac{1}{2}LI^2 = L\dot{q}^2/2$. We define the Lagrangian, L as

$$L = T - V = L\dot{q}^2/2 - q^2/2C \tag{2.2}$$

where we have taken the charge q as our generalized "spatial" coordinate with generalized velocity $\dot{q}$. The Lagrangian mechanical viewpoint relies on the idea that the response of dynamic systems, under holonomic constraints (constraints that depend only on position and time), remain stationary (of fixed time-dependent form), with regard to a system's action (the time-integrated Lagrangian). Use of Lagrange's equations $\frac{d}{dt}(\frac{dL}{d\dot{q}}) = \frac{dL}{dq}$, leads to the same KVL equation, with fixed time-dependent form, motivating the physical validity of this viewpoint for circuitry. From this Lagrangian equation we can also define a generalized momentum (sometimes denoted $p = mv = \frac{\partial L}{\partial \dot{q}}$),

$$\frac{\partial L}{\partial \dot{q}} = L\dot{q} = LI = \Phi \tag{2.3}$$

Physically Φ is the magnetic flux through the inductor in the oscillator. The Hamiltonian of such a system, given current-independent potential, is simply equal

to the total system energy (in this case):

$$H = T + V = \frac{1}{2}LI^2 + \frac{1}{2}CV^2 = \frac{\Phi^2}{2L} + \frac{q^2}{2C} \tag{2.4}$$

and can be derived directly from the Lagrangian via $H = \dot{q}\Phi - L$. We note that, had potential energy V and kinetic energy T, instead been defined as the inductive energy and capacitive energy, the same equation would have been derived as the Hamiltonian is invariant of canonical transformations of the form $p \leftrightarrow q$.

Taking our coordinates to the quantum mechanical limit, we can promote our individual coordinates to operators $q \rightarrow \hat{q}$ and $\Phi \rightarrow \hat{\Phi}$. While this is simple enough an operation to write, it has a deeper quantum mechanical meaning.

We used to have a macroscopic system, where two independent real values could fully describe the system parameters at any precise instant in time ($\Phi(t)$ and $q(t)$). Now these coordinates represent the value of individual quantum which are wavelike, not macroscopic averages. Since the position and momentum of a wave are (Fourier) conjugates, they are incompatible and cannot be defined at the same instant. The same is true of Φ and q. Operator notation describes this incompatability better than our prior classical and real variables as operators do not naturally commute in general ($\hat{a}\hat{b} - \hat{b}\hat{a} \neq 0$). For generalized position and momentum ($\hat{\Phi}$ and $\hat{q}$), the commutation relation (representing incompatability) is the Heisenberg uncertainty relation $[\hat{q}, \hat{\Phi}] = i\hbar$. Given this operator notation, the *quantum mechanical* description of the LC oscillator is given by

$$H = \frac{\hat{\Phi}^2}{2L} + \frac{\hat{q}^2}{2C} \tag{2.5}$$

While numerous texts cover the above Hamiltonian, known as the quantum harmonic oscillator (QHO), we will provide a short treatment here. The steady-state dynamics of this system can be solved for directly through the use of the Schrodinger Equation $H\psi = -i\frac{\partial}{\partial t}\psi$ where we have used ψ to represent the quantum wavefunc-

tion, a generalized function representing a steady-state system response. Instead of directly solving the Schrodinger equation, however, better insight can be gained into this system by considering a change of variable. We define ladder operators $\hat{a}, \hat{a}^\dagger$ where $\hat{a} = \frac{i}{\sqrt{2L\hbar\Omega}}\hat{\Phi} + \frac{1}{\sqrt{2C\hbar\Omega}}\hat{q}$ and Ω is the fundamental frequency $\frac{1}{\sqrt{LC}}$. These ladder operators obey the commutation relation $[\hat{a}, \hat{a}^\dagger] = 1$ (one can use the commutation relation for $[\hat{\Phi}, \hat{q}]$ to check). From these we have that $\hat{\Phi} = \sqrt{\frac{L\hbar\Omega}{2}}(\hat{a} + \hat{a}^\dagger)$ and $\hat{Q} = -i\sqrt{\frac{C\hbar\Omega}{2}}(\hat{a} - \hat{a}^\dagger)$. The prefactors $\sqrt{\frac{L\hbar\Omega}{2}}$ and $\sqrt{\frac{C\hbar\Omega}{2}}$ are known as the zero point fluctuations of flux and charge respectively and and can be seen as arising from the uncertainty relations of charge and flux in a vacuum. This allows us to rewrite the QHO Hamiltonian as:

$$\begin{aligned} H &= \frac{\hat{\Phi}^2}{2L} + \frac{\hat{q}^2}{2C} \\ &= \frac{\left(\sqrt{\frac{L\hbar\Omega}{2}}(\hat{a} + \hat{a}^\dagger)\right)^2}{2L} + \frac{\left(-i\sqrt{\frac{C\hbar\Omega}{2}}(\hat{a} - \hat{a}^\dagger)\right)^2}{2C} \\ &= \hbar\Omega(\hat{a}^\dagger\hat{a} + \frac{1}{2}) \end{aligned} \tag{2.6}$$

Such a Hamiltonian is diagonal with respect to "ladder" eigenstates or Fock states, enumerating the energy levels of the Hamiltonian.These levels are discrete and evenly spaced by energy $\hbar\Omega$ and, the states can be represented as "bras" ($\langle n|$) and "kets" ($|n\rangle$) in the typical "bra-ket" notation. These levels can be navigated, then, via these same ladder operators such that $\hat{a}\,|1\rangle = |0\rangle$ and $\hat{a}^\dagger\,|0\rangle = |1\rangle$.

Pictorally, this system can be represented as in Fig. 2-2. The equations for traditional physical quantities, such as voltage V and current I can now be written in relation to the newly defined ladder operators. This provides insight into how these quantities can be viewed with respect to a harmonic system.

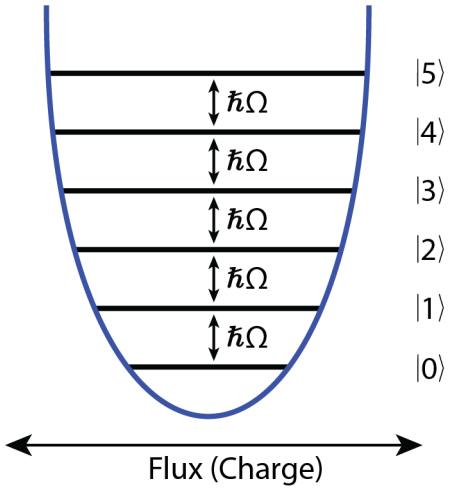


Figure 2-2: (black) The energy-level diagram of the Quantum Harmonic Oscillator (QHO) as functions of flux (or charge). (blue) the classically predicted energy of the quantum harmonic oscillator as a function of flux (or charge).

$$V = \frac{\hat{q}}{C} \text{ (by the definition of capacitance)} \qquad I = \frac{\hat{\Phi}}{L} \text{ (by the definition of flux)}$$
$$= \frac{-i\sqrt{\frac{C\hbar\Omega}{2}}(\hat{a} - \hat{a}^\dagger)}{C} \qquad = \frac{\sqrt{\frac{L\hbar\Omega}{2}}(\hat{a} + \hat{a}^\dagger)}{L}$$
$$= -i\sqrt{\frac{\hbar\Omega}{2C}}(\hat{a} - \hat{a}^\dagger) \qquad = \sqrt{\frac{\hbar\Omega}{2L}}(\hat{a} + \hat{a}^\dagger)$$

These quantities do not commute with the Hamiltonian, indicating that measurement of the system energy leads to a natural uncertainty in these values (or vice versa).

At this point it is often helpful to ground these systems in a numerical intuition if the physical intuition provided above has failed. Typically, we numerically represent quantum states as vectors in $\mathbb{R}^n$ with operators often defined as matrices in $\mathbb{R}^{n\times n}$ as neither operators nor matrices commute in general. Additionally, linear, finite operations, such as matrix vector operations, are typically simple to specify on classical computers. Unfortunately, quantum mechanics often requires us to simulate operators that are neither linear nor finite, which means that most matrix-vector representations will naturally have some truncation error. The quantum harmonic oscillator is

a good example of a system that is linear, but infinite, and we will see later examples of systems that are nonlinear and infinite. Numerically simulating an infinite system is often very time-consuming, as the accuracy of the simulation is often limited by the simulation (matrix) size. For this reason, we restrict many quantum simulations to just the single-qubit ($|0\rangle$ and $|1\rangle$) subspace. Given the previously defined ladder operators: $\hat{a}\,|1\rangle = |0\rangle$ and $\hat{a}^\dagger\,|0\rangle = |1\rangle$ we can rewrite these as matrices in the Fock basis with

$$\hat{a} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \qquad\qquad \hat{a}^\dagger = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

with quantum states $|0\rangle$ and $|1\rangle$ defined as

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad\qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

We can then write our Hamiltonian numerically as

$$H = \begin{bmatrix} 0 & 0 \\ 0 & \hbar\Omega \end{bmatrix}$$

where we have excluded the $\frac{1}{2}$ term as it is just a constant offset, so it doesn't change the system behavior. The dynamics of our system are typically solved using master equation solvers, which, in the steady state without dissipation, take the form of the Ehrenfest theorem:

$$\dot{\rho} = \frac{1}{i\hbar}[\rho, H]$$

Where ρ is the "density matrix", an outer-product description of the wavefunction. Typically, these systems do experience decay over time, which can be more accurately modeled by adding decay terms to the above differential equation. These types of

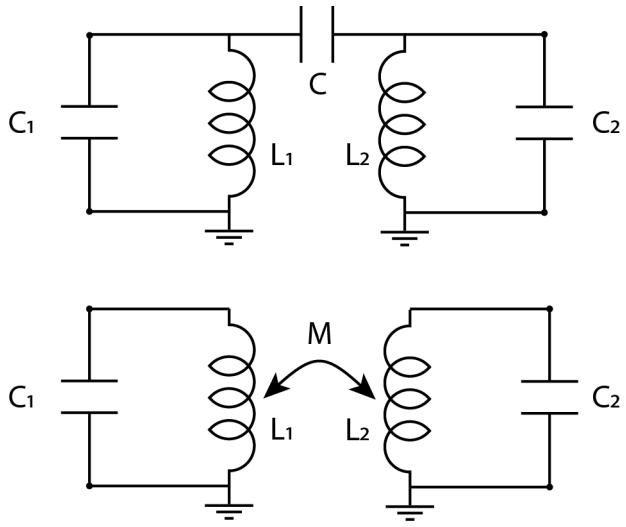


Figure 2-3: Electromagnetic oscillators are coupled via either a coupling capacitor or a mutual inductance.

effects can be accurately simulated using common python libraries such as QuTiP.

2.2 Circuit Quantum Electrodynamics

Circuit Quantum Electrodynamics (cQED) is the theory of the time-domain behavior of electromagnetic systems in the quantum regime. For such dynamics to occur, multiple non-orthogonal modes need to be present, such that energy can pass between them. This brings us then to the topic of coupled oscillators. Fortunately, the tools that we just used to study individual quantum harmonic oscillators extend well. The only new bit of circuitry is the coupling element. Coupling between electromagnetic oscillators can come in two distinct forms: a mutual inductance between the oscillator inductors, or a coupling capacitance between the oscillator voltage nodes. Both are shown in Fig. 2-3.

Algebraically, such a coupling (capacitive or inductive) can be easily represented via an additional energy term in the Hamiltonian, the energy of the coupling element. For capacitive coupling, we can use the typical energy expression, but substitute the

proper operators representing voltage:

$$H_{coup} = \frac{1}{2}C\Delta V^2 = \frac{1}{2}C(V_1 - V_2)^2 = \frac{1}{2}C\Big(-i\sqrt{\frac{\hbar\Omega_1}{2C_1}}(\hat{a_1} - \hat{a_1}^\dagger) + i\sqrt{\frac{\hbar\Omega_2}{2C_2}}(\hat{a_2} - \hat{a_2}^\dagger)\Big)^2$$

Where $\hat{a_1}$ and $\hat{a_2}$ are the lowering operators for oscillator 1 and oscillator 2, respectively. Likewise, the energy across a mutual inductance can be given by the inductive energy equation:

$$H_{coup} = MI_1I_2 = M\Big(\sqrt{\frac{\hbar\Omega_1}{2L_1}}(\hat{a_1} + \hat{a_1}^\dagger)\Big)\Big(\sqrt{\frac{\hbar\Omega_2}{2L_2}}(\hat{a_2} + \hat{a_2}^\dagger)\Big)$$

Combining these two expressions leads to the full Hamiltonian for coupled oscillators:

Capacitive:

$$\hbar\Omega_1\hat{a_1}^\dagger\hat{a_1} + \hbar\Omega_2\hat{a_2}^\dagger\hat{a_2} + \frac{1}{2}C\Big(-i\sqrt{\frac{\hbar\Omega_1}{2C_1}}(\hat{a_1} - \hat{a_1}^\dagger) + i\sqrt{\frac{\hbar\Omega_2}{2C_2}}(\hat{a_2} - \hat{a_2}^\dagger)\Big)^2$$

Inductive:

$$\hbar\Omega_1\hat{a_1}^\dagger\hat{a_1} + \hbar\Omega_2\hat{a_2}^\dagger\hat{a_2} + M\Big(\sqrt{\frac{\hbar\Omega_1}{2L_1}}(\hat{a_1} + \hat{a_1}^\dagger)\Big)\Big(\sqrt{\frac{\hbar\Omega_2}{2L_2}}(\hat{a_2} + \hat{a_2}^\dagger)\Big)$$

Classically, coupled oscillators hybridize to oscillate at different frequencies than their uncoupled frequencies. This can be seen explicitly in our quantum system after multiplying out the coupling terms and re-diagonalizing. We will do this in a step-by-step manner as follows:

Capacitive:

$$
\begin{aligned}
H &= \hbar\Omega_1 \hat{a_1}^\dagger \hat{a_1} + \hbar\Omega_2 \hat{a_2}^\dagger \hat{a_2} + \frac{1}{2}C\Big(-i\sqrt{\frac{\hbar\Omega_1}{2C_1}}(\hat{a_1} - \hat{a_1}^\dagger) + i\sqrt{\frac{\hbar\Omega_2}{2C_2}}(\hat{a_2} - \hat{a_2}^\dagger)\Big)^2 \\
&= \hbar\Omega_1 \hat{a_1}^\dagger \hat{a_1} + \hbar\Omega_2 \hat{a_2}^\dagger \hat{a_2} + \frac{1}{2}\Big(-\frac{\hbar\Omega_1}{2C_1}(\hat{a_1}^2 - \hat{a_1}\hat{a_1}^\dagger - \hat{a_1}^\dagger\hat{a_1} + \hat{a_1}^{\dagger 2}) \\
&- \frac{\hbar\Omega_2}{2C_2}(\hat{a_2}^2 - \hat{a_2}\hat{a_2}^\dagger - \hat{a_2}^\dagger\hat{a_2} + \hat{a_2}^{\dagger 2}) \\
&+ \sqrt{\frac{\hbar\Omega_1}{2C_1}}\sqrt{\frac{\hbar\Omega_2}{2C_2}}(\hat{a_1}\hat{a_2} + \hat{a_2}\hat{a_1} + \hat{a_1}^\dagger\hat{a_2}^\dagger + \hat{a_2}^\dagger\hat{a_1}^\dagger + \hat{a_1}\hat{a_2}^\dagger + \hat{a_2}^\dagger\hat{a_1} + \hat{a_1}^\dagger\hat{a_2} + \hat{a_2}\hat{a_1}^\dagger)\Big)
\end{aligned}
$$

Inductive:

$$
\begin{aligned}
H &= \hbar\Omega_1 \hat{a_1}^\dagger \hat{a_1} + \hbar\Omega_2 \hat{a_2}^\dagger \hat{a_2} + M\Big(\sqrt{\frac{\hbar\Omega_1}{2L_1}}(\hat{a_1} + \hat{a_1}^\dagger)\Big)\Big(\sqrt{\frac{\hbar\Omega_2}{2L_2}}(\hat{a_2} + \hat{a_2}^\dagger)\Big) \\
&= \hbar\Omega_1 \hat{a_1}^\dagger \hat{a_1} + \hbar\Omega_2 \hat{a_2}^\dagger \hat{a_2} + M\sqrt{\frac{\hbar\Omega}{2L_1}}\sqrt{\frac{\hbar\Omega}{2L_2}}(\hat{a_1}\hat{a_2} + \hat{a_1}^\dagger\hat{a_2} + \hat{a_1}\hat{a_2}^\dagger + \hat{a_1}^\dagger\hat{a_2}^\dagger)
\end{aligned}
$$

Using the identity $[\hat{a}, \hat{a}^\dagger] = 1$, the terms of the form $\hat{a}\hat{a}^\dagger$ can be converted to $1 + \hat{a}^\dagger\hat{a}$. This process is known as operator normal-ordering. This alternative form is easier to understand as a frequency shift. In addition, we can typically ignore terms of the form $\hat{a}^2$ or $\hat{a}^{\dagger 2}$ as these terms only directly impact higher-level states ($|2\rangle$ or higher) and therefore only indirectly impact the typical states of interest ($|0\rangle$ and $|1\rangle$ subspace). When considering a microwave drive applied to an individual oscillator - the dropping of $\hat{a}^2$ and $\hat{a}^{\dagger 2}$ terms is known as the rotating wave approximation (RWA). This assumption does not hold if the prefactor of the $\hat{a}^2$ term is of similar magnitude to the $\hat{a}^\dagger\hat{a}$ term, although the system dynamics can still be approximated by considering a slight energy shift known as the Bloch-Seigert shift. We have also kept in the mind that $\hat{a_1}$ and $\hat{a_2}$ commute (as they are associated with the uncoupled oscillators and act on completely different systems). We now have our coupled, normalized, RWA Hamiltonian (ignoring constant terms):

