Multi-objective Evolutionary Algorithms for the Development of NISQ Hardware Optimised Quantum Circuits

Christian Wood

November 23, 2024

Abstract

This literature review presents an overview of the existing literature around quantum circuits and the challenges being faced in the NISQ era of quantum hardware, alongside different approaches used to help mitigate them. We provide an overview of quantum principles before taking a deeper dive into the design of quantum circuits with multi-objective evolutionary algorithms, looking at how they could leverage existing developments presented in the literature. Finally, we present a hypothesis and a project plan for a dissertation project focusing on the development of a multi-objective evolutionary algorithm for the creation of a NISQ hardware-optimised version of Quantum Fourier Transform.

Contents

☒ **I have not used any GenAI tools in preparing this assessment.**

☒ **I certify that all material in this dissertation which is not my own has been identified.**

Signature: Signed by: Christian Wood 4660BCF46B344F5...

1 Introduction

As a field, quantum computing theoretically promises the exponential speed-up of challenges currently deemed computationally intractable, (see Shor's algorithm for factorisation of large numbers [30]), with a vast set of potential applications to enable huge industrial developments. For example in pharmaceutical research, as presented in *Potential of quantum computing for drug discovery* [4], where quantum computing's ability to model complex molecules and chemical reactions is currently under exploration, or in financial services, where quantum algorithms are being researched to enhance the efficiency of Monte Carlo simulations for the pricing of derivatives [22].

However, despite quantum computing's ability to provide exciting new research capabilities, the development of quantum circuits by hand is notoriously challenging due to their probabilistic and therefore unintuitive nature. Fortunately, other approaches for the automatic creation of quantum circuits do exist, such as the use of Evolutionary Algorithms (EAs) outlined in *Evolving quantum circuits using genetic algorithm* [18]. This enables us to do away with some of the complexity surrounding quantum circuit discovery, and search the quantum circuit solution space far more efficiently.

Additionally, as Preskill presented in *Quantum Computing in the NISQ era and beyond* [25], we see a broad set of challenges caused by the relatively early stage of development in quantum hardware we are in. This era is known as the Noisy Intermediate Scale Quantum (NISQ) era, where we face challenges with scale[1], high sensitivity to environmental noise and connectivity constraints between qubits, lead to less scalable and more error-prone quantum circuits, necessitating research into both hardware and software solutions. While hardware advancements aim to improve qubit fidelity, coherence, and scalability of the processors themselves, software approaches as outlined in *An overview of quantum error mitigation formulas* [26] like zero-noise extrapolation, probabilistic error cancellation, and symmetry verification instead aim to mitigate noise-related issues. One other exciting software approach suggested in *Multi-objective design of quantum circuits using genetic programming* [29] is the use of Multi-Objective Evolutionary Algorithms (MOEAs), a specialised form of EA which enables the automatic design of quantum circuits while optimising other criteria such as noise resilience or adherence to connectivity constraints.

Given these considerations, we will conduct a literature review in **(section 2)** to provide a clearer understanding of the subject matter and identify how we can leverage and build on existing knowledge to improve circuit resilience to NISQ era issues, specifically focusing on MOEA based approaches.

2 Literature Review

2.1 Background to Quantum Computing

Quantum computing is a development in computing hardware based on the principles of quantum mechanics. At its core, Quantum computing relies on the principle of superposition first presented in *A Superposition Principle in Physics* [13], where a quantum particle can exist in a combination of two states simultaneously, with probabilities that define its likelihood of being measured as one or the other. This ability to exist in multiple states simultaneously alongside this probabilistic manipulation is what gives quantum computers the potential to tackle traditionally intractable problems [20].

This quality is harnessed in qubits[2], where a quantum particle remains in a superposition of two states $|0\rangle$ and $|1\rangle$[3] with terms α and β which define the likelihood of being measured as one or the

[1] The largest processors to date having qubit counts in the low thousands, such as IBM's Condor with 1,121 qubits [15]

[2] Common name for quantum bits

[3] Also denoted by the vectors $\begin{bmatrix}1\\0\end{bmatrix}$ and $\begin{bmatrix}0\\1\end{bmatrix}$ respectively

other through $|\alpha|^2$ and $|\beta|^2$ respectively. This combines to provide us with the formula for a qubit phi's state:

$$|\phi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha\begin{bmatrix}1\\0\end{bmatrix} + \beta\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}\alpha\\0\end{bmatrix} + \begin{bmatrix}0\\\beta\end{bmatrix} = \begin{bmatrix}\alpha\\\beta\end{bmatrix},\ where\ |\alpha|^2 + |\beta|^2 = 1$$

