Log Book

COMP8805/8800 Elly Sinden ers31@kent.ac.uk

- The github repository related to this project can be found here

Aims & Objectives - Overall Project

What are the learning objectives I intend to achieve throughout the project?

- Understand UML
- Understand the VQE Algorithm
- Have a good understanding of quantum computing
- Establish why Q-SE and quantum UML adaptations are important
- Establish if UML is a viable modelling tool.
- Assess the strengths and weaknesses of both quantum UML adaptations.

Were these learning objectives achieved?

- Mostly, I have a thorough understanding of the UML diagrams I've made and a good knowledge of VQE and quantum computing concerning how it is utilised to execute the VQE algorithm. The reasons why Q-SE is essential have been given by the authors of the literature that forms the basis of this project; my summarisation of this point is that quantum and classical information hold entirely different properties which will need to be distinguished when modelling larger systems that the one presented in this project, as I had modelled Qiskit code which ultimately is all transpiled to classical data when with the user. It will also be necessary to analyze the cost of larger systems. This is more related to modelling than Q-SE as a whole, simply as the project was a less extensive study on Q-SE implementations. Whether UML is a good tool also wasn't answered as they weren't compared with other models. I've gained insights from several peers who use it or do it for their work and information from forums like Reddit. Still, I need to find out how necessary UML is by opening the question to a larger pool of professionals. I established the strengths and weaknesses of both UML adaptions, not only with my observations but also by comparing them to characteristics of good modelling, which have been established and referred to in other modelling papers.

Were there any other successes?

- I learnt a lot about UML modelling tools and what works and doesn't, mainly using PUML and Mermaid, which is fine for sequence diagrams but not class diagrams in this context. This opens up a suitable candidate for developing a PUML Q-UML extension for future master's projects.

Project Timeline

- Make a note of Aims & Objectives for any milestones during the project

- All decisions should be justified
- Record any insights gained (things you don’t understand) or advice given by peers
- Record mistakes and how they were rectified

16/01/2024

I wanted to pursue a main project related to quantum computing due to an interest in the topic. QUML presented an opportunity to work within this domain without heavily relying on physics knowledge, which I do not possess a strong background in. I contacted Carlos and read the paper linked in the forum - Towards a Quantum Software Modeling Language

17/01/2024

I initially met with Carlos and was given additional reading: A Quantum Software Modeling Language. I understood that the project would be related to how quantum computers can be represented in UML. I believe UML is an achievable language to learn and base my main project on, and I would have the opportunity to better understand quantum computers. Wrote the following notes regarding "Quantum Software Engineering Modelling Tool":

- Good programming skills
- Strong mathematical skills
- Strong independent learning skills
- Ability to learn new material outside usual area of comfort
- Moreover, software developers face significant challenges when coding quantum programs due to switching to an entirely different programming mindset with counterintuitive quantum principles [14]. Quantum Software Engineering: A New Genre of Computing

18/01/2024

The early concept of the project was to implement a quantum extension to an already existing UML platform. I spoke with a friend who is a software engineer who recommended the open-source UML package Mermaid to create UML designs - mermaid Before using this package, I needed a better theoretical understanding of UML as a practice. I also wanted to investigate the “go-to” UML packages for software engineers or people who use UML regularly.

19/01/2024

Carlos accepted me onto the project with him as my supervisor. I registered for the quantum computing module COMP8220 and attended the first lecture (Introduction to Linear Algebra). I will voluntarily participate in the module to gain an overall understanding of quantum computing while simultaneously learning UML.

24/01/2024

I read the government's National Quantum Strategy NQSafter attending a talk in the physics department that mentioned it. I believe it may illuminate relevant UK bodies related to quantum computing and that I could access more resources to further my understanding of quantum. Goal 3 in NQS aligned with Carlos' aim in the QUML paper: adoption of Quantum technologies in the UK, the goal being to integrate quantum technologies with industry end users, those without a core understanding of quantum physics.