Capacitive:

$$\begin{aligned} H &= \hbar\Omega_1 \hat{a_1}^\dagger \hat{a_1} + \hbar\Omega_2 \hat{a_2}^\dagger \hat{a_2} - \frac{C}{2}\Big(\frac{\hbar\Omega_1}{2C_1}(2\hat{a_1}^\dagger \hat{a_1}) + \frac{\hbar\Omega_2}{2C_2}(2\hat{a_2}^\dagger \hat{a_2}) + 2\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sqrt{\frac{\hbar\Omega_2}{2C_2}}(\hat{a_2}^\dagger \hat{a_1} + \hat{a_1}^\dagger \hat{a_2})\Big) \\ &= \hbar\Omega_1 \hat{a_1}^\dagger \hat{a_1} - C\frac{\hbar\Omega_1}{2C_1}\hat{a_1}^\dagger \hat{a_1} + \hbar\Omega_2 \hat{a_2}^\dagger \hat{a_2} - C\frac{\hbar\Omega_2}{2C_2}\hat{a_2}^\dagger \hat{a_2} + C\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sqrt{\frac{\hbar\Omega_2}{2C_2}}(\hat{a_2}^\dagger \hat{a_1} + \hat{a_1}^\dagger \hat{a_2}) \\ &= \big(\hbar\Omega_1 - C\frac{\hbar\Omega_1}{2C_1}\big)\hat{a_1}^\dagger \hat{a_1} + \big(\hbar\Omega_2 - C\frac{\hbar\Omega_2}{2C_2}\big)\hat{a_2}^\dagger \hat{a_2} + C\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sqrt{\frac{\hbar\Omega_2}{2C_2}}(\hat{a_2}^\dagger \hat{a_1} + \hat{a_1}^\dagger \hat{a_2}) \end{aligned} \tag{2.7}$$

Inductive:

$$H = \hbar\Omega_1 \hat{a_1}^\dagger \hat{a_1} + \hbar\Omega_2 \hat{a_2}^\dagger \hat{a_2} + M\sqrt{\frac{\hbar\Omega_1}{2L_1}}\sqrt{\frac{\hbar\Omega_2}{2L_2}}(\hat{a_1}^\dagger \hat{a_2} + \hat{a_1}\hat{a_2}^\dagger) \tag{2.8}$$

For numerical purposes, in matrix form, such a system (in the $|0\rangle$ and $|1\rangle$ excitation subspace) looks like:

Capacitive:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \hbar\Omega_1 - C\frac{\hbar\Omega_1}{2C_1} & 2C\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sqrt{\frac{\hbar\Omega_2}{2C_2}} & 0 \\ 0 & 2C\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sqrt{\frac{\hbar\Omega_2}{2C_2}} & \hbar\Omega_2 - C\frac{\hbar\Omega_2}{2C_2} & 0 \\ 0 & 0 & 0 & \hbar\Omega_1 - C\frac{\hbar\Omega_1}{2C_1} + \hbar\Omega_2 - C\frac{\hbar\Omega_2}{2C_2} \end{bmatrix}$$

Inductive:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \Omega_1 & M\sqrt{\frac{\hbar\Omega_1}{2L_1}}\sqrt{\frac{\hbar\Omega_2}{2L_2}} & 0 \\ 0 & M\sqrt{\frac{\hbar\Omega_1}{2L_1}}\sqrt{\frac{\hbar\Omega_2}{2L_2}} & \Omega_2 & 0 \\ 0 & 0 & 0 & \Omega_1 + \Omega_2 \end{bmatrix}$$

Where we have taken as a basis for the coupled system a binary encoding of the two uncoupled systems $|00\rangle, |01\rangle, |10\rangle, |11\rangle$. Modelling in this way, we can note the so-called "lamb-shift" in the capacitive-coupled matrix. This is the apparent shift in the frequencies of the "uncoupled" oscillator due simply to the existence of the vacuum

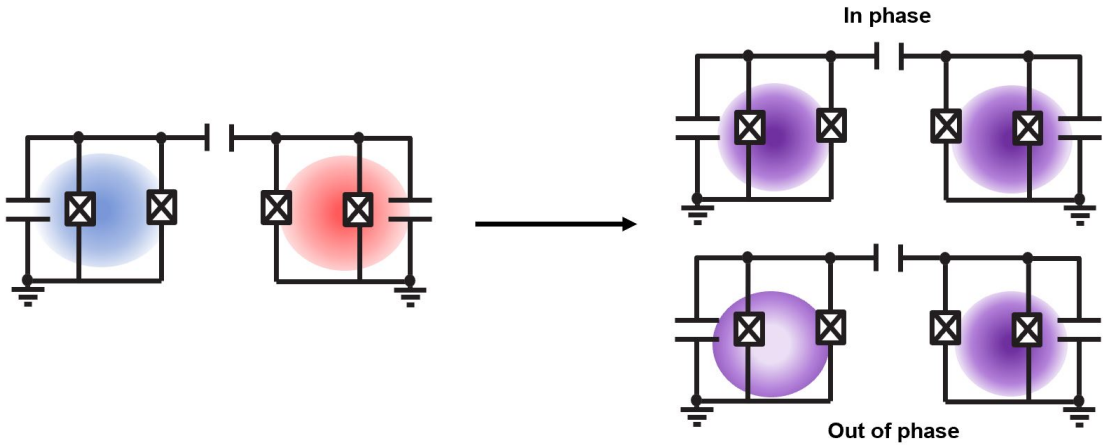


Figure 2-4: Independent oscillators oscillate at their individaul (red and blue) frequencies and phases. When coupled, these frequencies shift/combine (purple) and sync their phases to form either in-phase or out-of-phase modes in the steady state.

fluctuations of the other oscillator. This shift is a hallmark of quantum mechanics.

It is important to note that nothing in the approach to coupling, (except for non-RWA behavoir), is nonlinear- i.e., most everything shown in the inductive and capacitive matrices above (when ignoring the implications of vacuum fluctuations), can be applied to classical coupled linear oscillators. This means that we can borrow from these classical ideas to better understand what coupling our quantum oscillators has achieved. The primary terms that are new in our above matrices are the off-diagonal ($\langle 01| H |10\rangle$ and $\langle 10| H |01\rangle$) terms. If the coupling were not present, these terms would not exist, and the matrix would be diagonal with entries corresponding to the uncoupled oscillator frequencies. Now that these off-diagonal coupling terms are present, if we were to diagonalize these matrices (as we will do later), we would see the diagonal elements shift as compared to the uncoupled frequencies. This shifting of frequencies can be entirely be explained classically, as mode hybridization, where the new oscillation modes are the in-phase and out-of-phase oscillations in the coupled system - diagrammatically shown in Fig. 2-4

In diagonalizing, we replace the off-diagonal elements by a general coupling strength g and the diagonals by effective frequencies (including lamb shift) $\tilde{\Omega}_i$. So the general

coupled harmonic oscillator Hamiltonian for two two-level systems can be written as:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \tilde{\Omega}_1 & g & 0 \\ 0 & g & \tilde{\Omega}_2 & 0 \\ 0 & 0 & 0 & \tilde{\Omega}_1 + \tilde{\Omega}_2 \end{bmatrix} \tag{2.9}$$

This can be diagonalized analytically to obtain the frequencies of the hybridized modes:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{\tilde{\Omega}_1+\tilde{\Omega}_2-\sqrt{(\tilde{\Omega}_1-\tilde{\Omega}_2)^2+4g^2}}{2} & 0 & 0 \\ 0 & 0 & \frac{\tilde{\Omega}_1+\tilde{\Omega}_2+\sqrt{(\tilde{\Omega}_1-\tilde{\Omega}_2)^2+4g^2}}{2} & 0 \\ 0 & 0 & 0 & \tilde{\Omega}_1 + \tilde{\Omega}_2 \end{bmatrix} \tag{2.10}$$

These hybridized frequencies are simply an average of the individual frequencies, offset by an additional weighting due to the detuning $(\tilde{\Omega}_1 - \tilde{\Omega}_2)$ and coupling strength g. Note that when the detuning is 0, the difference in energies between the two levels is $2g$.

2.3 Readout and Control of Quantum Systems

2.3.1 Readout Intuition

Before discussing readout, it is often helpful to first describe the basic concept as it relates directly to the Hamiltonian just discussed. Readout is the process of learning information from a quantum system, typically performed by measuring the energy stored in a quantum bit. This measurement can be performed directly if one were to couple a qubit directly to a waveguide, a medium that hosts traveling photonic states. If we assume the use of a good single photon detector, then the energy in the qubit could be transferred from the qubit into the waveguide, travel as a photon, and be detected via a single-photon detector [33]. Unfortunately, there are two problems with

this type of setup. First, it requires the existence of an ideal single-photon detector. The energy contained within a single photon at typical microwave frequencies is so small (tens of μeV) that such a device is very difficult to realize in practice. Second, the setup requires projection of the information that is being measured. In other words, once the energy contained in the qubit leaks into the waveguide and is detected, it no longer exists in the qubit and cannot be measured again. This means that the system needs to be identically prepared before repeating the measurement.

The alternative to this measurement scheme is quantum non-demolition (QND) measurement. In this case, the quantum state representing the measured value (post-measurement) does not change during or after the measurement operation. I note that the term "post-measurement" is important. The quantum state still collapses and the qubit still enters one of its eigenstates, it's simply that, after this occurs, QND measurements allow the qubit to remain in this eigenstate. An example of such a measurement for a free particle (not a QHO) would be a momentum measurement, as this operator commutes with the free particle Hamiltonian.

In practice, QND measurements are often performed using an intermediate pointer system that is longitudinally coupled to the qubit. Measurement of the pointer system then provides information about the qubit without providing a decay path for the qubit excitation. When representing the qubit and pointer as quantum circuits, we can use coupled detuned quantum harmonic oscillators, as given in Chapter 2.2. The Hamiltonian for this system is the same as before, however, in order to avoid the qubit decaying into a coupled channel, we make the pointer frequency far detuned from qubit frequency ($\tilde{\Omega}_2 > \tilde{\Omega}_1$) and make the coupling strength g much less than either frequency $\tilde{\Omega}_1$ or $\tilde{\Omega}_2$. This condition is often called the "dispersive limit". Taylor expanding the square roots in our matrix equation, we find that the diagonal entries become $\tilde{\Omega}_1 - \frac{g^2}{\delta}$ and $\tilde{\Omega}_2 + \frac{g^2}{\delta}$ where δ is the detuning between the uncoupled modes.

In looking at decay of the qubit information out of the pointer state, this form of coupling is ideal. The eigenstates of the coupled system in the despersive limit can be found by diagonalizing just the $|01\rangle$ and $|10\rangle$ subspace of the coupled two-level Hamiltonian as there are no other off-diagonal elements in the original four-by-four

matrix:

$$H_{|01\rangle,|10\rangle} = \begin{bmatrix} \tilde{\Omega_1} & g \\ g & \tilde{\Omega_2} \end{bmatrix} \tag{2.11}$$

The eigenstates of this hamiltonian can then be given as

$$\psi^+ = \begin{bmatrix} \cos\Big(\tan^{-1}(\frac{g}{\delta})\Big) \\ \sin\Big(\tan^{-1}(\frac{g}{\delta})\Big) \end{bmatrix} \quad \psi^- = \begin{bmatrix} -\sin\Big(\tan^{-1}(\frac{g}{\delta})\Big) \\ \cos\Big(\tan^{-1}(\frac{g}{\delta})\Big) \end{bmatrix} \tag{2.12}$$

where the first entry indicates the qubit amplitude and the second the pointer amplitude. From the equations of ψ^+ and ψ^- (and applying the relevant trig identities) it is clear that, in the dispersive limit, ψ^+ is more "qubit-like" and ψ^- is more "pointer-like". From each of the uncoupled states, the (now hybridized) qubit and pointer can inherit loss. This loss can be determined in the dispersive limit via a weigted average of the loss mechanisms γ_q for the qubit internal loss and κ_c for the oscillator internalt loss. For the qubit-like mode we have that

$$\gamma_{total} = P_{qubit}\gamma_q + P_{pointer}\kappa_c = \cos\Big(\tan^{-1}(\frac{g}{\delta})\Big)^2\gamma_q + \sin\Big(\tan^{-1}(\frac{g}{\delta})\Big)^2\kappa_c \tag{2.13}$$

and so loss is largely dominated by the qubit internal loss γ_q (often relatively small) despite the presence of the pointer. It should be noted that this is not the only formula that represents qubit decay in the coupled qubit-resonator system. More exact methods exist which rely on Fermi's Golden Rule or, even more exactly, the Quantum Master Equation (for states) and Quantum Langevin Equation (for operators).

In order to determine if the qubit-like mode is in the ground or first excited state, one can look at the pointer levels of the coupled system, the higher two energy levels of the coupled two-level system matrix. If the energy transition that is visible via spectroscopy is at $\tilde{\Omega_2}$ then the qubit is in the ground state. If the transition is at $\tilde{\Omega_2} - \frac{g^2}{\delta}$ then the qubit is in the excited state. This factor $\frac{g^2}{\delta}$ is known as the dispersive shift χ of the resonator mode. This shift is also often called the "cross-kerr" between the two oscillators.

2.3.2 Measurement

With the intuition for readout established, we are now able to dive into the specifics of measurement. As mentioned before, measurement in superconducting systems relies on a "pointer" system which is coupled to the system of interest. Knowledge of the qubit is therefore gained indirectly and the level of knowledge gained can be explicitly represented using Bayes' rule

$$P(y|x) = \frac{P(x,y)}{P(x)} \tag{2.14}$$

If the conditional probability, $P(y|x)$ is large, x the pointer system and y the qubit, then a strong measurement is said to occur, otherwise the measurement is weak.
To apply this concept to superconducting circuits, we consider the common case of a two-level qubit coupled to a resonator (pointer system) with a measurement performed via phase, i.e., we look at the phase difference between an incoming and outgoing coherent tone in order to determine the pointer frequency (and hence the qubit state). To write this explicitly, we consider our initial state as an disentangled product state between the qubit and the pointer given by

$$(a\,|0\rangle + b\,|1\rangle)\,|\alpha\rangle \tag{2.15}$$

where $|\alpha\rangle$ is a coherent state, a classical state of light initially incident on the pointer from a microwave source. Here we have assumed that such a state already exists in a waveguide incident on the pointer. Such a choice of state for the pointer may not be obvious, but it is what is commonly used in practice for superconducting systems as it is simple to generate. The reflection coefficient for a QHO pointer coupled to a qubit is given by the state-dependent relation

$$r[\omega] = \frac{\chi\sigma_z + i\kappa/2}{\chi\sigma_z - i\kappa/2} = e^{i\theta_0\sigma_z} \tag{2.16}$$

where χ is the previously defined dispersive shift from Chapter 2.3.1 and θ_0 is the resulting phase accrual that is dependent on this shift $\theta_0 = 2\tan^{-1}(\frac{\kappa}{2\chi})$. For a derivation

of this relation and a brief overview of quantum input-output theory see the appendix of Ref. [31]. The resulting output state reflected off of the pointer is therefore

$$a[e^{-i\theta_0\sigma_z}\,|0\alpha\rangle] + b[e^{i\theta_0\sigma_z}]\,|1\alpha\rangle \tag{2.17}$$

When the qubit is in the $|0\rangle$ state, the reflected wave has a phase that is shifted by $-\theta_0$ as compared to the incident wave and when the qubit is in the $|1\rangle$ state the reflected wave has a phase that is shifted by θ_0 as compared to the incident wave. After this reflection, the phase and amplitude of the reflected wave has to be measured.

Unfortunately, there is natural uncertainty in this measurement owing to the Heisenberg uncertainty principle. This uncertainty leads to a measurement uncertainty that is present even if the qubit is fully in one state or the other ($a = 1$ or $b = 1$). For coherent states, as are typically used, phase and amplitude have equal variances. A typical set of measurement outcomes (over several measurements) is given in Fig. 2-5 and plotted on the complex plane as phasors where the haze of the plot represents the certainty of the measurement result. Given that phase and amplitude have equal uncertainty, the hazy results form circles in the plane. The level of overlap of these circles is controllable via the amplitude of the output, the phase angle θ_0 and the radius of the circles. If additional noise is present beyond the natural Heisenberg uncertainty, then the radius of the circles will be larger. Typically, signal amplitude is limited by several sources including the excitation of spurious transitions in the system and fridge-heating due to absorbed signal power. Therefore the phase θ_0 is often the parameter of interest for optimization. As is clear in the plot, the optimal angle θ_0 minimizes the overlap between the circles and is therefore $\theta_0 = \pi/2$. This gives the common design criterion then of $2\chi = \kappa$ for optimal θ_0.

2.3.3 Control of Two-Level Quantum Systems

Control of quantum systems requires many of the same tools that we used for readout. To control a $|0\rangle$-$|1\rangle$ system, similar to 2.3.1, a coherent tone is applied, although this time it is directly to the qubit. Assuming that the system stays within

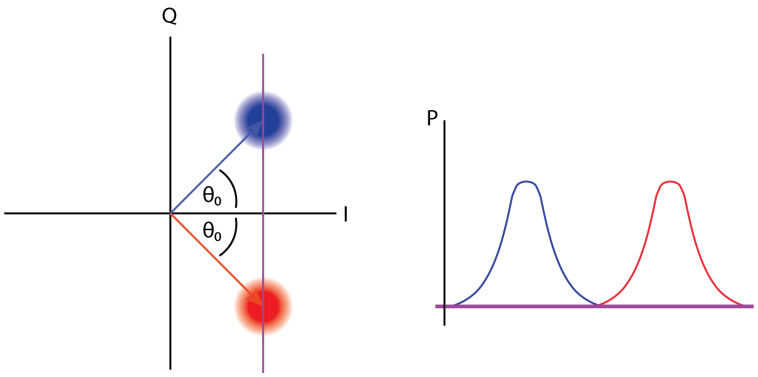


Figure 2-5: (Left) A visualization of the reflected signal in 2D. The relative phase is represented as the angle from the x (in-plane) axis. (Right) a single slice of the I-Q plane. During readout, a signal with phase and magnitude closer to the left gaussian would indicate that the qubit is likely in state $|0\rangle$.

the $|0\rangle$-$|1\rangle$ subspace, a photon is absorbed from this driving field by the qubit and re-emitted if the drive proceeds for too long. Fortunately, we have already seen a system similar to this in the coupled oscillator model, although it was a static system. To borrow from this intuition, we use a semi-classical approach and represent an (AC) driving field as a classical signal applied to our quantum system with a form $V_I(t)\cos(\omega_d t) + V_R(t)\sin(\omega_d t)$. Where we have assumed that the AC driving field is strongly modulated at a single frequency ω_d with relatively slow varying amplitude modulation represented by $V_I(t)$ and $V_R(t)$. Given that typical superconducting qubit frequencies are on the order of GHz, this puts the modulation on the order of nanoseconds, fundamentally limiting the speed of operations. In our Hamiltonian model of this process, we can borrow from our previous coupled oscillator Hamiltonian, but replace the second oscillator with the classical driving field.

$$H_{drive} = \hbar\Omega \hat{a_1}^\dagger \hat{a_1} + \frac{-i}{2} C_{coupling} \sqrt{\frac{\hbar\Omega_1}{2C_1}} (\hat{a_1} + \hat{a_1}^\dagger) \Big(V_R(t)\cos(\omega_d t) + V_I(t)\sin(\omega_d t) \Big) \tag{2.18}$$