When measured, the qubit is said to "collapse" into one of the two possible states, probabilistically taking one of the two possible values. However, before a qubit has been measured and therefore collapsed into one of the two possible states, we can apply a variety of unitary operations known as quantum gates [9] on the qubit, which are represented as n×n[4] complex-valued matrices. These gates alter the state of the qubit, changing the parameters α and β and therefore the probabilities of the particle taking either state, enabling the processing of a qubit. For example let's look at the application of a Pauli-Not gate (represented by X), which flips the probabilities of state measurement, to the qubit ϕ from the example above:

$$X|\phi\rangle = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}|\phi\rangle = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}\alpha\\\beta\end{bmatrix} = \begin{bmatrix}\beta\\\alpha\end{bmatrix} = \beta|0\rangle + \alpha|1\rangle$$

Additionally, we can arrange multiple qubits into a quantum circuit. The combined state of two or more qubits is represented using what is known as the Tensor product (represented by $\otimes$). The probability of having qubit states the following qubit combinations $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$ is as follows[5]:

$$|\phi\rangle \otimes |\psi\rangle = \begin{bmatrix}\alpha_\phi\\\beta_\phi\end{bmatrix} \otimes \begin{bmatrix}\alpha_\psi\\\beta_\psi\end{bmatrix} = \begin{bmatrix}\alpha_\phi\alpha_\psi\\\alpha_\phi\beta_\psi\\\beta_\phi\alpha_\psi\\\beta_\phi\beta_\psi\end{bmatrix}$$

By arranging multiple qubits in quantum circuits, similar to logic circuits in classical computing, we can build larger circuits that are able to perform complex multi-qubit processing. Circuits are typically represented using circuit diagrams such as in **Figure 1** representing a Quantum Fourier Transform:

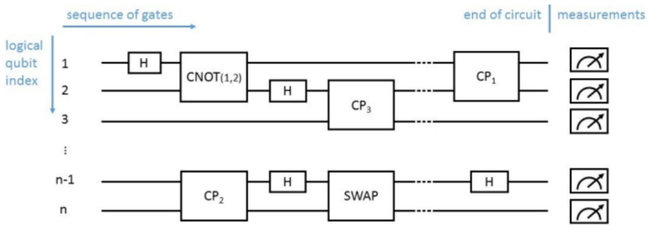

Figure 1: Quantum Fourier Transform Circuit Diagram

2.2 Areas for Optimisation for NISQ Era Performance

As briefly touched upon in the introduction, low noise resilience and connectivity constraints are significant challenges holding back the area of quantum computing.

As detailed in *Introduction to quantum noise, measurement, and amplification* [6], quantum noise is the result of the inherent uncertainty present in quantum mechanics and has the effect of introducing errors to qubit states. As presented, the probability of errors is positively correlated to the length

[4] Typically 2x2 although higher order gate matrices exist

[5] Note left-to-right coincides with top-to-bottom

of time a qubit is in superposition, the result for quantum computing being that a circuit's runtime is therefore positively correlated to the error rate caused by quantum noise[31]. Consequently, one way to optimise a circuit's noise resilience is to minimise its circuit's circuit depth[6], as discussed in *Navigating the noise-depth tradeoff in adiabatic quantum circuits* [2].

Another critical factor to consider is gate fidelity, which refers to how gates perform compared to their ideal operation, with fidelity being a metric for their error rate. As highlighted in *Universal quantum gate set approaching fault-tolerant thresholds with superconducting qubits.* [5] Quantum noise impacts a gate's fidelity to varying extents according to the type of gate. For example, a CNOT gate is shown to have the lowest fidelity, scoring 0.9507. By contrast, the highest fidelity gate, a single-qubit rotation gate has a fidelity of 0.9687 (ignoring the identity gate which has a fidelity of 0.9691). When gates are placed in a circuit these errors accumulate, leading to higher error rates in the overall circuit. Therefore, through the use of higher-fidelity gates, we can design circuits with higher overall fidelity.