At this point, it is often better to represent the qubit in the $|0\rangle$-$|1\rangle$ subspace, as opposed the using the raising and lowering operators $\hat{a}$ and $\hat{a}^\dagger$. To do so, we use the pauli operators $\frac{\sigma_z + I}{2}$ to represent $\hat{a}^\dagger\hat{a}$ and σ_x to represent $\hat{a} + \hat{a}^\dagger$. Rewriting in this

way gives (ignoring the constant term):

$$H_{drive} = \hbar\frac{\Omega}{2}\sigma_z + \frac{-i}{2}C_{coupling}\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sigma_x\Big(V_R(t)\cos(\omega_d t) + V_I(t)\sin(\omega_d t)\Big)$$

It is typical, and helpful, to transform the Hamiltonian into a frame "rotating" at the drive frequency ω_d. This is a valid operation as it is merely a unitary transformation, so it keeps all eigenvalues fixed, but removes the fast-oscillating carrier terms in the Hamiltonian that are otherwise confusing to follow in time. To perform this unitary transformation we use $U = e^{i\frac{\omega_d}{2}t\sigma_z}$ when the system is driven on-resonance such that

$$\begin{aligned}
H_{rot} &= e^{-i\frac{\omega_d}{2}t\sigma_z}He^{i\frac{\omega_d}{2}t\sigma_z} + i\dot{U}U^\dagger \\
&= \hbar\frac{\Omega}{2}\sigma_z + e^{-i\frac{\omega_d}{2}t\sigma_z}\left(\frac{-i}{2}C_{coupling}\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sigma_x\Big(V_R(t)\cos(\omega_d t) + V_I(t)\sin(\omega_d t)\Big)\right)e^{i\frac{\omega_d}{2}t\sigma_z} - \hbar\frac{\omega_d}{2}\sigma_z \\
&= e^{-i\frac{\omega_d}{2}t\sigma_z}\left(\frac{-i}{2}C_{coupling}\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sigma_x\Big(V_R(t)\cos(\omega_d t) + V_I(t)\sin(\omega_d t)\Big)\right)e^{i\frac{\omega_d}{2}t\sigma_z} \\
&= \Big(\cos(\omega_d t) - i\sin(\omega_d t)\sigma_z\Big)\times \\
&\left(\frac{-i}{2}C_{coupling}\sqrt{\frac{\hbar\Omega_1}{2C_1}}\sigma_x\Big(V_R(t)\cos(\omega_d t) + V_I(t)\sin(\omega_d t)\Big)\right)\times \\
&\Big(\cos(\omega_d t) + i\sin(\omega_d t)\sigma_z\Big) \\
&= \frac{-i}{2}C_{coupling}\sqrt{\frac{\hbar\Omega_1}{2C_1}}\Big(V_R(t)\sigma_y + V_I(t)\sigma_x\Big) + O(2\omega)
\end{aligned} \tag{2.19}$$

where we have used Euler's identity to convert exponentiated operators to sines and cosines. With only a σ_x and σ_y term present, such a Hamiltonian applied to a state as $e^{-iHt}|0\rangle$, precesses the state under its natural frequency in the non-rotating frame and drives a $|0\rangle$ to $|1\rangle$ transition. It should be noted that these equations (and hence this operation) only hold when $\Omega = \omega_d$. If this is not the case, then the static term in the Hamiltonian $\frac{\Omega}{2}\sigma_z$ will not cancel with the $-i\dot{U}U^\dagger = -\frac{\omega_d}{2}\sigma_z$ term. This off-resonance will cause an additional phase to accrue during the drive and will keep the drive from fully exciting the qubit.

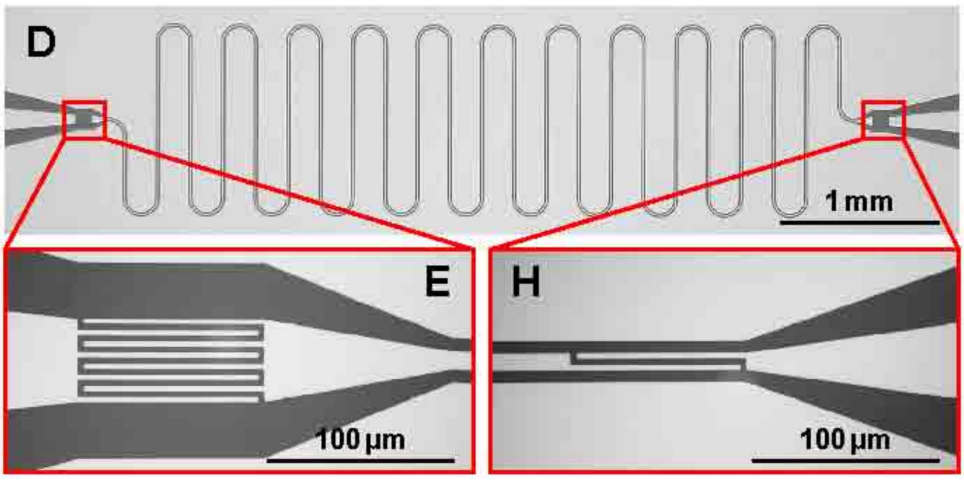


Figure 2-6: An image of a 2D coplanar waveguid resonator on-chip. Highlighted are the interdigitated capacitors coupling the waveguide meander to the rest of the transmission line.

2.4 Superconducting Quantum Circuits

Superconducting circuits are a particular implementation of the methods outlined in the previous sections. These circuits utilize two or three dimensional microwave and high-frequency radio-wave (tens of MHz to tens of GHz) elements to generate components like the QHO example given in the previous sections. While there are several properties of superconductors that can make them useful for this application, the main reason for their use is their extremely low loss within the superconductor, allowing for coherent operations to be performed before the information decays out of the system.

In practice, to create a QHO structure as described in the prior sections, superconducting circuit designers often use two-dimensional resonator structures made of coplanar-waveguides Fig. 2-6. This type resonator is often made in either a quarter-wave or half-wave configuration with the resonant frequency (which corresponds to the energy contained in a single excitation of the resonator Ω), set by both the length of the resonator and (weakly) by the strength of the resonator's coupling to its environment.

Thus far, we have assumed that our quantum systems have only two levels (at least

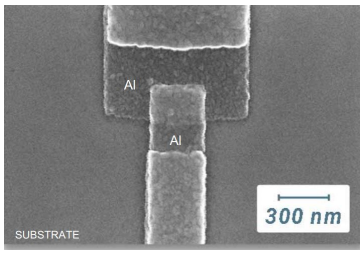


Figure 2-7: An image of a Josephson junction on-chip in aluminum. Shown are a top conductor and bottom conductor with a thin insulating layer in between (not visible), patterned on top of a substrate.

in all conversations of qubit dynamics). This isn't always a very good approximation, however. In fact, this approximation does not hold at all for harmonic oscillators. This is because QHO's have evenly spaced transitions in energy (as show in Fig. 2-2). Therefore, a microwave drive at a particular frequency used to excite the system will simultaneously drive multiple transitions - taking the qubit far out of the $|0\rangle$-$|1\rangle$ subspace.

The way to prevent this from happening is for our levels to be unevenly spaced. In other words, we need the energy of each level not to be equivalent to some constant times the number of excitations, as is the case for a QHO (Chapter 2-2). Instead we need the energy to be some *nonlinear* function of the number of excitations. The nonlinear element used in superconducting circuits for this purpose is the Josephson junction. Physically, this element, shown in Fig. 2-7, is simply a small insulating layer sandwiched between two superconducting plates, much like a capacitor. The difference is that charges ("Cooper" pairs of charges, to be precise) can tunnel through this barrier, similar to a leaky MOS capacitor. In addition, and more importantly, a phase-matching condition holds across this barrier (as Cooper-pairs follow a wave-particle duality condition). This phase matching condition is similar (and in fact nearly the same) as the phase matching condition across a classical inductor (90 degrees, given that $V \propto \dot{I}$ where I is a sinusoid). However, we will claim (without proof, for proof see [31]), that in a Josephson junction, the condition is now nonlinear.

We still have that $V = \dot{\Phi}$ as in the QHO case, but now we have that $I = I_c \sin(\phi)$

instead of $I = \frac{\Phi}{L}$ as in the the QHO case. The quantity ϕ is the so-called superconducting phase and is related to the flux via $\Phi = \frac{\Phi_0 \phi}{2\pi}$. I_c is a temperature, material and magnetic field dependent (but not ϕ-dependent) quantity known as the critical current. This sinusoidal flux can be seen as a nonlinear inductance which is only weakly nonlinear for small values of ϕ. To better treat this nonlinearity We can use Faraday's law:

$$\begin{aligned} V &= \dot{\Phi} \\ &= \frac{\Phi_0 \dot{\phi}}{2\pi} \\ &= L\frac{\partial I}{\partial t} \\ &= L I_c \cos(\phi)\dot{\phi} \\ L_J &= \frac{\Phi_0}{2\pi I_c \cos(\phi)} \end{aligned}$$

The above quantity is the "Josephson Inductance" which, to first order is simply $L_J \approxeq \frac{\Phi_0}{2\pi I_c}$.

2.4.1 The Transmon

We now have the tools necessary to define the workhorse of superconducting qubits (at least so far). This is the transmon, shown in Fig. 2-8. The transmon is simply a QHO, where the inductor has been replaced by a Josephson junction. For this reason the transmon is often referred to as a quantum *an*-harmonic oscillator.

The transmon Hamiltonian can be written in the charge and flux basis in the same way as the QHO (except with the sinusoidal flux dependence) as:

$$H_t = -\frac{\Phi_0 I_c}{2\pi}\cos(\phi) + \frac{\hat{q}^2}{2C} \tag{2.20}$$

Where the factor $\frac{\Phi_0}{2\pi}$ and the $\cos(\phi)$ in the first term comes from the fact that we have integrated the current over time in order to get energy. We will note that the factor $\frac{\Phi_0 I_c}{2\pi}$ is commonly known as the Josephson energy (E_J) and the parameter (including

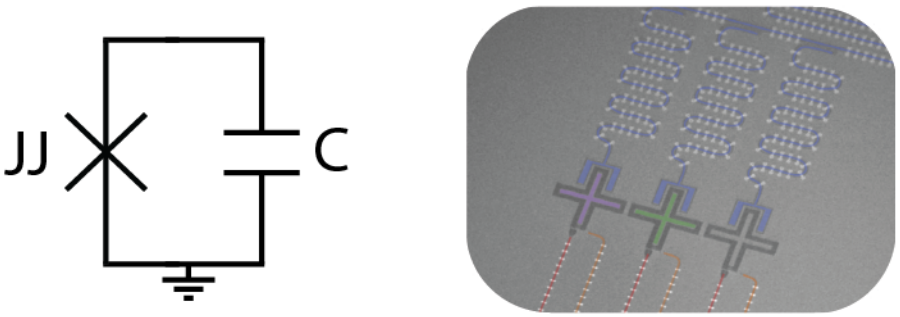


Figure 2-8: (Left) The circuit diagram of a transmon with an X (representing the Josephson junction) on the left arm and a shunt capacitor on the right. (Right) A false-color image of a set of three transmons on-chip in the "x-mon" formation, where the shunt-capacitor forms an X shape. Charge lines and flux lines are in orange and red respectively while readout resonators are in blue.

the vacuum-fluctuation prefactors) $\frac{(-i\sqrt{\frac{C\hbar\Omega}{2}})^2}{2C}$ is the charging energy $E_C = \frac{e^2}{2C}$. Taking the previously defined raising and lowering operators and expanding the cosine, we can write this Hamiltonian in terms of ladder operators as:

$$\begin{aligned} H_t &= \frac{\hat{q}^2}{2C} - \frac{\Phi_0 I_c}{2\pi}\Big(-\frac{\phi^2}{2!} + \frac{\phi^4}{4!} - ...\Big) \\ &= \frac{\Big(-i\sqrt{\frac{C\hbar\Omega}{2}}(\hat{a} - \hat{a}^\dagger)\Big)^2}{2C} - \frac{\Phi_0 I_c}{2\pi}\left(-\frac{\Big(\frac{2\pi}{\Phi_0}\sqrt{\frac{L\hbar\Omega}{2\Phi_0}}(\hat{a} + \hat{a}^\dagger)\Big)^2}{2!} + \frac{\Big(\frac{2\pi}{\Phi_0}\sqrt{\frac{L\hbar\Omega}{2\Phi_0}}(\hat{a} + \hat{a}^\dagger)\Big)^4}{4!} - ...\right) \\ &= \hbar\Omega(\hat{a}^\dagger\hat{a}) - \frac{\alpha}{2}\hat{a}^{\dagger 2}\hat{a}^2 - \alpha\hat{a}^\dagger\hat{a} + ... \end{aligned} \tag{2.21}$$

Where we have taken $\Omega = \frac{1}{\sqrt{LC}} = \sqrt{8E_J E_C}$ and only explicitly shown terms up to first order in perturbation theory. We have also "normal ordered" the ladder operators (moved all conjugates to the left), using the identity $\hat{a}\hat{a}^\dagger = 1 + \hat{a}^\dagger\hat{a}$. From the final line of the previous algebra, we see that the transmon Hamiltonian has the same term as the QHO Hamiltonian, a $\Omega(\hat{a}^\dagger\hat{a})$ term, however, this term is modified by the higher orders of the Taylor expansion. Namely, to first-order, the frequency of the transmon at a single-excitation level is actually down-shifted from Ω by the

"anharmonicity" $\alpha = E_c$. The more important part of this Hamiltonian, however, is the energy relationship between the energy of the first excitation and that of the second excitation.

If we apply the Fock state $|1\rangle$ to this Hamiltonian, the term $\langle 1| \frac{\alpha}{2}\hat{a}^{\dagger 2}\hat{a}^2 |1\rangle = 0$ since $\hat{a}^2 |1\rangle = \hat{a}\hat{a} |1\rangle = \hat{a} |0\rangle = 0$. However, if we apply the second Fock state $|2\rangle$ and determine its energy, we see that the term $\langle 2| \frac{\alpha}{2}\hat{a}^{\dagger 2}\hat{a}^2 |2\rangle$ contributes an additional $-\alpha$ to this energy. This is the reason that α is called the anharmonicity. It is the difference between the energy that one would expect for the second excited state of a QHO (twice the first-excited state frequency $2\Omega - 2\alpha$) and the actual (to first order) energy. Sometimes this anharmonicity is also known as the "self-kerr" of the transmon, similar to the dispersive energy shift which we noted was called "cross-kerr". We can again write this Hamiltonian in matrix form, although its exactness depends on the order of the Taylor expansion as well as the number of levels included in the representation (size of the matrix).

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & \Omega - \alpha & 0 \\ 0 & 0 & 2\Omega - 3\alpha \end{bmatrix}$$

Now that we have this anhamonic Hamiltonian, though, and given that anharmonicity is often on the order of hundreds of MHz, it is fair to make the two-level truncation to get the matrix:

$$\begin{bmatrix} 0 & 0 \\ 0 & \Omega - \alpha \end{bmatrix}$$

In many of the upcoming sections, this is the matrix used to represent the functional elements in the superconducting circuit design with transmon qubits.

Chapter 3

Fundamentals of Quantum and Quantum-inspired Optimization

3.1 Concepts in Black-Box Optimization

Within black-box global combinatorial optimization, a few distinct research areas have emerged. The original successful algorithms for combinatorial optimization typically relied on basic heuristics and were inspired by natural phenomena. Examples of these are genetic algorithms, first developed in Ref. [34] and based on evolutionary processes; simulated annealing, based on the process of solid purification and refinement during fabrication [35]; and Ant Colony Optimization [36]. Owing to their success, all of these algorithms are among the most often used algorithms in the field.

Beginning in the late 1990s, a new trend emerged in combinatorial optimization: the use of models [37]. Models are parameterized heuristics which learn optimal methods of solving via experience during the optimization process. In particular, genetic algorithms were frequently combined with models in either the offspring generation state, fitness evaluation stage, or candidate selection stages [38]. The fundamental idea is that, while genetic algorithms can globally converge using mutation and crossover operations, models may help in reaching that global minimum once offspring are in the vicinity. These estimation of distribution algorithms, as they are

known, are still an active area of research to this day, although they have waned in popularity for single-objective global optimization and instead are typically employed for multi-objective optimization [39]. In this context they often fall under the name of "model-based evolutionary algorithms". Concurrent to this movement in evolutionary algorithms, a related but more general approach to model-based optimization was developed known as the cross-entropy method [40]. In this method, a parameterized but simple surrogate model is trained to represent the problem of interest by minimizing the cross entropy or Kullbach-Leibler divergence between a subset of the known sampled solutions and the output distribution of the model. The model is then sampled in order to provide new points to test. The cross entropy method, as given in this form, fell largely out of explicit favor in the realm of combinatorial optimization by the end of the 2000s, but it still remains a very popular and successful method for training reinforcement learning algorithms.

While the previous examples of combinatorial algorithms involved the use of models in existing heuristic algorithms a largely separate branch of literature also developed in the early 2000s. This branch involved the explicit use of models for representing correlations in data and sprang from the continuous space - notably Bayesian Optimization [28]. In these optimization schemes, a correlation function, often based on a distance metric such as the Hamming distance. In the original formulation, correlations are assumed to exist strongly between datapoints that are close to each other and weaken the further the datapoints are from each other. These correlations (specifically the rate of correlational decay) are trained, via regression on samples from a cost function. Using the stochasticity represented by the trained correlation matrix, an expectation value over possible cost functions is then performed in order to find the expected improvement, or the likelihood of alternative options outperforming known minima. This expected improvement is used to suggest new samples. Note that this process is distinct from the surrogate model/cross-entropy approach, although it is trained in the same manner. This is because the cross-entropy approach uses a surrogate model that best fits the data to suggest new minima, whereas the Bayesian method takes an expectation over all possible surrogates.