As mentioned in *Quantum Computing in the NISQ era and beyond* [25] Qubit connectivity is another key NISQ era challenge, impacting a circuit's ability to scale. In physical circuits[7], qubits are currently only able to sustain certain connections to certain other qubits, which varies by physical implementation. This can lead to certain operations not being physically possible without the prior application of "SWAP gates", which switches the states of two connected qubits. These SWAP gates are composed of three consecutive CNOT gates, significantly adding to our circuit's depth as well as decreasing our circuit's fidelity through the use of the more error-prone CNOT gates. Various techniques for SWAP gate minimisation such as that discussed in *Layout Optimization for Quantum Circuits with Linear Nearest Neighbour Architectures* [24] and typically run after the development of the logical circuit when mapping the logical circuit onto a physical circuit.

2.3 Multi-Objective Evolutionary Algorithms

The general format of any MOEA is very similar in nature to that of the standard evolutionary algorithm structure presented in *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence* [14], having three basic components; a genome representation of candidate solutions, a fitness function, and a set of genetic operators that act on the genome. However, unlike standard evolutionary algorithms, the MOEA approach utilises various, sometimes conflicting fitness functions, each representing a distinct objective of the problem being solved. MOEAs operate with the goal of identifying a set of Pareto-optimal solutions known as the Pareto-front [8], where competing fitness criteria are balanced, and any improvement to one fitness criteria would negatively impact the others.

As summarised in *Multiobjective evolutionary algorithms: A survey of the state of the art* [32], MOEAs can be categorised based on their approach to handling competing objectives. Common types include indicator-based MOEAs, preference-based MOEAs, and hybrid MOEAs, each suited to specific optimisation challenges. Indicator-based MOEAs primarily prioritise the exploration of the diversity (or "width") of the Pareto front, using metrics such as hypervolume or epsilon indicators [11] to "indicate" areas for exploration, helping to ensure a well-distributed set of solutions while simultaneously encouraging convergence towards the Pareto-front. By contrast, Preference-based MOEAs focus on leveraging user-defined priorities, such as a weighted sum, rankings, or thresholds of component fitnesses, guiding the search toward specific regions on the Pareto front, enabling a more targeted (or "deep") form of exploration in areas aligned with user objectives. Lastly, Hybrid MOEAs offer a blend of the two, combining indicators for diversity with user-defined preferences for targeted depth and making them adaptable to complex problems where exploration and exploitation are valuable.

Another approach suggested as appropriate to circuit design in *Multiobjective evolutionary algorithms: A survey of the state of the art* Table 2 and explored in more detail in *Evolutionary multiobjec-*

[6] A measure of its time complexity

[7] Where a circuit design is mapped to physical qubits

tive design of combinational logic circuits [7] is the use of a decomposition based MOEA (MOEA/D) that breaks multi-objective optimisation problems (MOPs) into various scalar objective optimisation problems (SOPs) that are separately optimised. Optimal solutions found are then shared with neighbouring SOPs influencing their optimisation. This approach works particularly well when neighbouring solutions offer incremental improvements, however with regards to quantum circuit design unlike in traditional circuit design, even slight changes to a circuit's structure (like modifying a few gates) can sometimes lead to non-linear effects on output therefore making this approach less appropriate.

Another key component of an MOEA is the Decision Maker (DM) which will decide between various component solutions on the Pareto front. Various forms of DMs exist, including priori, posteriori, and interactive approaches as outlined in *Nonlinear multiobjective optimization* [19]. As discussed, a priori DM relies on all preference information being made available before the commencement of the optimisation, enabling the conversion of the MOP to an SOP, which is an MOEA with a single fitness measure. This is typically done through similar methods to preference-based MOEAs leading to them working well together. By contrast with a posteriori approach, the DM only comes into play once the MOEA has generated a well-distributed approximation of the Pareto-front, from where it will then review the set of solutions to select the ones that most closely conform to its preferences. This makes posteriori DMs particularly suited to Indicator-based MOEAs which focus on the generation of a wide set of solutions on the Pareto front. Lastly, we have the interactive DM, which requires the most active involvement from the DM. This approach focuses on the interaction between the MOEA and the DM, with the MOEA presenting intermediate solutions to the DM, which then adapts the search strategy accordingly.

2.4 Representations of Quantum Circuits

One main feature to consider when implementing MOEAs is the representation of candidate circuits. This is explored extensively throughout the literature, with many different representations used by researchers. These representations tend to vary according to the evolutionary algorithm's optimisation goals, due to the critical impact of the representation of logical circuits in determining the set of candidate solutions the algorithm can represent and therefore evolve.

One interesting approach outlined in *Evolving Quantum Circuits using Genetic Programming* [28], relies on serial, parametrised gates, with the set of parameters taken by each gate being predefined in a separate table. For example, a CNOT gate will take a parameter indicating the target qubit(s) (qubits acted upon by operation) and a separate parameter indicating the control qubit(s) (qubits controlling the operation). This representation importantly enables the representation of complex gate operations between non-adjacent qubits (the connectivity between these qubits will be left to the physical implementation to resolve) but has its shortcomings in that it is unable to represent circuits with parallelised gate operations, a crucial consideration when looking at minimising circuit depth as part of your MOEA.

Conversely, another popular approach presented across much of the literature relies on breaking circuits down into blocks representing timesteps, such as shown in *Evolving quantum circuits using genetic algorithm* [18] or similarly in *On the Evolutionary Design of Quantum Circuits* [27]. In their particular implementation of this block based approach, Lukac and Perkowski present blocks with a fixed number of qubits. Within each block, various strings represent the different possible operations applied on each of the block's component qubits. The example in the paper uses the string "W" to represent a wire (i.e. where no operation is executed on a qubit), "S" to represent a SWAP operation, and "CCNOT" to represent a controlled-not gate with 2 target cubits and one control qubit. We should note that in this representation each gate has a predefined number of qubits it acts on, meaning that although the number of gates in each block may not be the same, the number of qubits acted upon is. We also see the start and end of each block are being represented with a character "p", enabling the adjoining of various blocks into a chromosome representing a logical quantum circuit. Unlike Rubinstein's approach, this block-based representation sacrifices the ability to represent gates between non-adjacent qubits, possibly a constraint present in Lukac and Perkowski's implementation hardware. However, this is not the case for all hardware meaning this poses a severe drawback to this block-based approach due to its limitation on the set of circuits we can represent without the use of SWAP gates, which as discussed are computationally expensive and are typically added at the hardware implementation stage. That said, the block-based representation of circuits has one key advantage over Rubinstein's serial representation, which is its ability to encode circuits with parallelised gate operations, which may allow for the representation of more optimal circuits.

2.5 Fitness Function Constituents

As presented in *Multiobjective evolutionary algorithms: A survey of the state of the art* [32], the next step in an MOEA is to establish a fitness function composed of various competing criteria. For NISQ-era quantum circuits, these functions need to balance competing objectives to optimise solutions with regard to the challenges outlined in **section 2.2**. How these various components are actually balanced will depend on the choice of MOEA approach as discussed in **section 2.3**.

One important feature of an MOEA for quantum circuit design is the measuring of candidate solutions' overall fidelity, that is how closely it replicates the behaviour of our target algorithm. This is important as it drives the evolution of candidate solutions towards areas of the search space that accurately replicate the target algorithm's behaviour. One approach is the use of Mean Square Fidelity (MSF) as presented in *Quantum Program Synthesis: Swarm Algorithms and Benchmarks* [1], which compares the output of our circuit, presented as $A|x_i\rangle$ with the target state $|y_i\rangle$, and calculates fidelity over a set of input-output pairs.

$$MS\mathcal{F}(A, X, Y) = \frac{\sum_{i=1}^{|X|} \mathcal{F}\big(|y_i|, A\,|x_i|\big)^2}{|X|}$$

A key feature of MSF is it ignores global phase differences, as states differing only by a global phase produce identical experimental results[1], making MSF particularly suitable for quantum circuit fitness evaluation.

With regard to other optimisation objectives, various other forms of fitness function could be applied, although they are not as extensively researched as circuit fidelity and tend to be much more dependent on the representation and the optimisation criteria at hand, providing an area where our approach may be able to make headway in the field.

2.6 Genetic Operators for Quantum Circuit Generation

As presented in Holland's original book [14], there are two standard approaches inspired by nature for producing a new generation of candidate solutions; mutation and crossover. The actual implementation of these genetic operators tends to vary highly, due to its dependence on the representation of candidate solutions, but the underlying strategy is consistent throughout the literature.