It is within this sea of classical model-based algorithms and somewhere between surrogate models and the Bayesian Optimization that quantum and quantum-inspired model-based combinatorial optimization algorithms were developed. In this Chapter, we will largely focus on a particular quantum-inspired model: a Matrix Product State (MPS) Tensor Network. The fundamentals of MPs (and Tensor Networks in general) are discussed in section 3.2. A particular method of using MPS models for combinatorial optimization is then discussed in section 3.3. Finally, A method of utilizing fully quantum models for optimization is discussed in section 3.4

3.2 Tensor Networks

3.2.1 Quantum representations

Quantum systems can be represented in a number of equally-correct ways. Common examples of ways in which systems are represented are as matrices and statevectors or as operators and bras and kets as used in Chapter 2 and given below:

Matrices

$$\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Bra-ket

$$\hat{a}^\dagger \qquad |0\rangle$$

Tensor networks are another such representation and have the useful property that they allow for controllable approximations of quantum states. This allows them to represent states of exponentially scaling size with subexponential resources, up to an approximation error that depends on the level of entanglement in the system. The drawback of Tensor Networks, as opposed to these other methods of representation, is that they require some heavy notational background to be intuitive. Over the next few sections, I will provide an overview of the notation of tensor networks. In particular, I will attempt to show how tensor networks provide an answer to the question

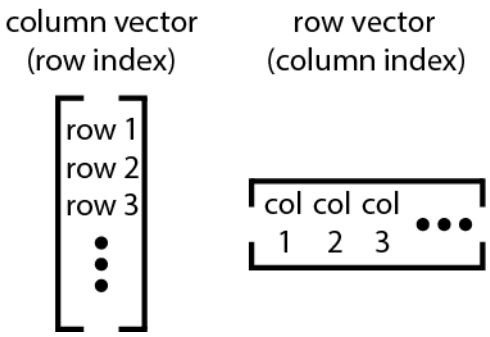


Figure 3-1: (Left) A column vector where an index enumerates the rows. (Right) A row vector where an index enumerates the columns.

of how to best approximate an exponential quantum state. I will then then use the developed notation to show why matrix product states (a particular type of tensor network) are useful for generative modeling and optimization.

Mathematically, a tensor is an object with one or more indices. The common "order-one" tensor, or a tensor with only one index (or dimension), is a vector v_j where j is the index. The index of a tensor can either be a column-space index or a row-space index - i.e., the index counts columns of the tensor (therefore making the vector a row-vector) or it counts rows of a tensor (therefore making it a column-vector). I will reiterate this previous point as it is one of common confusion: row vectors have column-space indices and column-vectors have row-space indices, as shown in Fig. 3-1. For example, the common "order-two" tensor is a matrix $M_{i,j}$ and is in general an outer product of a row vector with a column vector. Therefore, it has two indices, one row-space index (i) and one column-space index (j). Tensors are in general non-square and are more precisely described as being of "order-(p,q)", where "p" is the number of row-space indices of the tensor and "q" is the number of column-space indices of the tensor.

This distinction is important when we consider tensor contraction. Contraction for tensors of arbitrary order works in the same way as for matrices and vectors. For example, given a matrix M (order-(1,1)) and column-vector v (order-(1,0)) we have

that:

$$Mv = \sum_j M_{i,j} v_j \tag{3.1}$$

where the i in $M_{i,j}$ is a row-space index and the j in $M_{i,j}$ is a column-space index, while the j in v_j is a row-space index (as it is a column-vector). After contraction then, one would expect that only a row-space index (i) would remain (a column vector) as the common index between row and column spaces is summed over. One issue with the common notation is that it does not differentiate between row-space and column space indices, which makes it difficult to tell the shape of the tensor. From this point onward, following [1], row-space indices will be placed in the superscript, and column-space indices will be placed in the subscript. Rewriting equation 3.1 we have:

$$Mv = \sum_j M^i_j v^j \tag{3.2}$$

Extending this to arbitrary order tensors, we can write a general tensor (T) with p rows and q columns as

$$T = T^{i(1),i(2),..,i(p)}_{i(p+1)...i(p+q)}$$

The (1) through (p) parentheticals next to each index are merely used as a way to differentiate one row/column-space index from another without resorting to using more letters (k, m, n, etc).

3.2.2 Tensor Network Diagrams

The tensor network representation relies heavily on diagrams, which help to reduce the complexity of the mathematical treatment of tensor networks provided in the previous section. The simplest tensor representation in the diagrammatic notation is a line (see Fig. 3-2). A line indicates that no operation is done. The next most complicated tensor is a single-index column vector, indicated by a right-facing triangle. In

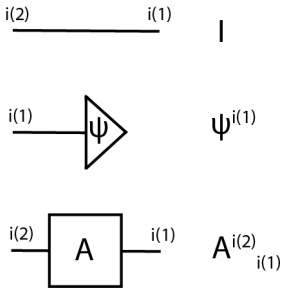


Figure 3-2: Three basic tensor network diagrams. The first is the identity operator. The second is a ket with rowspace index i, and the third is a general matrix operator A with row space index i(2) and column space index i(1).

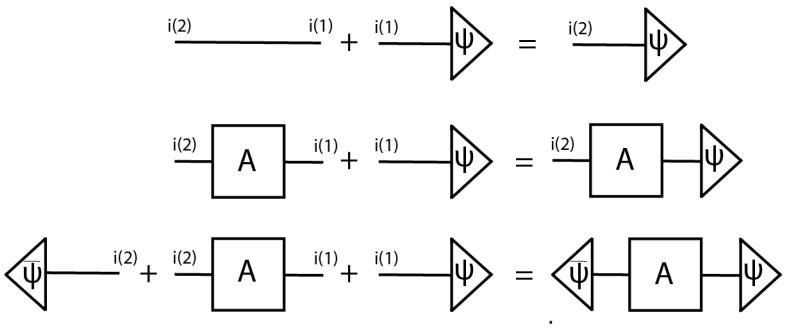


Figure 3-3: Concatenating shapes represents tensor multiplication. Using two triangles and a square, one can write an expectation value.

Fig. 3-2, this triangle, representing wavefunction ψ, has one index $i(1)$ representing the entries in a row-space vector. Column-space vectors, as opposed to row-space vectors, are similarly given by a left-facing triangle instead of a right-facing triangle.

The next most complicated tensor is a rank-(1,1) matrix represented by the box (A) in Fig. 3-2. An expectation value on observable A can then be diagrammatically performed by combining the previous elements, a left-facing triangle, a matrix A, and a right-facing triangle as shown in Fig. 3-3 and given as: $\sum_{i(1),i(2)} \psi_{i(2)} A^{i(2)}_{i(1)} \psi^{i(1)} = \langle\psi| A |\psi\rangle$.

The previous examples of tensor network diagrams have only included matrices and column/row vectors, they have not included higher-rank tensors. Examples of higher-rank tensors are given in Fig. 3-4. The way to consider higher-order tensors, like the far left tensor in Fig. 3-4, is that the rows of A are now indexed by (i(1),i(2)),

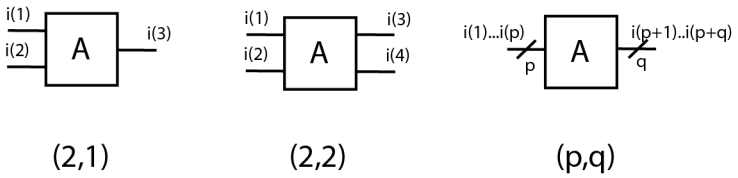


Figure 3-4: Three examples of higher-rank tensors, the first being a tensor of rank (2,1), the second a tensor of rank (2,2), and the third a tensor of rank (p,q)

a tensor product of the spaces indexed by i(1) and i(2).

Thus far, little has been shown that cannot easily be shown using bra-ket notation. There exist a larger set of operations (some of which are shown in Fig. 3-5) that make this notation so powerful. In Fig. 3-5, the first operation is a swap operation. This is the same representation as used in quantum circuit diagrams and should be familiar. The second operation shown is the adjoint operation, it takes a row index and converts it to a column index as shown in the example in Fig. 3-5. This same operation can be used to vectorize a matrix A, or to matricize a vector A. As an example of matricization, consider the entangled GHZ state, where we have matricized by flipping qubit 1 kets to bras and renormalized to make the operator unitary:

$$\begin{aligned}\begin{bmatrix}\frac{1}{\sqrt{2}} & 0 & 0 & \frac{1}{\sqrt{2}}\end{bmatrix} &= \frac{1}{\sqrt{2}}|00\rangle + 0|01\rangle + 0|10\rangle + \frac{1}{\sqrt{2}}|11\rangle \\ &= \frac{1}{\sqrt{2}}|0\rangle\langle 0| + 0|0\rangle\langle 1| + 0|1\rangle\langle 0| + \frac{1}{\sqrt{2}}|1\rangle\langle 1| \\ &= \begin{bmatrix}1 & 0 \\ 0 & 1\end{bmatrix}\end{aligned}$$

In addition, we can illustrate an example of vectorization. Consider the Hadamard

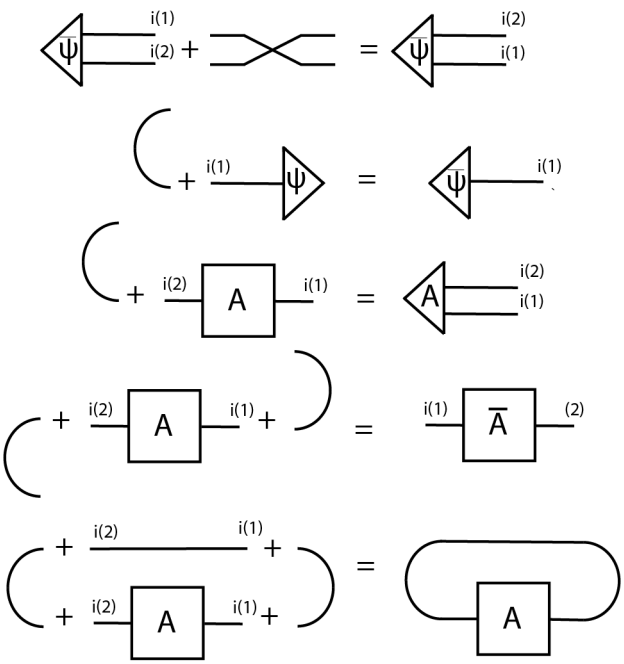


Figure 3-5: In descending order: the swap operator applied to a row vector, the adjoint applied to a column vector, the adjoint applied to a matrix, the adjoint applied to one of the indices of a vector- where the end of the adjoint points to the index that was flipped, the adjoint applied to both sides of a matrix, the adjoint combined with an identity to create a trace.

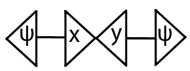
gate where we vectorize by flipping qubit 1 bras to kets:

$$\begin{aligned} H &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ &= \frac{1}{\sqrt{2}}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|) \\ &= \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle - |11\rangle) \\ &= \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & -1 \end{bmatrix} \end{aligned}$$

It is not these operators, swap and adjoint, in particular, that are so useful, but rather the general idea that they represent: any configuration of lines between vectors and operators is a valid operation to perform and has real meaning.

Differentiation with respect to a elements of a tensor network diagram can also be easily expressed using tensor network diagrams. To demonstrate this, I will first provide a mathematical example of differentiation of an inner product ($\langle\psi| M |\psi\rangle$) with respect to a matrix element (M^x_y), where x and y are arbitrary, but fixed, values taken on by the indices i(2) and i(1) respectively:

$$\begin{aligned} \frac{\partial}{\partial M^x_y}(\langle\psi| M |\psi\rangle) &= \sum_{i(1),i(2)} \frac{\partial(\psi_{i(2)} M^{i(2)}_{i(1)} \psi^{i(1)})}{\partial M^x_y} \\ &= \sum_{i(1),i(2)} \begin{cases} \psi_{i(2)} 1 \psi^{i(1)} & \text{if } i(2) = x \text{ and } i(1) = y \\ \psi_{i(2)} 0 \psi^{i(1)} & \text{if } i(2) \neq x \text{ or } i(1) \neq y \end{cases} \\ &= \psi_y \psi^x \\ &= \sum_{i(1),i(2)} \psi_{i(2)} x^{i(2)} y_{i(1)} \psi^{i(1)} \\ &= \end{aligned}$$



Where the triangles $|x\rangle$ and $\langle y|$ are computational statevectors with 1's at indices x

and y respectively and 0's elsewhere. So taking the derivative of an inner product of a tensor network with respect to an element of a tensor results in a multiplication of projections. This is true in general and can be shown for matrices below:

$$\begin{aligned}
\frac{\partial}{\partial M_{1,2}}\left(\begin{bmatrix} v_1^* & v_2^* \end{bmatrix} \begin{bmatrix} M_{1,1} & M_{1,2} \\ M_{2,1} & M_{2,2} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}\right) &= \frac{\partial}{\partial M_{1,2}}\left(\begin{bmatrix} v_1^* & v_2^* \end{bmatrix} \begin{bmatrix} M_{1,1}v_1 + M_{1,2}v_2 \\ M_{2,1}v_1 + M_{2,2}v_2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}\right) \\
&= \frac{\partial}{\partial M_{1,2}}\left(\begin{bmatrix} v_1^* & v_2^* \end{bmatrix} \begin{bmatrix} M_{1,1}v_1 + M_{1,2}v_2 \\ M_{2,1}v_1 + M_{2,2}v_2 \end{bmatrix}\right) \\
&= \frac{\partial}{\partial M_{1,2}}(v_1^* M_{1,1} v_1 + v_1^* M_{1,2} v_2 + v_2^* M_{2,1} v_1 + v_2^* M_{2,2} v_2) \\
&= v_1^* v_2
\end{aligned}$$

To summarize, in this subsection we have shown that tensor network diagrams can be used to represent operations using tensors, where statevectors are triangles and operators are boxes or lines (in the case of the identity). In addition, derivatives of tensor networks can be represented using projectors (two triangles).

3.2.3 Singular Value Decomposition and Matrix Product States

The singular value decomposition (SVD) is the key reason why tensor networks are useful for approximating quantum states. Typically, the SVD is a method for rewriting and approximating matrices, as shown in Fig. 3-6. The goal of this section is to apply the notation and techniques from the previous section in order to explain how the SVD can be used to also find good approximations for statevectors, not just matrices. For an operator $A \in \mathbb{C}^{m \times n}$, the SVD is a factorization of the form $A = U\Sigma V$ where $\Sigma \in \mathbb{C}^{m \times n}$ is diagonal and non-square in general, and U and V are square matrices and elements of $\mathbb{C}^{m \times m}$ and $\mathbb{C}^{n \times n}$ respectively. The diagonal elements of the matrix Σ are "singular values" unique to the operator. SVD is useful for approxima-

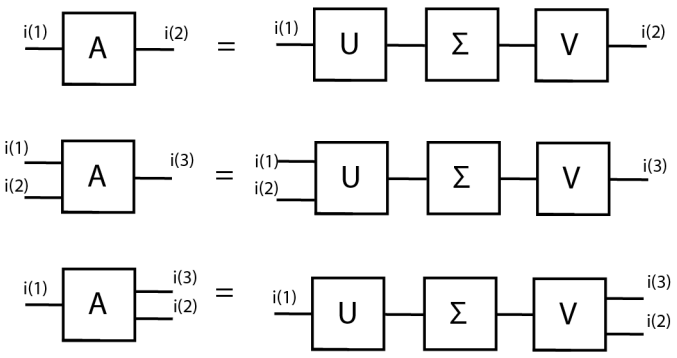


Figure 3-6: The "diagrammatic" SVD. This can be applied to higher-order tensors as well, provided they are first matricized. In the bottom panel it is made clear that the extra index (i(3)), can be associated with either U or V depending on how the operator A is matricized before the SVD is performed.

tions because one can change the number of singular values kept in Σ via truncation - i.e. change the dimension of Σ from $\Sigma \in \mathbb{C}^{m\times n}$ to $\Sigma \in \mathbb{C}^{m-1\times n-1}$, in addition, the operators U and V must be altered from being an element of $\mathbb{C}^{m\times m}$ to an element of $\mathbb{C}^{m-1\times m}$ and $\mathbb{C}^{n\times n}$ to an element of $\mathbb{C}^{n-1\times n}$. Provided the largest singular values are kept, the approximation of the actual operator A can be quite good, with an error equal to the square of the largest singular value truncated. At the same time, only $(m \times (m-k)) + \min(m-k, n-k) + ((n-k) \times n)$ complex numbers need to be managed as opposed to $m \times n$ complex numbers, where k is number of singular values that have been excluded from the approximation. The traditional SVD can still be used directly when applied to higher-order tensors, provided the operator A is written explicitly matrix form with tensored-indices. In other words, for a tensor with three indices of dimension 2, the tensor must first be written as an element of $\mathbb{C}^{2\times 4}$ or $\mathbb{C}^{4\times 2}$ before the SVD is performed. This is also shown diagramatically in the bottom panel of Fig. 3-6.

We now move on to the key point of understanding in tensor networks - the application of SVD (iteratively) to a statevector (not matrix) to create tensor networks. There are several different ways to go about doing this iterative SVD, which leads to several different types of tensor networks and even different forms within each type of network. The general types of tensor networks include matrix product states (MPS), projected entangled pair states (PEPs), tree tensor networks (TTNs), and

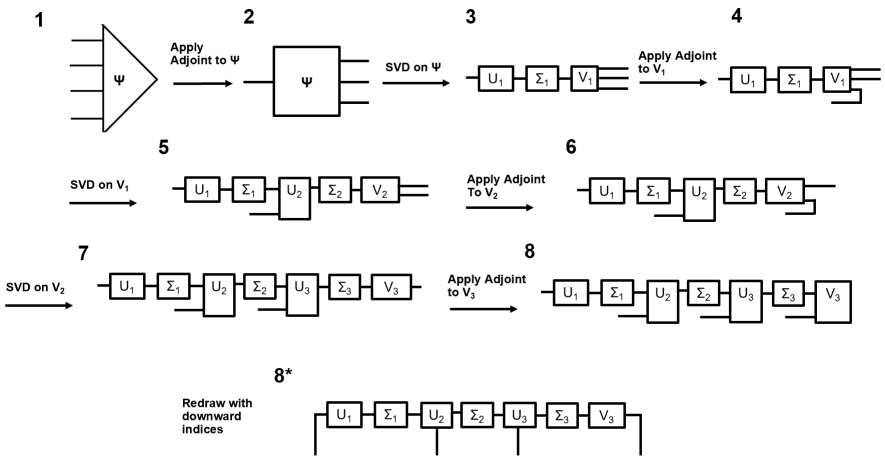


Figure 3-7: The iterative creation of a right-canonical MPS. In step 1, the adjoint is applied to one of the tensored indices of state Ψ, converting it to a matrix. The SVD is then applied to this matrix in step 2. This process is repeated until all indices are separated by singular value matrices Σ. The final state, given in step 8, is a matrix product state. The step 8* does not involve any additional operations, but is included here since typical drawing of MPS states often have downward facing indices as opposed to left-facing indices (for notational convenience).

more. We will specifically look at matrix product states which can take on a few different forms. In the following pages, I demonstrate the generation of an MPS in the "right-canonical form" starting from an arbitrary four qubit state ψ. The iterative SVD process is represented in Fig. 3-7. I highlight that this is just one of many ways to generate and define matrix products states. For additional details into these and other methods, please see Ref [41]. In the final panel of Fig. 3-7 operators U_1 and V_3 are unitary matrices, operators U_2 and U_3 are isometries (non-square as they are tensors but distance-preserving) and matrices Σ_1, Σ_2, and Σ_3 are diagonal and non-square matrices of the singular values. If this is a four-qubit state, as described, then each index (each of the downward-facing lines) can take on two values (0 or 1), that means that each of matrices Σ_1 and Σ_3 contain up to two singular values and that the matrix Σ_2 contains up to 4 singular values, the minimum dimension of their respective SVD steps. As we have just shown, in theory, this MPS state can represent any arbitrary state provided the number of singular values captured by matrices Σ_1, Σ_2, and Σ_3 is sufficient. In practice, the benefit of the MPS representation lies in

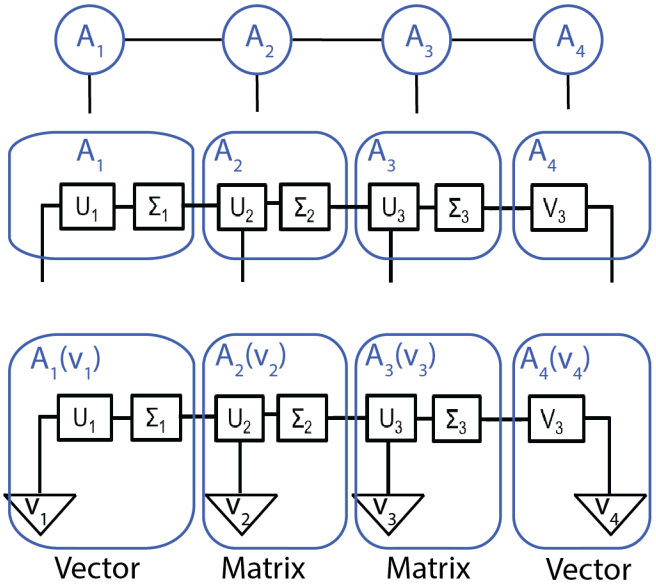


Figure 3-8: (Top) The common figure for MPS as given in the literature. (Middle) The equivalent representation for four qubits as given in the example MPS decomposition. (Bottom) Representations of the functions $A(v)$ in the form given by the example, where the outermost tensors are vectors and the innermost tensors are matrices.

using fewer singular values than necessary to represent arbitrary states. This means that the state representation likely won't be perfect (although it often still can be if these singular values are near zero), but most of the information contained in the state is still preserved. The maximum number of singular values contained within any of the SVD decompositions (Σ_1, Σ_2, and Σ_3 in the example) in a matrix product state is the "bond dimension". A four-qubit MPS state representing any arbitrary 4-qubit state (with no truncation) would therefore require a bond dimension of 4.

Often, matrix product states are represented in a slightly different (and less informative) way than the method given in panel 8 of Fig. 3-7. Instead, the representation at the top of Fig. 3-8 is often used. The downside of this more common representation is that it does not show explicitly the singular value matrices. This more common

representation follows a more common definition of MPS as:

$$\psi(v_1, ..v_n) = \mathrm{Tr}(A_1(\overrightarrow{v}_1)A_2(\overrightarrow{v}_2)A_3(\overrightarrow{v}_3)...) \tag{3.3}$$

I note that this subscripts in A_n and $\overrightarrow{v}_n$ are not index subscripts as used in the previous sections. From this point forward, we will do away with the previously used notation. Subscripts will now indicate a particular tensor A_1 through A_n and $\overrightarrow{v}_1$ through $\overrightarrow{v}_n$. Only superscripts will be treated as indices. Each A is represented by one of the circles in the top of Fig. 3-8, and there is a direct correspondence between the A operators/circles and the square operators given in the example in Fig. 3-7. $A_1(\overrightarrow{v}_1)$ is given by the multiplication of U_1 with a single-qubit statevector $\overrightarrow{v}_1$ and a diagonal matrix of singular values Σ_1 when compared with the example. $A_2(\overrightarrow{v}_2)$ is given by U_2 multiplied with a single qubit statevector $\overrightarrow{v}_2$ and by the diagonal matrix of singular values Σ_2, $A_3(\overrightarrow{v}_3)$ is given by the multiplication of U_3, the diagonal matrix Σ_2 and a single-qubit statevector v_3. A_4 is given by the multiplication of V_3 and a single-qubit statevector v_4. In practice, determining a state amplitude of an MPS state (using equation 3.3) requires the multiplication of two vectors with n-2 matrices, where n is the number of qubits. The number of parameters (distinct entries in the set of matrices and vectors used to evaluate the MPS) for a given set of $\overrightarrow{V}_k$ is then $\sum_k D_{k-1}D_k$ where k is the index of the matrix/vector from 1..n and D_k is the dimension of each matrix/vector, set by the bond-dimension (number of singular values kept in the factorization). Given that there are two computational states that $\overrightarrow{v}_k$ can represent ($|0\rangle$ and $|1\rangle$), the total number of parameters used in an MPS representation of arbitrary $\overrightarrow{v}_k$ is $\sum_k 2D_{k-1}D_k$.

The previous MPS was an example in the "right-canonical form", there also exist MPS's in "left-canonical form", where the SVD would start from the rightmost index in the example, and work back to the leftmost index. "Mixed-canonical form" representations also exist where the SVD can proceed from both left and right simultaneously. The reason for this distinction is because of an important property of

canonical form MPS, the product of the set of right (or left) A operators is equivalent to a unitary matrix for right (or left) canonical form. This property follows from the reversal of the expansion performed in Fig. 3-8 and the fact that the matrix V in any SVD is unitary. Using these properties of canonical form, MPs can be efficiently contracted (with fewer multiplications). This contraction is shown explicitly for the right-canonical case derived above in Fig. 3-9.

Conversion between right, left and Mixed-canonical form can be performed by applying local transformations to the individual A operators as shown in figure Fig. 3-10.

To summarize, we have shown how to iteratively apply the SVD to a general four-qubit matricized statevector Ψ in order to generate a unique representation of a quantum state. This representation can be made more dense, and less accurate, by changing the bond-dimension. The approximation error is related to the size of the singular values left out of the approximation as compared to those kept in the approximation. In addition, we have explained the useful property of left, right, and Mixed-canonical forms, which are that they can be contracted efficiently from the left or right.

3.3 Quantum-inspired Optimization

Quantum-inspired optimization is a broad term for any method of optimization performed on a classical computer which borrows intuition or form from quantum-mechanics. In common practice, the forms borrowed from quantum mechanics are representations of quantum states such as MPS and other tensor networks, which are then used to model and solve black-box combinatorial problems. This is related to the model-based optimization discussed in the introduction of this chapter. In the proceeding sections, I will first introduce quantum-inspired generative models and explain how tensor networks can be used to improve the performance of these models. I will then introduce the particular model-based quantum combinatorial optimization method (tensor network generator enhanced optimization, TN-GEO) which relies on

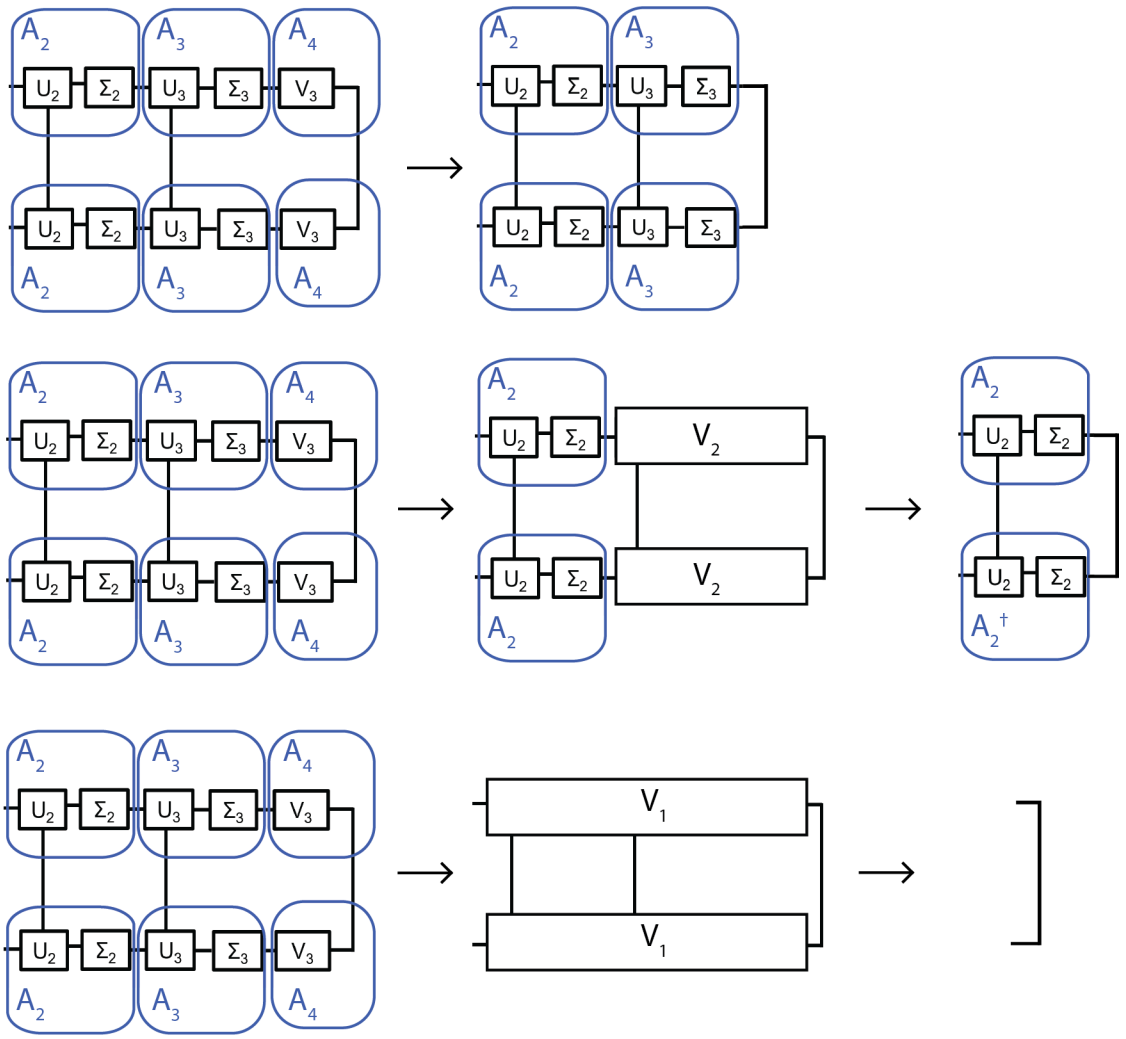


Figure 3-9: (top) The unitary property demonstrated for the furthest right operator for an MPS in right-canonical form. (middle) The unitary property demonstrated for the right two operators in the MPS. (bottom) the unitary property demonstrated for the right three operators in the MPS.

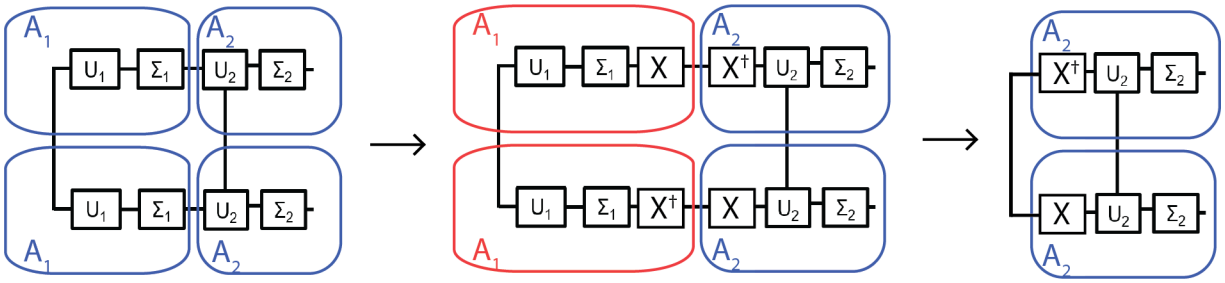


Figure 3-10: Converting a right-canonical MPS operator A_1 (blue) to a left-canonical MPS operator (red) via local application of an operator X which normalizes the rows of the product $U_1\Sigma_1$ such that $U_1\Sigma_1 X$ is now unitary and contracts from the left. Additional details on this procedure can be found in Ref. [2]

tensor network generative models and is used to generate the results in Chapter 4 of this work [26].

3.3.1 Quantum-inspired Generative Models

Quantum-inspired (tensor network) generative models provide the foundation of many quantum-inspired optimization methods. Here we specifically draw from Ref. [42], which was also used in [26]. Here, we utilize the probability distribution over computational (binary) states as a generative model distribution.

$$P = \frac{|\Psi(v)|^2}{Z}$$

where $\Psi(v)$ is the overlap between Ψ and some computational state v and Z is a partition function used to normalized the distribution (taken as $Z = \sum_v |\Psi(v)|^2$) since, generally, any SVD approximation of a state will break its normalization. For any binary bitstring v with entries v_k, an MPS representation will give a probability

$$P = \frac{|\Psi(v)|^2}{Z} = \frac{|\mathrm{Tr}(\Pi_k A_k(v_k))|^2}{Z}$$

This makes probabilities of computational states v efficient to evaluate provided the bond dimension of the MPS model is kept sufficiently low.

In generative models, the goal is not typically to find a model Ψ whose overlap with one particular state v is a desired value $P = |\Psi(v)|^2$, but instead to set up a model Ψ whose overlap with several (k in general) states v_1, ..., v_k match some set of desired values $P(v_k)$. We will denote the these several states as being part of a "training set" set T with $v_k \in T$. To create a model that outputs a desired distribution $P(v_k)$ is parameterize it and train it. In other words, we define as set of parameters θ which can be updated to minimize some distance metric between the distribution output by the model $|\Psi(v_k, \theta)|^2$ and the desired distribution $P(v_k)$. Several distance metrics exist for this purpose, but one of the most common is the Kullbach-Leibler (KL)

divergence,$D_{KL}(P||Q)$. The notation || indicates the divergence of P with respect to distribution Q as the KL divergence is not symmetric with respect to swapping these two distributions.

$$\begin{aligned}
\min_\theta D_{KL}(P||Q(\theta)) &= \min_\theta \sum_{\text{all } v} P(v) \log\Big(\frac{P(v)}{Q(v,\theta)}\Big) \\
&= \min_\theta \sum_{\text{all } v} P(v)\big(\log(P(v)) - \log(Q(v,\theta))\big) \\
&= \min_\theta \sum_{\text{all } v} -P(v)\log(Q(v,\theta)) \\
&\approxeq \min_\theta \frac{1}{|T|} \sum_{v_k \in T} -\log(|\Psi(v_k,\theta)|^2)
\end{aligned} \tag{3.4}$$

where v are all binary bitstrings (computation states), Q is the distribution output by the model, and P is the desired distribution. The final step is obtained through a finite sampling approximation, where it is assumed that the set of training data T is sampled from the desired distribution P, such that P is well-approximated by T, i.e. $P(v_k) \approx \frac{v_k \in T}{|T|}$ where the numerator is the number of copies of a particular v_k in the set T of all v_k. This is the reason for the disappearance of the explicit $P(v)$ in the summation in equation 3.4. It is instead approximated up to an error set by the number of sampled points (size of T). The factor of $\frac{1}{|T|}$ is used for normalization of this $P(v)$ approximation.

The utility of the MPS representation of the probability distribution $Q = |\Psi_{MPS}(v)|^2$ is that it provides an "analytical gradient" of $D_{KL}(\theta)$ that can be calculated in polynomial time, greatly speeding up training time. The formula for this analytical gradient is given as:

$$\frac{\partial D_{KL}}{\partial A_k^{i,i+1,w}} = \frac{Z'}{Z} - \frac{2}{|T|}\sum_v \frac{\Psi'(v)}{\Psi(v)} \tag{3.5}$$

where $A_k^{i,i+1,w}$ is a parameter (θ) of the wavefunction Ψ_{MPS} and is a particular element (i,i+1,w) of the rank (2,1) tensor A_k (See Fig. 3-8). Z' is the partial derivative of the

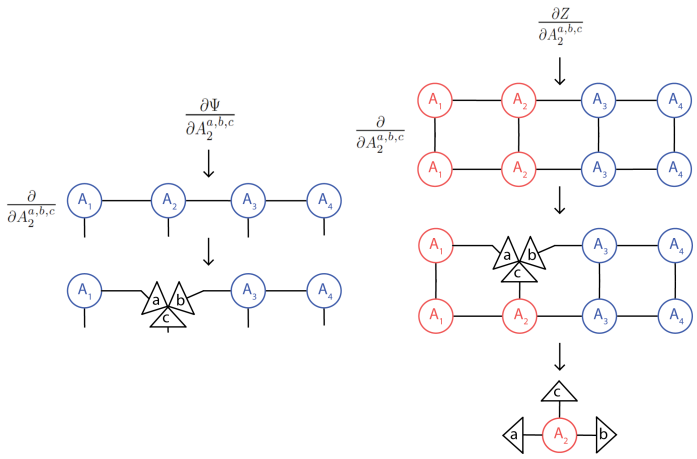


Figure 3-11: (Left) Derivative of the MPS state Ψ with respect to an element (a,b,c) of the second tensor A_2, where we have used the projection rule for derivatives-described in the previous section. (Right) The derivative of the partition function Z with respect to an element (a,b,c) of the second tensor A_2. The final step follows from the property of left (red) and right (blue) canonical form MPS states.

partition function and $\Psi'(v)$ is the partial derivative of the wavefunction amplitude for a given computational state v. Both of these two quantities can be easily represented by tensor network diagrams and efficiently evaluated by low-bond-dimension MPSs, as shown in Fig. 3-11. The above process allows for efficient computation of gradients of $D_{KL}(\Psi_{MPS}(\theta))$, which means that training of the model Ψ_{MPS} can be done using stochastic gradient descent, one of the most effective training algorithms for generative models [43].

One downside of the approach described above is that it does not allow the model to change bond dimension. This means that a programmer can easily set the bond dimension to be higher than needed, making the training take longer than necessary, or may also set the bond dimension too low, making the model perform suboptimally. The approach to an MPS generative model described in [42] and used in this thesis work, which is related to the classic DMRG method with two-site update, is to first contract two of the MPS tensors A_k and A_{k+1} before calculating gradients. This method of calculating gradients allows the bond dimension to change during training based on the computed gradient, allowing the model/data to set their own accuracy.