Crossover refers to the re-combination of existing solutions' chromosomes to form new candidate solutions, enabling the exploitation of successful solutions. This strategy's implementation varies by representation, as different representations may allow different forms of crossover. For example in Rubinstein's serial representation [28] discussed in **section 2.4**, two different crossovers take place. In the first place, we have crossover at the gate level by selecting a random gate from each parent circuit and will swap all gates after this between the two parents. This approach also performs crossover at the parameter level, where we crossover between strings representing qubits of the same category. Rubinstein also presents an exploration between uniform crossover and weighted crossover, differing in their probabilities of a particular gate being selected as the crossover point, however, no conclusion is reached or discussed regarding their relative impact on the algorithm's performance. By contrast in Lukac and Perkowski block-based implementation [18], crossover is instead performed at a block level, instead of at a gate level, achieved through the selection of an index and switching all blocks after this index between parents to product the new generation.

Mutation also has a crucial role to play, being responsible for devising new components. Although these new components may or may not be more successful than existing ones, this operation plays a crucial role in enabling a successful exploration of the solution space. The implementation of this is again dependent on the representation. In Rubinstein's approach, mutation is implemented at the gate level, where a gate is replaced with a new gate with some small random probability. In Lukac and Perkowski's mutation is performed at a variety of levels, including at the gate level where a gate may change to another or may switch positions with another gate in the block with some given probability, or at the block level where blocks may be added or removed.

Evaluating these two approaches, we can identify one important disadvantage in the block-based approach as presented by Lukac and Perkowski crossover when compared to Rubinstein's; because crossover occurs at the block level, as opposed to at the gate level, Lukac and Perkowski's approach may lead to a less exploratory algorithm, perhaps leading to a slower exploration of the candidate solution space [8]. However, this may not be an issue due to the difference in implementations regarding mutation, where we can observe Lukac and Perkowski's implementation to have more forms of mutation which may make up for its less fine-grained approach to crossovers regarding its exploration of the candidate solution space.

2.7 Quantum Circuit Optimisation Challenges

When looking at potential quantum circuits we could optimise for execution on NISQ-era hardware, various problems jump out in the literature, namely, Grover's algorithm, Quantum Phase Estimation (QPE), and Quantum Fourier Transform (QFT) which play crucial roles in complex applications like quantum-enhanced machine learning [17] and Shor's algorithm [30].

Grover's algorithm [12] is primarily used in searching for items within unstructured datasets. Where traditional algorithms execute with a time complexity of O(n), as on average half of the dataset must be searched before the item is found, Grover's algorithm is able to leverage quantum's principles of superposition and interference to amplify the probability of finding the correct item in an unsorted database, with an $O(\sqrt{n})$ time-complexity, offering a quadratic speed-up over classical algorithms. Grover's algorithm involves repeated iterations of an oracle operation (which inverts the phase of the correct item) followed by a diffusion operator, a combination of Hadamard gates and conditional phase inversion, to amplify the probability of the correct result. This iterative process continues for a predetermined number of iterations. The number of iterations of this process progressively increases the probability of finding the item until we reach an optimal suggested number of iterations of $\frac{\pi}{4}\sqrt{N}$ [3]. However, this approach to identifying the optimal number of iterations doesn't factor in the decreasing noise resilience of the circuit as we increase circuit depth with progressive iterations. Therefore, using MOEA we would theoretically be able to adapt our circuit design to balance the number of iterations run against the circuit's noise resilience to identify an ideal circuit for NISQ-era hardware. One problem with using an MOEA to optimise Grover's algorithm is the structurally iterative nature of the circuit, which leaves little to be balanced in terms of circuit design, making the MOEA approach an over-complication.

QPE [21], is utilised to find an estimate of the phase ϕ of a quantum gate U when acting on an eigenstate $|\psi\rangle$ without altering the value of said eigenstate $|\psi\rangle$. To achieve this, we repeatedly apply a controlled-U (our quantum gate) controlled by an auxiliary qubit known as an "ancilla" qubit, after which this ancilla qubit is rotated by some specific angle βr which encodes information about the eigenvalue phases into the state of the ancilla qubit. We can then measure this ancilla qubit which will provide information regarding the eigenvalue phases. As in Grover's algorithm, the precision of

[8] All representable circuits

phase estimation in QPE increases with the number of controlled-U operations and therefore reduces noise resilience on NISQ-era hardware. Similarly to Grover's algorithm, QPE's iterative nature also means there is limited flexibility in circuit design, however, while the basic structure involves applying controlled gates and then performing a QFT, an MOEA could aim to optimise the efficiency of the quantum Fourier transform to balance circuit fidelity and noise-resilience.

Unlike both approaches outlined above, QFT as presented in *Quantum algorithms and the Fourier transform* [16] is structurally non-iterative and therefore poses a more appropriate opportunity for MOEA-based optimisation. QFT is a quantum implementation of the traditional Discrete Fourier Transform (DFT) and provides an exponential speedup of the algorithm with a time complexity of $O(n^2)$ vs DFTs $O(2^n)$. QFT is able to achieve this through the representation of all 2^n possible states of n qubits simultaneously through the principle of superposition, enabling the parallelised execution of QFT compared to the iterative DFT. This is done through a sequence of Hadamard and controlled-phase gates and has a more adjustable circuit structure regarding gate arrangements.

2.8 Critical Analysis

Having explored the various potential aspects challenging NISQ-era hardware, as well as approaches aiming to mitigate them, we can surmise a strong approach would be the use of a preference-based or hybrid MOEA with an interactive DM; preference-based as the goal is the identification of a single successful circuit, as opposed to the exploration of the Pareto-front, and an interactive DM for its ability to adjust our optimisation goal as we approach better circuit replication, adjusting towards circuit depth minimisation. Given our desire to optimise both circuit fidelity and circuit depth through our circuit's design, the QFT problem is the clear choice for MOEA implementation.

3 Project Plan

3.1 Hypothesis

The primary goal of this research is to utilise a new representation scheme combining aspects from both serial and block-based representations for use in a multi-objective evolutionary algorithm optimising circuit fidelity in NISQ-era hardware. We will specifically be aiming to improve the noise resilience of the Quantum Fourier Transform, to be achieved by balancing the accurate replication of the algorithm's output against improved noise resilience (driven by the minimisation of circuit depth). We hypothesise that this approach will produce a version of QFT that maximises overall performance on NISQ devices by reducing the impact of quantum noise and therefore improving circuit fidelity under current hardware constraints.

3.2 Project Requirements

Regarding hardware requirements, we will require access to a quantum simulation platform such as IBM Quantum Experience, Amazon Braket, or Qiskit's Aer simulator. Careful consideration must be taken when selecting the platform that best aligns with our project's goals, particularly in terms of the simulator's noise modelling capabilities and NISQ compatibility. Additionally, given the high computational demands of running an MOEA, we may require the use of optimised hardware for local execution, such as the use of a GPU to enable parallelised execution of our algorithm to reduce runtime.

From a software standpoint, requirements for this project include the programming language, key libraries for quantum simulation and MOEA implementation, and version control tools. For our programming language, we will primarily use Python due to the extensive libraries available as well as the ease of development. Although Python may result in slower execution times compared to compiled languages, this can be easily mitigated through the compilation of computationally expensive code sections. Given our language choice of Python, we will additionally be able to leverage many of the Quantum and MOEA libraries available to us.

3.3 Evaluation Criteria

One critical aspect of this project will be the evaluation of the designs created by our MOEA. To this end, we will want to compare our circuits with the traditional QFT as proposed in *Quantum algorithms and the Fourier transform* [16]. For this, we will have to run both circuits on a simulator able to replicate the effects of quantum noise as it affects NISQ-era hardware. Various existing toolkits such as Quiskit Aer [10] do exist that are able to provide inbuilt basic noise simulations. Using $MS\mathcal{F}$ (see **section 2.5**) we will be able to compare the fidelity of the two competing circuits under close to real-world noise conditions.

3.4 Risk Assessment

One possible challenge regards limitations in simulators and other libraries, such as restricted support for specific quantum gates or limitations in representing complex circuits, which may restrict the types of quantum circuits that can be simulated or limit the exploration of the solutions. This could necessitate the review of documentation and online sources for the mitigation of specific issues.