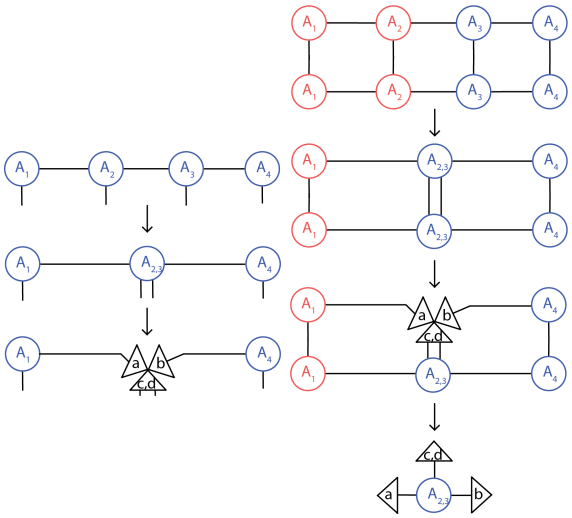


Figure 3-12: (Left) the derivative of the partition function Z with respect to an element of the rank-(2,2) tensor given by contracting A_2 and A_3. (Right) the derivative of the statevector Ψ with respect to an element of the rank-(2,2) tensor given by contracting A_2 and A_3

The full steps for generative model training are provided in the following list:

1. Contract A_k and A_{k+1}
2. Choose an element of the resultant rank-(2,2) tensor $A_{k,k+1}$ to update (i.e., $A^{a,b,c,d}_{k,k+1}$).
3. Take the derivative of Ψ and Z using the same schemes as presented in Fig. 3-12 where we now have a 4-way projector instead of a 3-way projector.
4. Use these derivatives to calculate the D_{KL} gradient
5. Update the entry in $A^{a,b,c,d}_{k,k+1}$ by following the gradient
6. Take the SVD of $A^{a,b,c,d}_{k,k+1}$ to get A_k and A_{k+1} with a newly chosen bond dimension (typically based on some cutoff ratio versus the maximum singular value)
7. Repeat until convergence

In summary, we have explained a method that uses a MPS model to learn training data in a set T through the minimization of the KL-divergence between that data

and the MPS model. We have also shown that it is the gradients and contraction schemes provided by MPS that allow for this training to proceed efficiently.

3.3.2 Optimization Using Generative Models

Now that we have demonstrated a method for using MPS's to learn from a set of training data, it is finally time to apply an MPS-basked generative model to combinatorial optimization via the generative-model-based optimization framework [26]. This final step is simple and proceeds as follows:

1. Start from a random initial MPS state Ψ
2. Find the computational state v which maximized $\Psi(v)$. This is possible using efficient MPS contraction schemes.
3. Test this computational state v on the given problem to be optimized - i.e. apply the cost function to v.
4. Map this cost to a probability value using some [0,1] function, e.g. the sigmoid function.
5. Train the MPs model Ψ (using the method described in the previous section) to output a distribution that matches the known set of costs for the set of known states v_k
6. Repeat until some convergence criteria are met.

This process is described for the particular problem used in this thesis work in Fig. 4-2 in Chapter 4.

3.4 Quantum Generative Models

Quantum generative models often follow in much the same way as described in the previous section [17]. They consider a parameterized wavefunction $\Psi(\theta)$ as a model which outputs a distribution $|\Psi(\theta, v)|^2$. In this case, the wavefunction Ψ is now

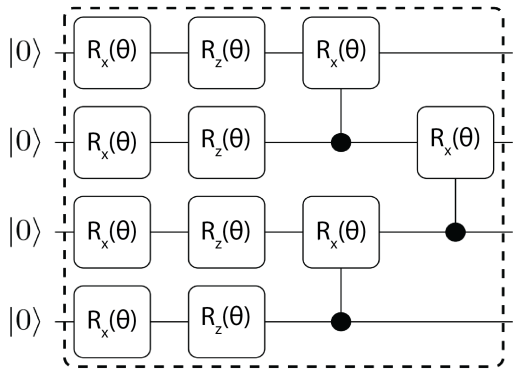


Figure 3-13: An example of a parameterized circuit originally taken from [3]. The R stands for rotation of each qubit about the x or z axis by angle θ.

contained on a quantum computer instead of a classical computer. These quantum generative models are commonly called "Born Machines" in literature, in reference to the postulate the square amplitude of a statevector is equivalent to its measurement probability [44]. This naming convention also contrasts nicely with "Boltzmann Machines", machine learning (and classical spin-glass) models used for modeling classical distributions [45].

The parameters used by these models are quite different from the ones used by MPS. A typical example of a parameterized quantum circuit is given in Fig. 3-13.

Although these wavefunctions are now quantum, classical optimizers are still used to update the parameters θ until convergence. This makes the use of gradient-based optimizers tricky, as they rely on additional quantum measurements/resources. For instance, Ref. [46] used the chain rule, combined with the parameter-shift rule to obtain analytical gradients, at the cost of doubling the number of quantum measurements required. This follows the single-parameter gradient formula for parameterized unitaries:

$$\frac{\partial \langle M \rangle_\theta}{\partial \theta_j} = \frac{\langle M \rangle_{\theta + \frac{\theta_j}{2}} - \langle M \rangle_{\theta - \frac{\theta_j}{2}}}{2}$$

Alternative schemes use finite difference methods to estimate the gradient [47]. Though these schemes can be unstable (the cost function of a parameterized quantum circuit is highly nonlinear), they have been shown to perform well in NISQ contexts [48].

Chapter 4

Quantum-inspired Optimization of Production Planning Problem

4.1 Introduction

This thesis work seeks to address some of the many questions still remaining regarding quantum-inspired optimization, as described in Chapter 3. Specifically, while theoretical work suggests that quantum-inspired optimization provides an enhancement over classical optimization for classical cost functions, it has yet to be proven so [49, 26]. In addition, even though quantum optimization may be beneficial for some problems, there is still the question of its utility for practically relevant cases.

In this chapter, we consider a particular instance of a hard industry-relevant combinatorial optimization problem, production planning in a BMW assembly line. The instance that we consider is realistic and therefore has immediate application for BMW and for the manufacturing sector in general. Using this problem, we numerically compare the performance of quantum-inspired and traditional optimization methods to probe for advantage. Many of the figures and much of the analysis contained in this chapter is drawn from Ref. [50].

We find that the tensor-network generator-enhanced optimization (TN-GEO) method, as describe in Chapter 3.3 ties or outperforms traditional methods of optimization for the BMW production planning problem in a majority of tested cases. The highest

performance is achieved when the problem is formulated such that the correlations between parameters are efficiently encoded. In fact, these "problem inspired" configurations improved performance for all considered optimizers. Our results support that problem-knowledge is an important component in optimizer selection, even among blackbox optimization methods [51, 52, 53]. In addition, we find that intermediate problem knowledge is best for accurate model-based optimization in this setting.

In the following chapter, we describe the BMW manufacturing plant and problem formulations, explain the use of problem knowledge, in particular in regards to preprocessing of cost function evaluations, define the optimizers used for benchmarking and analysis, and show numerical results describing optimizer performance.

4.2 Problem Description

The production planning problem involves the optimization of a BMW assembly line. In a BMW assembly line, as shown diagramatically in Fig. 1-3, car production proceeds as follows:

1. Car bodies are fabricated in a first stage consisting of two parallel body shops.
2. Car bodies are stored in a storage lot with capacity limit 500.
3. Car bodies are painted in a second stage consisting of two parallel paint shops.
4. Painted car bodies are stored in a storage lot with capacity limit 700.
5. Finished cars are assembled in a third stage consisting of two parallel assembly shops.

The problem of optimizing this simulation is constrained such that each shop only operates during specific time periods or shifts and at certain rates. There are 15 discrete options for the shifts that shops can follow and 5 discrete options for the rates. Given that there are 6 different shops, there are about 244 million ways in which the assembly line can be configured.

Over the course of one year, the assembly line is required to produce a number of cars (P_i) to meet production targets (T_i) on the order of thirty thousand cars per month i. At the same time, the line should be as efficient as possible, minimizing the hours per month i during which each shop j is idle (I_{ij}). This idling is due to storage lot overflows or underflows caused by mismatched shop working schedules. These two metrics of performance – production and efficiency – are quantified using a weighted (W) cost function.

$$C = \sum_{i=1}^{12 \text{ months}} |\mathrm{T}_i - \mathrm{P}_i| + \mathrm{W} \sum_{\mathrm{j}=1}^{6 \text{ shops}} \mathrm{I}_{ij}.$$

For the purposes of this thesis work, the weighting (W) has been chosen such that typical variations in idle hours are on the same order as monthly production and target variations. For our specific problem, idle time with respect to scheduled time typically varies at the single percentage level while production variations with respect to targets are on the order of hundreds of cars per month, leading to a weighting of 1000. For a given assembly line state, i.e. the set of shift schedules and production rates used by the line, a time-domain simulation of the assembly line is performed which directly tracks the monthly production and idle hours. The simulation was performed at a half-hour timestep to maintain accuracy while minimizing runtime. An illustration of the time-domain simulation process is given in Fig. 1-3.

4.3 Methods of Problem Representation and Preprocessing

While the general definition of the BMW problem, as described above, holds for the entirety of this thesis work, a typical step in combinatorial optimization involves finding problem representations which make the problem more easily solvable [51]. Therefore, before fully comparing the performance of quantum-inspired and traditional methods, several investigations were performed into the problem's representa-

tion. Specifically, the way in which parameter values, such as shifts and rates, are organized and ordered, before optimization is performed, can have a large impact on time to solution. To investigate these methods of parameter organization, we first consider two potential "parameterizations", or ways of grouping parameters. we then consider three potential binary representations, or encodings of the the chosen parameterizations and incorporate these into a preprocessing step before solving.

4.3.1 Problem Parameterization

Two different parameterizations were investigated in order to better elucidate the impact of problem knowledge on problem performance. By parameterization, we mean the manner in which the production rates and shift schedules can be changed. The first parameterization is a basic “12-body" or “no-knowledge" parameterization. This parameterization considers all parameters as independent with 6 shift parameters, each ranging from 1 to 15, and 6 production rate parameters each ranging from 1 to 5. A change in shift can therefore happen in a single update step (changing a 4 to a 5, for example). This parameterization has the benefit of assuming little about inter-shop interactions, allowing solvers to explore these themselves. A second, “three-body" or “problem-inspired," parameterization is also explored which uses the assembly line structure to inform parameter representation. Each set of body, paint, and assembly shops are combined into a single data entry, such that the problem reduces to only three parameterized stages, with each stage having enumerated states ranging from 1 to 5625. A change in shift may now effectively require several steps as opposed to a single step. Such a parameterization treats correlations between stages as having less impact than those within stages, an assumption based on problem intuition which may not be valid for all parameter settings.

4.3.2 State Encoding

In the “three-body” representation (either reduced or not), every state is designated by a triple of natural numbers. On the other hand, GEO requires states to be represented

as bitstrings (sequences of binary digits). In this work, we compare three different binary encodings. The simplest approach is to enumerate all the triples, and then encode the number of every state as a binary number. We call this the basic encoding.

While simple, the basic encoding may not represent the structure of the problem well. This is because states that are close to each other by parameterization may have significant Hamming distance incurred via the basic encoding. We test a method of alleviating this by applying the Gray encoding to every part of the triple, pad with zeros if necessary and concatenate into a bitstring.

A significant factor which may negatively impact the effectiveness of both of the straightforward representations described above is the fact that nearest neighbours may have significantly different production cost. We test a method of minimizing this impact by enumerating single-stage states of the first stage according to the absolute value of the difference between the estimated production of the first stage and the target production. For the remaining stages we use slightly different ordering, employing the absolute value of the difference between estimated production of the given stage and the estimated production of the first stage. We call this reordering the Production Guided (PG) encoding. Then we apply the Gray encoding to every part of the triple, pad with zeros if necessary, and concatenate into a bitstring. We call it the PGGray encoding.

4.3.3 Search Space Reduction

Preprocessing is utilized to effectively reduce the problem space in a deterministic manner, which is a practical and often necessary step for many large optimization problems [54, 55]. The preprocessed configurations extrapolate the ideal annual car production of each assembly line state when buffer limits are ignored and compare it to the annual production target, i.e. the sum of the monthly targets (see Appendix 5.2.1 for the details). If the state produces enough cars to be within 5%, 2.5%, 2%, or 1.5% of the annual target, then it is included in the allowed state space. The preprocessed configurations are therefore referred to as “Reduced-5%", “Reduced-2.5%", “Reduced-2%", and “Reduced-1.5%" (see Appendix 5.2.1 for details). However, we note that

Table 4.1: Sizes of the reduced solution spaces.

Solution space		Solution space size
2%	noDev	384
2.5%	noDev	1056
5%	noDev	11856
100%	noDev	11390625
1.5%	yesDev	4777500
2%	yesDev	12329982
2.5%	yesDev	22261792
5%	yesDev	202461840
100%	yesDev	177978515625

the minimum reduction percentage that conserves the global minimum is not known *a priori*. This has been verified for the cases studied here.

4.4 Optimization Methods

The traditional optimization methods used in this thesis work include one-crossover, two-crossover, and uniform-crossover genetic algorithms (GA1, GA2 and GAU, respectively), simulated annealing (SA) [35, 56], and parallel tempering (PT) algorithms. These algorithms are the the most commonly investigated algorithms in literature and should provide an even benchmark by which to test quantum-inspired performance [57]. These traditional methods run until 240 cost function evaluations are reached, a limit chosen to fall within typical black-box optimization run times of hundreds of cost function evaluations.

4.4.1 Genetic Algorithms

The genetic algorithm is a meta-heuristic algorithm inspired by the theory of natural evolution [34]. For implementing the Genetic algorithm, we used the DEAP [58] library equipped with elitism mechanism from Ref. [59]. We initialized the solver with a population of 10 individuals. In each iteration, a subset of size 9 of the population from the previous iteration is selected using tournament among 3 individuals. Then, a new population of the same size is generated *via* mutation (with probability 0.8) of

each individual in the subset and crossover (with probability 0.8) of consecutive ones. At the end of iteration, the best individual observed in all previous iterations is added to the population. We used three different types of crossover operations: one-point, two-point and uniform. For mutation, one shop (or one stage in 3-body formulation) is chosen uniformly at random and its state is updated to a random possible state.

4.4.2 Simulated Annealing

For Simulated Annealing (SA), we implement a version of the algorithm based on Ref [35]. In our implementation, starting from an initial state with cost C_0, the initial temperature T_0 is set to 50, and in each iteration, the state of one shop (or one stage in 3-body formulation), chosen uniformly at random, is updated to a random state with probability

$$p_i = \min\{1, \exp((C_{i-1} - C_i)/T_{i-1})\}$$

where C_i is the cost of the state at the end if i-th iteration. At the end of each iteration, the temperature is decreased by a factor of 1.2.

4.4.3 Parallel Tempering

For Parallel Tempering (PT), we implemented a version of the algorithm based on Ref. [60]. PT can be thought of as an advanced version of Simulated Annealing where one considers independent copies of the car plant's state, called replicas, each at a different temperature T_r. In each iteration, first the state of each replica gets updated according to its temperature, as in the SA algorithm. Then, the state of two neighbouring replicas, r and $r+1$, get swapped with probability

$$p_r = \min\{1, \exp((C_r - C_{r+1})(\beta_r - \beta_{r+1})\}$$

where C_r is the cost of the state in r-th replica, and $\beta_r = 1/T_r$. In our implementation, we had 5 replicas with 4 state-updates in each iteration (before each replica swap),

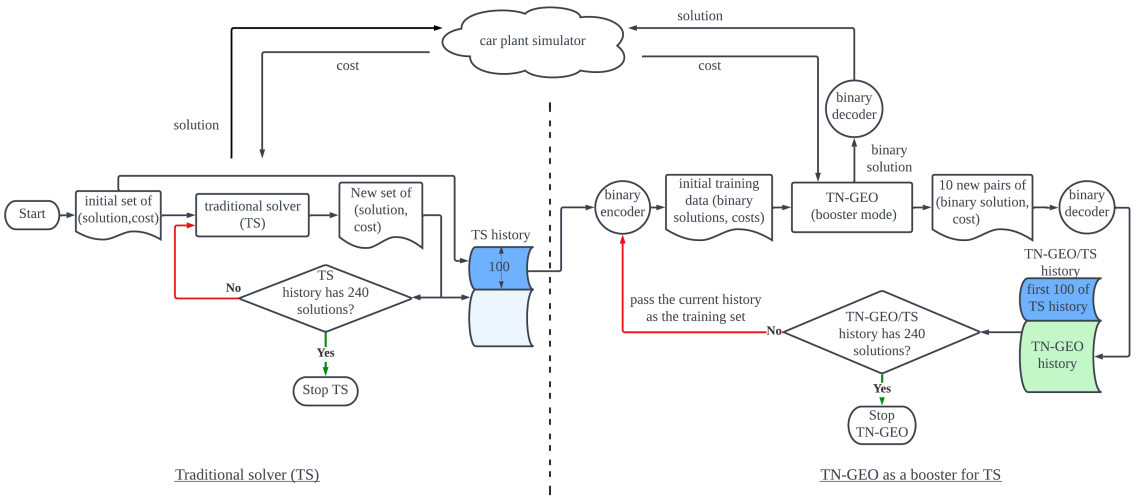


Figure 4-1: Scheme of our approach for benchmarking a traditional solver against TN-GEO as a booster for that traditional solver.

and we used a geometric sequence of evenly spaced (in log space) numbers between 0.1 and 10 for β_r values.

4.4.4 Quantum-inspired Methods

The quantum-inspired model-based optimization step relies on TN-GEO, a method described in [26] and in Chapter 3.3. This optimization works as a enhancement method for traditional methods by building models based on the results of those methods. The specific model used is a Matrix Product State (MPS), a specific type of Tensor Network (TN), an overview of which is given in Chapter 3. When using TN-GEO to boost a traditional solver, the first 100 cost function evaluations of a traditional solver are incorporated into the TN-GEO model and model-based optimization begins. In each iteration of TN-GEO, the model is first trained on a seed data set (evaluated states and their costs) initially generated by the traditional optimization step. This training is performed based on a gradient method used in prior work [42] and described in Chapter 3.3.1. The model is then sampled in order to generate a set of likely low-cost states. From these states, the ones that are new (not evaluated before) are then pruned according to problem constraints and a fixed-size subset of them is chosen (according to some heuristics) to get their costs checked ex-

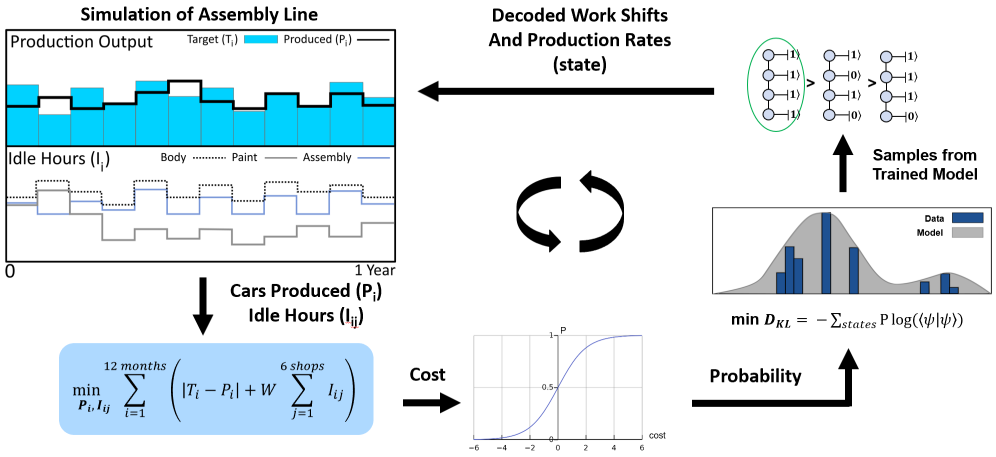


Figure 4-2: The workflow of TN-GEO where a state is chosen from the highest outputs of the MPS model. The workshifts and production rates (decoded from binary) are then simulated to get the cars producted and the idle hours. A cost function is then applied to determine the fitness of the settings. This cost is then mapped to a probability using the sigmoid function. Finally, this probability, as well as any previously calculated probabilities, are used to retrain the MPS model, and the cycle continues.

plicitly on the BMW assembly-line problem. These states and their costs get added to the seed data set and the solver iterates until convergence or reaching to a maximum number of cost evaluations. An overview of this process in general is provided in Fig. 4-1, and specifically for the case of TN-GEO applied to the BMW problem is provided in Fig. 4-2.

4.5 TN-GEO Bond-Dimension

When using TN-GEO, a fixed hyperparameter, the maximum bond dimension, sets the maximum number of singular values kept in the matrix product state factorization of correlation tensors. This corresponds to the level of correlation that can be captured by the model. Too high a maximum bond dimension is likely to overfit a given dataset while too low does not capture relevant features. We tested TN-GEO directly on the studied optimization problem in order to choose the maximum bond dimension that provides best performance. For each bond dimension, we tabulated the number of cases (tested combinations of margins and rate deviation inclusion) in

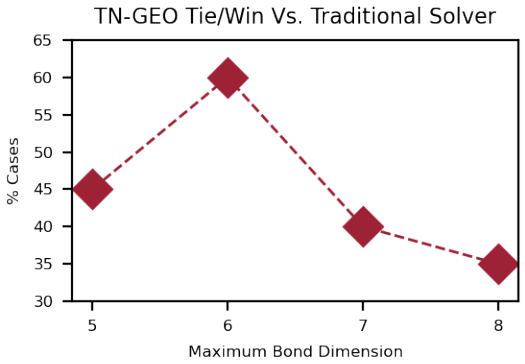


Figure 4-3: The percentage of total tested cases (combinations of margin and rate deviation) in which TN-GEO improves upon the performance of the traditional optimizer. A maximum of cases occurs when a maximum bond dimension of 6 is used for the TN-GEO matrix product state optimizer.

which the TN-GEO optimizer outperformed its traditional counterpart. We anticipated a “sweetspot" of performance would exist across maximum bond dimensions, and found one at a maximum bond-dimension of 6 singular values (see Fig. 4-3) when using 3-body formulation and the production-guided encoding.

4.6 Numerical Results

Our first numerical investigation explored the difference in optimizer performance due to problem parameterization. Each optimizer is implemented as described in 4.4 and allowed to run for an empirically-based choice of 240 cost-function evaluations (TN-GEO) being performed explicitly on only the final 140, as described in 4.4. These sets of optimization trials were then repeated using 300 randomized initial states to acquire statistics. The averaged results of this study are plotted in Fig. 4-4. All optimizers, no matter their type, traditional or quantum-inspired, perform significantly better when the problem is formulated in the 3-body parameterization, achieving lower costs in fewer iterations on average. We also note that, over the course of the optimization 81% to 69% of the runtime was consumed by this simulation.

Following this investigation, we focused solely on the three-body parameterization and performed a comparison of binary encoding methods across all state spaces given

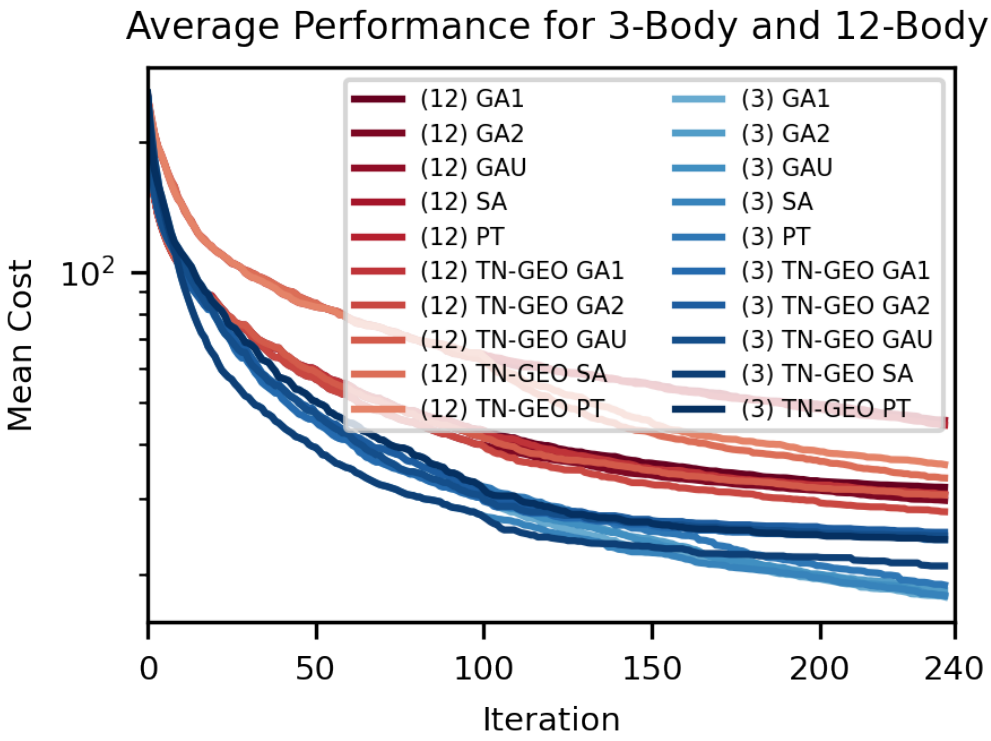


Figure 4-4: Performance of all optimizers for 3-body and 12-body formulations of the problem. We note that 0 as defined in this plot is not the global minimum but instead is the global minimum of a simplified form of the problem, as the true global minimum is not known. See text for the detailed protocol of calculations.

in Table 4.1 (basic, Gray and PGGray encodings). As before, for each scenario, we ran each traditional solver 300 times to accumulate statistics, each time performing up to 240 cost evaluations. A description of the traditional solvers and the hyperparameters we used in these benchmarks are provided in Appendix 4.4.

For TN-GEO an important hyperparameter is the *maximum bond dimension* of the Matrix Product State, which is an indicator of the correlation radius in the training data. For the problem at hand, we tested TN-GEO performance while sweeping this parameter, as shown in 4.5. Through this process we found that a maximum bond dimension of 6 is optimal as it achieves equal or better performance (across solvers and state space sizes) as compared to other bond-dimensions.

The results of these studies (for a maximum bond dimension of 6) are depicted in Fig. 4-5, where there is a heatmap for each state encoding. The number within each heatmap square shows the largest amount (of the 300 runs) by which TN-GEO, when trained using the first 100 cost function evaluations of a traditional solver, could outperform that traditional solver. To facilitate a more detailed analysis Fig. 4-6 shows the performance of the traditional solvers with respect to each other and Fig. 4-7 depicts dependence of the relative performance of TN-GEO using the PGGray encoding with respect to the traditional solver results used as the training set.

Before we take a closer look at the difference between various solvers, let us note that the 3 smallest cases (2% noDev, 2.5% noDev, 5% noDev) are so small that all solvers reached exactly the same minimum (see Figs. 4-6 and 4-7), which we verified to be the global minima for those cases. Hence, to differentiate between the solvers we will use results only for the remaining (larger) state space cases.

From the point of view of the performance of the traditional solvers, SA performs best across the board. However, when we additionally remove the cases where the solution space is not reduced (100% noDev, 100% yesDev), which should not be used in practical applications, and which are apparent outliers, GA2 becomes the best solver, closely followed by PT.

The issue of TN-GEO performance is more complicated. Firstly, the encoding of the states plays a significant role, and the PGGray encoding is the best, as demon-

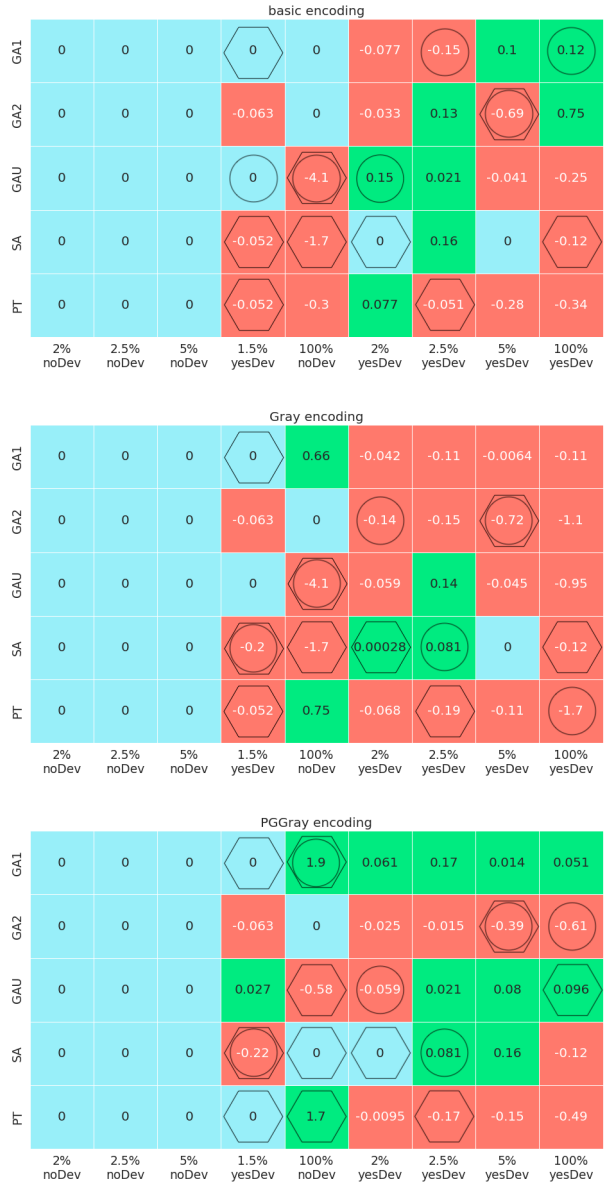


Figure 4-5: Performance of TN-GEO in 3-body formulation. This figure contains one heatmap for each state encoding, each depicting the difference between the lowest cost found by a traditional solver (one of GA1, GA2, GAU, SA or PT) after 300 independent runs, and the lowest cost found by TN-GEO boosting the same traditional solver for the same 300 runs, in different problem configurations (with/without deviation and different margins). The difference is positive (green cells) when the best cost found by TN-GEO is strictly smaller than the best cost found by the traditional solver, zero (blue cells) when they tie in their best run, and negative (red cells) when TN-GEO could not achieve the lowest cost found by the traditional solver. For each problem configuration, there is (at least one) cell with a circle/hexagon indicating that either the corresponding traditional solver or TN-GEO boosting that traditional solver is the worst/best solver among all of the considered solvers. In the event of a tie, multiple circles and hexagons appear.

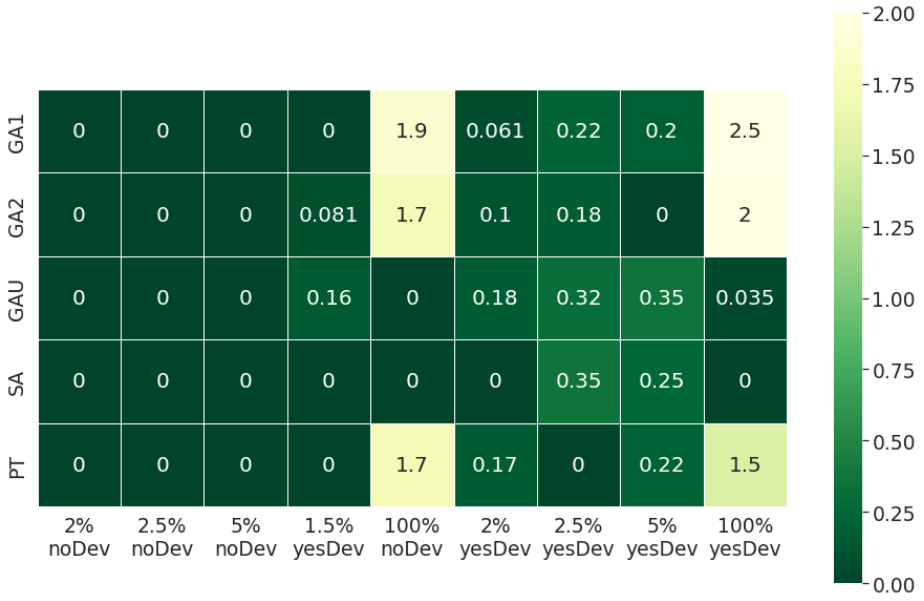


Figure 4-6: Relative performance of the traditional solvers. Each cell in the grid gives the difference between the minimum achieved by the best traditional solver and the minimum achieved by the traditional solver on the left axis. The overlaid heatmap allows for quick comparison of these values with green indicating that the solver performed nearly or exactly as well as the best solver.

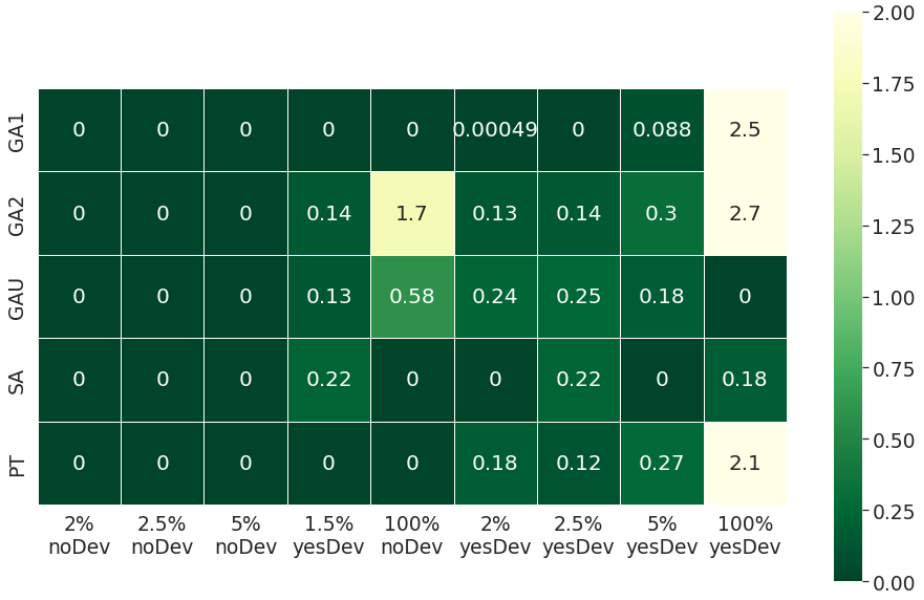


Figure 4-7: Relative performance of TN-GEO trained with data obtained from the traditional solvers. Each cell in the grid gives the difference between the minimum achieved by the best of solvers and the minimum achieved by TN-GEO boosting the solvers on the left axis. This is distinct from Fig. 4-6 which only compares traditional solvers against the best of traditional solver.

strated in Fig. 4-5. Now, looking only at the best encoding, and again disregarding the non-reduced cases, we can see in Fig. 4-7 that TN-GEO trained on the data obtained from GA1 performs best. Given the relative poor results of GA1 with respect to the best traditional solvers, it becomes apparent that the quality of generated training set is not straightforwardly correlated with the performance of a traditional solver.

4.7 Conclusions from this Work

In summary, we compared the performance of a family of quantum-inspired optimization methods with that of a set of conventional methods. Through this comparison it was found that problem representation plays a significant role, with parameter groupings based on known correlations allowing for lower minima to be achieved by all solvers more quickly. In addition, we find that problem-motivated encoding of the assembly line parameters leads to greatly improved results as compared to naïve representations. Combining these two techniques, we show that quantum-inspired generative model based solvers tie or improve upon the tested traditional optimization methods in a majority of tested cases, particularly when used in intermediate solution space sizes. Based on this comparison, we make the general conclusion that TN-GEO ties or outperforms all tested traditional optimization methods in a large number of tested cases, showing the power of these optimization methods in industrially-relevant settings.

Chapter 5

Quantum Optimization on Quantum Hardware: A Proposal

5.1 Quantum Optimization on a Superconducting Processor

In the previous chapter, Chapter 4, quantum-inspired methods were utilized to optimize a BMW production planning problem. The acceptable performance of these methods as compared to conventional solvers makes valid the comparison of quantum solvers as well. In addition, the characterization already performed using tensor networks makes easy the comparison between the quantum model and the known best classical solvers. In this section I give a brief proposal on how to implement these quantum solvers using superconducting processors, based on Chapter 2 and Chapter 3.4.