Another challenge could be limited access to high-performance hardware, such as GPUs which may be subject to availability constraints. Limited computing resources could slow down our MOEA algorithm or reduce the number of trials we can run, affecting the overall project timeline, however, if this were to occur, we might be able to schedule high-computation tasks to make efficient use of available university resources. Additionally, we could plan for alternative execution options, such as cloud-based computational resources if local hardware becomes unavailable.

Lastly regarding ethical risks, given that there are no users for this system, that it does not have any impact on the public, and that it has no direct use beyond possible optimisation of existing quantum systems, we can determine there are no ethical risks associated with this project.

3.5 Project Timeline

For this project, we will adopt a Waterfall development methodology [23], which breaks work into sequential, well-defined phases. Each phase is completed before advancing to the next, ensuring that base tasks are fully established before more complex tasks are undertaken. The Waterfall approach provides a structured, linear framework that supports clear planning and resource allocation at each stage. Given the complexity of implementing, and testing an MOEA for quantum circuit optimisation, the Waterfall model offers an appropriate development framework, and also allows us to identify points where we may shift focus to optional extension goals (labelled ”Push Goals” in the **Figure 2**) as well as the accounting for slippage time should certain phases take longer than expected (highlighted in lighter shades).

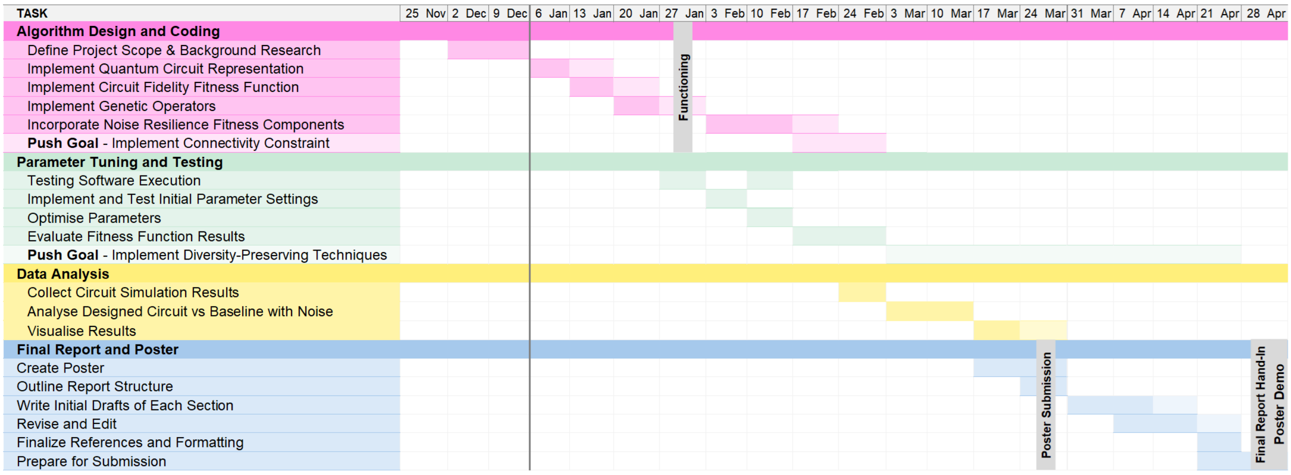

Figure 2: Project Timeline Gantt Chart

References

[1] Timothy Atkinson et al. "Quantum Program Synthesis: Swarm Algorithms and Benchmarks". In: *Genetic Programming: 22nd European Conference, EuroGP 2019, Held as Part of EvoStar 2019, Leipzig, Germany, April 24–26, 2019, Proceedings*. Springer-Verlag, 2019, pp. 19–34.

[2] Daniel Azses et al. "Navigating the noise-depth tradeoff in adiabatic quantum circuits". In: *Physical Review B* 107.12 (2023).

[3] Michel Boyer et al. "Tight bounds on quantum searching". In: *Protein Science* 46 (1996), pp. 493–505. URL: `https://api.semanticscholar.org/CorpusID:10032711`.

[4] Yudong Cao, Jhonathan Romero, and Alán Aspuru-Guzik. "Potential of quantum computing for drug discovery". In: *IBM J. Res. Dev.* 62 (2018), 6:1–6:20.

[5] Jerry M. Chow et al. "Universal quantum gate set approaching fault-tolerant thresholds with superconducting qubits." In: *Physical review letters* 109 6 (2012), p. 060501.