In this scheme, the same workflow is utilized as in the case of TN-GEO except that parameterized quantum circuits are now utilized instead of tensor networks, as shown in Fig. 5-1. In order to implement these parameterized quantum circuits, the rotation gates shown in the parameterized model will need to be mapped to operations on physical hardware based on Chapter 3. The hardware of-interest has already

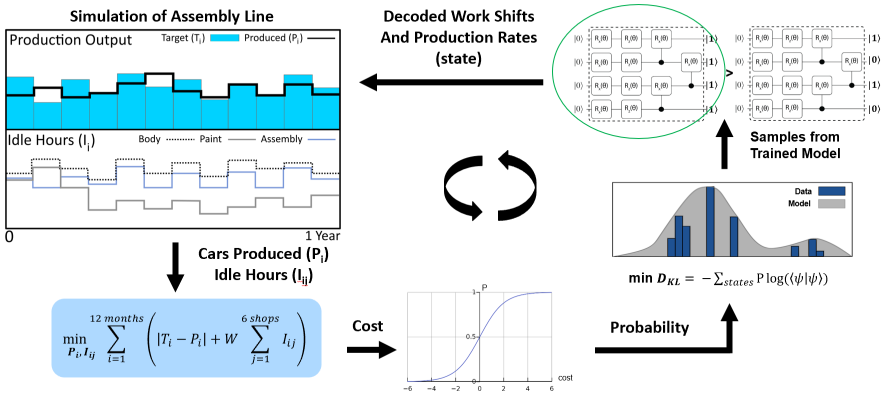


Figure 5-1: A similar workflow as presented in Chapter 4. A set of work shifts and rates is provided to a year-long simulator. The simulator outputs the monthly production and idle hours. These are mapped to a cost. The cost is mapped to a probability using the sigmoid function. The (parameterized quantum) model is trained on this new distribution and the most probable output from this model is taken as the new set of work shifts and production rates to try.

been designed based on Ref. [61] and is currently being fabricated at MIT Lincoln Lab, with a sample chip design displayed in Fig. 5-2. This style of chip prioritizes three things: control simplicity, operation speed, and high number of qubits. In this design, two processing bits are placed on the outer edges of the chip while several storage modes occupy the center region of the chip. The qubits are superconducting transmons, as described at the end of Chapter 2. The storage modes are made up of the hybridized modes of a series of resonant strongly-coupled resonators. In mathematical terms they take the form:

$$\sum_k \omega a_k^\dagger a_k + g(a_k^\dagger a_{k+1} + a_{k+1}^\dagger a_k) \rightarrow \sum_k \tilde{\omega}_k \tilde{a}_k^\dagger \tilde{a}_k \text{ post diagonalization}$$

In the single-excitation subspace these take the form:

$$\begin{bmatrix} \omega & g & 0 & \dots \\ g & \omega & g & \dots \\ 0 & g & \omega & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \rightarrow \begin{bmatrix} \tilde{\omega}_i & 0 & 0 & \dots \\ 0 & \tilde{\omega}_k & 0 & \dots \\ 0 & 0 & \tilde{\omega}_k & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \text{ post diagonalization}$$

Where $\tilde{\omega}_k = \omega - 2g\cos(\frac{k\pi}{n+1})$ and n is the total number of storage modes. For the case of our chip, this number is 10. This number of qubits still only allows for comparison with the two smallest cases presented in Chapter 4, however results could still be suggestive of advantage.

As operations cannot be performed directly on these storage modes, they must be mediated by the processing qubits. This tradeoff of storage and processing modes makes accesible a much larger state space while still allowing for relatively simple control. In addition, because we use hybridized storage modes, our processor exhibits all-to-all coupling between storage modes, making the transpilation process, the process of converting non-native algorithmic gates to on-chip implementable operations, far less costly.

In summary, we already have a processor design that provides for relatively high dimensional problem-solving. It exhibits fast gates due to its all-to-all design and relatively simple control as only the processor qubits need to be calibrated. Given the promising results presented in the preceeding chapter, this experiment may serve as a means for probing practical quantum advantage in combinatorial optimization on a superconducting quantum processor.

5.2 Conclusions and Outlook

This thesis work investigated the utility of quantum-inspired, generator-enhanced optimization methods in a realistic use-case: optimization of a BMW production planning problem, a hard combinatorial optimization problem. This thesis also gave a short proposal for the use of quantum generator-enhanced solvers in this same setting. Through this work find that use of important operations research techniques including the incorporation of domain-specific information as well as increasing state-space pruning improves solver performance as compared to conventional methods up to a certain point: after which both achieve similar performance. In all we find that

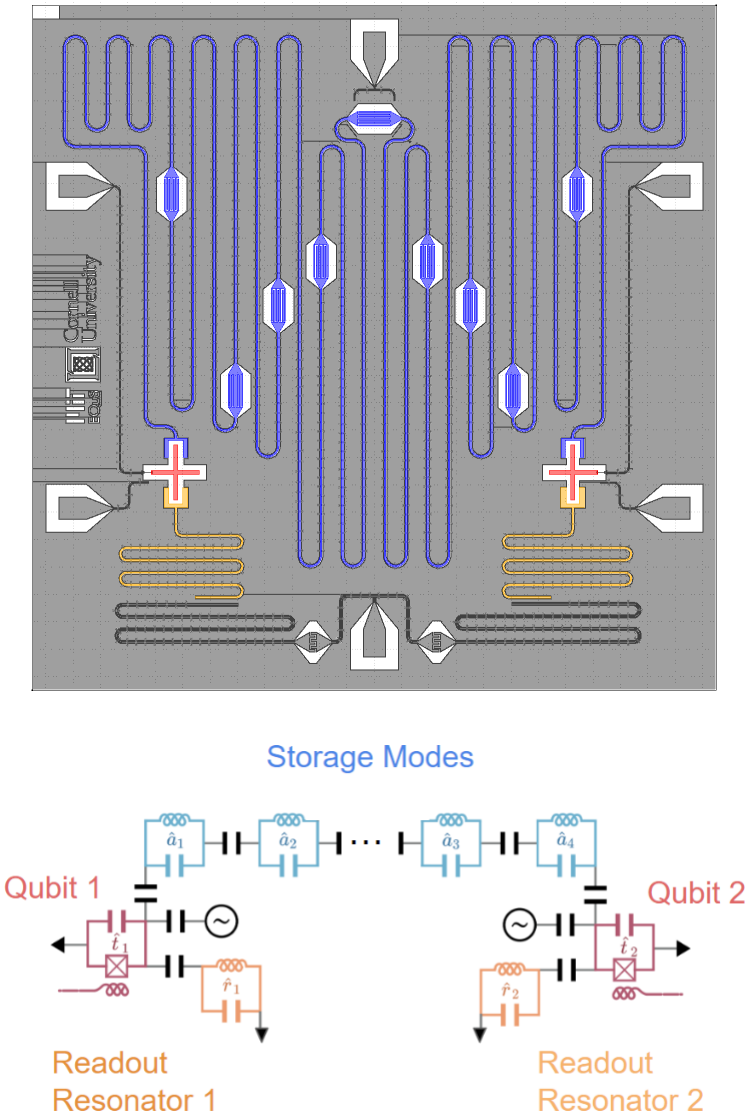


Figure 5-2: (Top) An illustration of the chip-level layout of the processor design with storage modes in blue, qubits in red and readout resonators in orange. Unlabelled are purcell filters whose operation are beyond the scope of this work but can be discussed in more detail in [4]. (Bottom) A circuit diagram of the proposed processor.

quantum-inspired GEO finds lower minima than its conventional counterparts in a majority of tested cases depending on the conventional solvers used for pretraining and the level of state space pruning. This work provides useful characterization of quantum-inspired methods for combinatorial optimization in a new context, serving to identify and provide some explanation for the problem settings which benefit from quantum inspired techniques. This work also provides justification for the proposal laid out in Chapter 5.1 on the quantum implementation with superconducting circuits. It is our hope that this work will provide a template for future research into characterization of these techniques.

Appendix A

5.2.1 The free-stage approximation

Production estimation

In the actual model, the production cost is determined both by the state of every stage (producing cars or not at a given time) and the inter-stage interactions (backlogs in the storage lots). In order to simplify the problem we introduce the free-stage approximation, in which we neglect the inter-stage interactions (storage lots and backlogs). At this level of approximation, we can decouple the model and consider every stage separately.

We will start by introducing $\tilde{p}_{n,m}$ as the estimated monthly production of stage $n \in \{1, \ldots, N\}$ being in the state $m \in \{1, \ldots, M\}$. This quantity can be readily calculated by taking the worktime of the stage obtained from its state, multiplying it by the production rate of the shops comprising the stage, and summing the results. Obviously, for every configuration $m_1, m_2, \ldots, m_N$

$$p_{m_1,m_2,\ldots,m_N} \leqslant \tilde{p}_{m_1,m_2,\ldots,m_N} = \min\{\tilde{p}_{1,m_1}, \tilde{p}_{2,m_2}, \ldots, \tilde{p}_{N,m_N}\}$$

where $p_{m_1,m_2,\ldots,m_N}$ and $\tilde{p}_{m_1,m_2,\ldots,m_N}$ is the is the actual and approximate monthly production, respectively. Additionally, we can expect that

$$\frac{|p_{m_1,m_2,\ldots,m_N} - \tilde{p}_{m_1,m_2,\ldots,m_N}|}{\tilde{p}_{m_1,m_2,\ldots,m_N}} < \varepsilon$$

for a relatively small value of ε, although the concrete value will depend on the

specific parameterization of the model and is difficult to predict accurately. Numerical experiments suggest the ε value for the global minimum configuration between about 0.01 and 0.025, depending on the parameterization.

Reduction of the solution space

For this reason, it should be sufficient to restrict the search for the global minimum to the reduced search space

$$\mathcal{R} = \left\{ (m_1, m_2, \ldots, m_N) \in \mathcal{S} | 1 - \varepsilon \leqslant \frac{\tilde{p}_{i,m_i}}{p_t} \leqslant 1 + \varepsilon) \right\}$$

using the value of of ε estimated above. We note that the cost of the reduction scales proportionally to the number of single-stage states, thus avoiding the exponential explosion.

Bibliography

[1] Jacob Biamonte. Lectures on quantum tensor networks, 2020.

[2] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, 2014.

[3] Sukin Sim, Peter D. Johnson, and Alán Aspuru-Guzik. Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. *Advanced Quantum Technologies*, 2(12):1900070, 2019.

[4] Eyob A Sete, John M Martinis, and Alexander N Korotkov. Quantum theory of a bandpass purcell filter for qubit readout. *Physical Review A*, 92(1):012325, 2015.

[5] William A. Fedak and Jeffrey J. Prentis. The 1925 Born and Jordan paper "On quantum mechanics". *American Journal of Physics*, 77(2):128–139, 02 2009.

[6] David J. Griffiths and Darrell F. Schroeter. *Introduction to quantum mechanics*. Cambridge University Press, Cambridge ; New York, NY, third edition edition, 2018.

[7] W. Shockley. The theory of p-n junctions in semiconductors and p-n junction transistors. *The Bell System Technical Journal*, 28(3):435–489, 1949.

[8] T. H. Maiman. Stimulated optical radiation in ruby. *Nature*, 187(4736):493–494, Aug 1960.

[9] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, Jun 1982.

[10] Yuri Manin. Computable and uncomputable. *Sovetskoye Radio*, 128, 1980.

[11] P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[12] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 212–219, New York, NY, USA, 1996. Association for Computing Machinery.

[13] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018.

[14] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5(1):4213, Jul 2014.

[15] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm applied to a bounded occurrence constraint problem, 2015.

[16] A.B. Finnila, M.A. Gomez, C. Sebenik, C. Stenson, and J.D. Doll. Quantum annealing: A new method for minimizing multidimensional functions. *Chemical Physics Letters*, 219(5):343–348, 1994.

[17] Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. Parameterized quantum circuits as machine learning models. *Quantum Science and Technology*, 4(4):043001, nov 2019.

[18] Jarrod R McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics*, 18(2):023023, 2016.

[19] Tameem Albash and Daniel A Lidar. Demonstration of a scaling advantage for a quantum annealer over simulated annealing. *Physical Review X*, 8(3):031016, 2018.

[20] Bin Yan and Nikolai A Sinitsyn. Analytical solution for nonadiabatic quantum annealing to arbitrary ising spin hamiltonian. *Nature Communications*, 13(1):2212, 2022.

[21] Phillip C Lotshaw, Thien Nguyen, Anthony Santana, Alexander McCaskey, Rebekah Herrman, James Ostrowski, George Siopsis, and Travis S Humble. Scaling quantum approximate optimization on near-term hardware. *Scientific Reports*, 12(1):12388, 2022.

[22] Gian Giacomo Guerreschi and Anne Y Matsuura. Qaoa for max-cut requires hundreds of qubits for quantum speed-up. *Scientific reports*, 9(1):1–7, 2019.

[23] Brian Coyle, Daniel Mills, Vincent Danos, and Elham Kashefi. The born supremacy: quantum advantage and training of an ising born machine. *npj Quantum Information*, 6(1):60, 2020.

[24] Steven R. White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, 69:2863–2866, Nov 1992.

[25] Samuel Mugel, Carlos Kuchkovsky, Escolástico Sánchez, Samuel Fernández-Lorenzo, Jorge Luis-Hita, Enrique Lizaso, and Román Orús. Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks. *Phys. Rev. Res.*, 4:013006, Jan 2022.

[26] Javier Alcazar, Mohammad Ghazi Vakili, Can B. Kalayci, and Alejandro Perdomo-Ortiz. GEO: Enhancing combinatorial optimization with classical and quantum generative models, 2021.

[27] S. Bond-Taylor, A. Leach, Y. Long, and C. G. Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and

autoregressive models. *IEEE Transactions on Pattern Analysis amp; Machine Intelligence*, 44(11):7327–7347, nov 2022.

[28] Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, Dec 1998.

[29] Google AI Quantum, Collaborators*†, Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Sergio Boixo, Michael Broughton, Bob B Buckley, et al. Hartree-fock on a superconducting qubit quantum computer. *Science*, 369(6507):1084–1089, 2020.

[30] Yunseong Nam, Jwo-Sy Chen, Neal C Pisenti, Kenneth Wright, Conor Delaney, Dmitri Maslov, Kenneth R Brown, Stewart Allen, Jason M Amini, Joel Apisdorf, et al. Ground-state energy estimation of the water molecule on a trapped-ion quantum computer. *npj Quantum Information*, 6(1):33, 2020.

[31] Steven M. Girvin. Circuit qed: Superconducting qubits coupled to microwave photons, July 2011.

[32] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[33] Arne L. Grimsmo, Baptiste Royer, John Mark Kreikebaum, Yufeng Ye, Kevin O'Brien, Irfan Siddiqi, and Alexandre Blais. Quantum metamaterial for broadband detection of single microwave photons. *Phys. Rev. Appl.*, 15:034074, Mar 2021.

[34] John Holland. *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, 1992.

[35] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[36] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1):29–41, 1996.

[37] Mark Zlochin, Mauro Birattari, Nicolas Meuleau, and Marco Dorigo. Model-based search for combinatorial optimization: A critical survey. *Annals of Operations Research*, 131(1):373–395, Oct 2004.

[38] Yaochu Jin, M. Olhofer, and B. Sendhoff. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation*, 6(5):481–494, 2002.

[39] Aishwaryaprajna and Jonathan E. Rowe. Evolutionary and Estimation of Distribution Algorithms for Unconstrained, Constrained and Multi-objective Noisy Combinatorial Optimisation Problems. *Evolutionary Computation*, pages 1–27, 02 2023.

[40] Jiaqiao Hu, Michael C. Fu, and Steven I. Marcus. A model reference adaptive search method for global optimization. *Operations Research*, 55(3):549–568, 2007.

[41] Ulrich Schollwöck. The density-matrix renormalization group in the age of matrix product states. *Annals of Physics*, 326(1):96–192, 2011. January 2011 Special Issue.

[42] Zhao-Yu Han, Jun Wang, Heng Fan, Lei Wang, and Pan Zhang. Unsupervised generative modeling using matrix product states. *Phys. Rev. X*, 8:031012, Jul 2018.

[43] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[44] Brian Coyle, Daniel Mills, Vincent Danos, and Elham Kashefi. The born supremacy: quantum advantage and training of an ising born machine. *npj Quantum Information*, 6(1):60, Jul 2020.

[45] David Sherrington and Scott Kirkpatrick. Solvable model of a spin-glass. *Physical review letters*, 35(26):1792, 1975.

[46] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. Quantum circuit learning. *Phys. Rev. A*, 98:032309, Sep 2018.

[47] James C. Spall. A one-measurement form of simultaneous perturbation stochastic approximation. *Automatica*, 33(1):109–112, 1997.

[48] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature*, 549(7671):242–246, Sep 2017.

[49] Xun Gao, Eric R. Anschuetz, Sheng-Tao Wang, J. Ignacio Cirac, and Mikhail D. Lukin. Enhancing generative models via quantum correlations. *Phys. Rev. X*, 12:021037, May 2022.

[50] William P. Banner, Shima Bab Hadiashar, Grzegorz Mazur, Tim Menke, Marcin Ziolkowski, Ken Kennedy, Jhonathan Romero, Yudong Cao, Jeffrey A. Grover, and William D. Oliver. Quantum inspired optimization for industrial scale problems, 2023.

[51] Florian Arnold and Kenneth Sörensen. What makes a vrp solution good? the generation of problem-specific knowledge for heuristics. *Comput. Oper. Res.*, 106:280–288, 2018.

[52] Tirtharaj Dash, Ashwin Srinivasan, and A. Baskar. Inclusion of domain-knowledge into gnns using mode-directed inverse entailment. *Mach. Learn.*, 111:575–623, 2021.

[53] Tirtharaj Dash, Sharad Chitlangia, Aditya Ahuja, and Ashwin Srinivasan. A review of some techniques for inclusion of domain-knowledge into deep neural networks. *Scientific Reports*, 12(1):1040, Jan 2022.

[54] Flavien Lucas, Romain Billot, Marc Sevaux, and Kenneth Sörensen. Reducing space search in combinatorial optimization using machine learning tools. In *Learning and Intelligent Optimization: 14th International Conference, LION 14, Athens, Greece, May 24–28, 2020, Revised Selected Papers*, page 143–150, Berlin, Heidelberg, 2020. Springer-Verlag.

[55] Joao Guilherme de Carvalho Costa, Yi Mei, and Mengjie Zhang. Guided local search with an adaptive neighbourhood size heuristic for large scale vehicle routing problems. *Proceedings of the Genetic and Evolutionary Computation Conference*, 2022.

[56] Robert H. Swendsen and Jian-Sheng Wang. Replica monte carlo simulation of spin-glasses. *Phys. Rev. Lett.*, 57:2607–2609, Nov 1986.

[57] Jann Michael Weinand, Kenneth Sörensen, Pablo San Segundo, Max Kleinebrahm, and Russell McKenna. Research trends in combinatorial optimization. *International Transactions in Operational Research*, 29(2):667–705, 2022.

[58] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner Gardner, Marc Parizeau, and Christian Gagné. Deap: Evolutionary algorithms made easy. *J. Mach. Learn. Res.*, 13(1):2171–2175, jul 2012.

[59] Eyal Wirsansky. *Hands-On Genetic Algorithms With Python*. Packt Publishing, 2020.

[60] Ulrich H.E. Hansmann. Parallel tempering algorithm for conformational studies of biological molecules. *Chemical Physics Letters*, 281(1):140–150, 1997.

[61] RK Naik, N Leung, S Chakram, Peter Groszkowski, Y Lu, Nathan Earnest, DC McKay, Jens Koch, and David I Schuster. Random access quantum information processors using multimode circuit quantum electrodynamics. *Nature communications*, 8(1):1904, 2017.