[6] Aashish A. Clerk et al. "Introduction to quantum noise, measurement, and amplification". In: *Reviews of Modern Physics* 82 (2008), pp. 1155–1208.

[7] Carlos A. Coello Coello, Arturo Hernández Aguirre, and Bill P. Buckles. "Evolutionary multiobjective design of combinational logic circuits". In: *Proceedings. The Second NASA/DoD Workshop on Evolvable Hardware* (2000), pp. 161–170.

[8] Kalyan Deb. "Multiobjective Optimization Using Evolutionary Algorithms. Wiley, New York". In: Jan. 2001.

[9] David Deutsch. "Quantum computational networks". In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 425 (1989), pp. 73–90.

[10] *Exact and noisy simulation with Qiskit Aer primitives*. `https://docs.quantum.ibm.com/guides/simulate-with-qiskit-aer`. Accessed: 2024-11-18.

[11] Jesús Guillermo Falcón-Cardona and Carlos A. Coello Coello. "Indicator-based Multi-objective Evolutionary Algorithms". In: *ACM Computing Surveys (CSUR)* 53 (2020), pp. 1–35.

[12] Lov K. Grover. "A fast quantum mechanical algorithm for database search". In: *Symposium on the Theory of Computing*. 1996. URL: `https://api.semanticscholar.org/CorpusID:207198067`.

[13] Stanley P. Gudder. "A Superposition Principle in Physics". In: *Journal of Mathematical Physics* 11 (1970), pp. 1037–1040.

[14] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[15] Jay Gambetta. *The hardware and software for the era of quantum utility is here*. `https://www.ibm.com/quantum/blog/quantum-roadmap-2033`. Accessed: 2024-19-10. 2023.

[16] Richard Jozsa. "Quantum algorithms and the Fourier transform". In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 454 (1997), pp. 323–337.

[17] Bikram Khanal et al. "Quantum Machine Learning: A Case Study of Grover's Algorithm". In: *2021 International Conference on Computational Science and Computational Intelligence (CSCI)* (2021), pp. 79–84.

[18] Martin Lukac and Marek A. Perkowski. "Evolving quantum circuits using genetic algorithm". In: *Proceedings 2002 NASA/DoD Conference on Evolvable Hardware* (2002), pp. 177–185.

[19] Kaisa Miettinen. *Nonlinear multiobjective optimization*. Vol. 12. Springer Science & Business Media, 1999.

[20] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Vol. 2. Cambridge university press Cambridge, 2001.

[21] Thomas E O'Brien, Brian Tarasinski, and Barbara M Terhal. "Quantum phase estimation of multiple eigenvalues for small-scale (noisy) experiments". In: *New Journal of Physics* 21.2 (2019), p. 023022.

[22] Román Orús, Samuel Mugel, and Enrique Lizaso. "Quantum computing for finance: Overview and prospects". In: *Reviews in Physics* (2018).

[23] Rupali Pravinkumar Pawar. "A Comparative study of Agile Software Development Methodology and traditional waterfall model". In: 2015.

[24] Massoud Pedram and A. Shafaei. "Layout Optimization for Quantum Circuits with Linear Nearest Neighbour Architectures". In: *IEEE Circuits and Systems Magazine* 16 (2016), pp. 62–74.

[25] John Preskill. "Quantum Computing in the NISQ era and beyond". In: *Quantum* (2018).

[26] Dayue Qin, Xiaosi Xu, and Ying Li. "An overview of quantum error mitigation formulas". In: *Chinese Physics B* (2022).

[27] Timothy G. Reid. "On the Evolutionary Design of Quantum Circuits". In: 2005.

[28] Benjamin I. P. Rubinstein. "Evolving quantum circuits using genetic programming". In: *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)* 1 (2001), 144–151 vol. 1.

[29] Moein Sarvaghad-Moghaddam. "Multi-objective design of quantum circuits using genetic programming". In: (2016). DOI: `10.48550/arXiv.1604.00642`.

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

[31] Kaitlin N. Smith et al. "Error Mitigation in Quantum Computers through Instruction Scheduling". In: 2021.

[32] Aimin Zhou et al. "Multiobjective evolutionary algorithms: A survey of the state of the art". In: *Swarm Evol. Comput.* 1 (2011), pp. 32–49.