01/02/2024

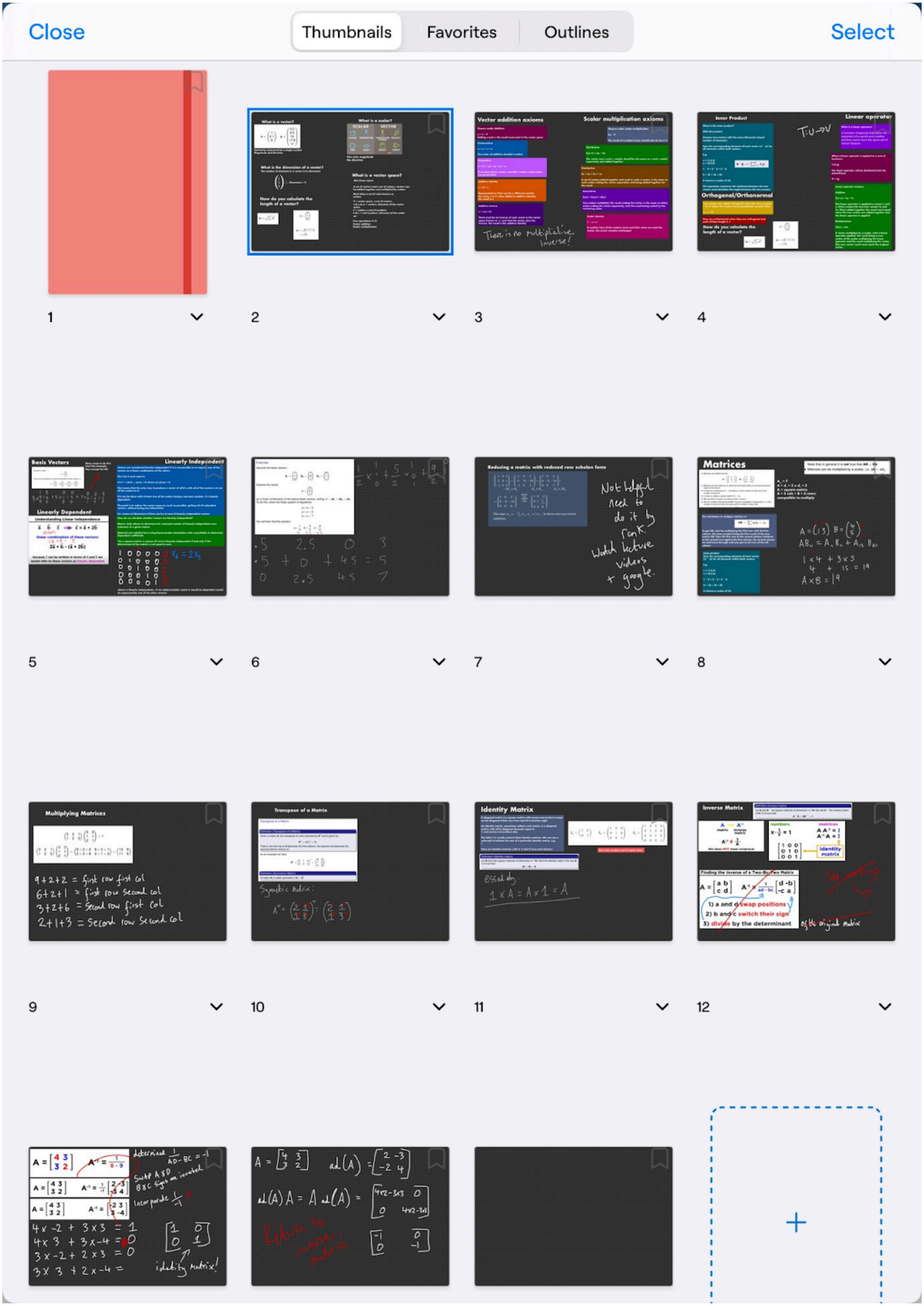
Researched links to UK Quantum organisations mentioned in NQS to source potential contacts or opportunities to practise using quantum software emulators. Focused on NQCC, so far, could only garner opportunities for paid courses but could do with further research.

07/02/2024

I am aware of IBM’s Qiskit, an open-source, Python-based quantum SDK. It has not yet been adequately investigated, but it is worth noting that it is possible to code in a quantum language and how this could be applied to UML.

11/02/2024

I started doing independent learning in linear algebra to build the understanding necessary for the first half of the COMP8220 module. However, it was proving difficult and taking time away from other tasks:

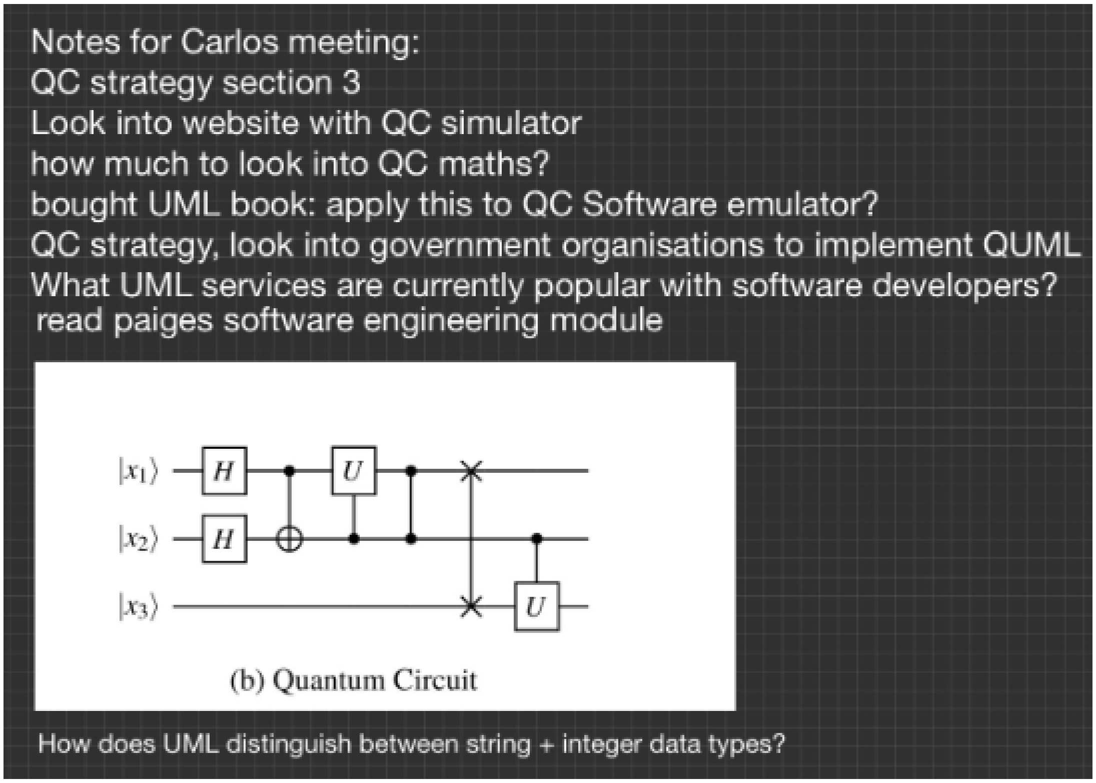

Close
Thumbnails
Favorites
Outlines
Select
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
+

13/02/2024

I started reading UML @ Classroom, which is referenced in QUML and recommended by Carlos as a good resource for understanding the basic concepts of UML.

14/02/2024

First group meeting with Carlos and other students working on quantum-related projects. Carlos mentioned his new paper Q-Cosmic, which I'll read after the meeting to discuss afterwards and give a broader understanding of quantum software engineering—we discussed what background knowledge we'd need for our projects, the first step, how to achieve it, and which technologies. The project proposal is due roughly 1st March and will be a concrete dissertation topic. We mentioned organising a regular reading group. What is a blockchain? Related to other peers' work



Notes for Carlos meeting:
QC strategy section 3
Look into website with QC simulator
how much to look into QC maths?
bought UML book: apply this to QC Software emulator?
QC strategy, look into government organisations to implement QUML
What UML services are currently popular with software developers?
read paiges software engineering module

(b) Quantum Circuit

How does UML distinguish between string + integer data types?

20/02/2024

I looked into the practicalities of UML and the people who use it. Feedback from Reddit states there are so many nuances to UML, and it is better to leave it ambiguous to avoid being incorrect, so they tend to employ a simplified version of UML. They would need a basic understanding of UML in the first place to do this. Anecdotal information from Reddit and friends who are software developers say they don't use UML. However, friends who are technical business analysts mostly use sequence diagrams and use case diagrams. Sometimes, class diagrams. They use UML frequently but on a case-by-case basis. They have not used a communication diagram. States that business analysts and solution architects use it the most.

Links:

WebSequenceDiagrams What is Communication Diagram? C4 model UML Design Resources How often do you use UML? Miro Unified Modeling Language (UML)

Following this research, I’ve decided to expand on the work in the QUML paper and create an example of one of the other UML diagrams not included in the paper and how it can be used for quantum computing. I decided to leave investigating how to create a quantum extension for a UML package. I was now unsure how popular it would be if software developers said they don't use UML packages but instead do their diagrams based on UML diagrams. By creating an example of other UML diagrams for quantum, it could be adopted by different professionals either explicitly using UML or their variation of UML. I want to start by creating a communication diagram as it’s related to sequence diagrams and can drill down into the communication between different quantum/classical elements.

21/02/2024

Second group meeting. To look into Grover and Shor algorithms, understanding quantum algorithms will aid in fully comprehending the UML examples provided in the QUML paper and creating other UML diagrams. We discussed Q-Cosmic; it's the beginning of thinking about how to quantify the size of the software before its existence to charge it in the future. Q-Cosmic: Hybrid Communication. Carlos suggested that to investigate communication diagrams further, look into Quantum State Preparation when moving from classical to Quantum and Quantum Measurement when moving from Quantum to classical.

12/03/2024

I started looking into quantum state preparation; it comes in many forms, as it's a technique to modify the quantum system to observe a quantum state. I found the following information related to the topic:

Quantum state preparation of normal distributions using matrix product states Preparation of Entangled States by Quantum Markov Processes On the reality of the quantum state What is a quantum "state preparation procedure" and what isn't What exactly does state preparation mean in quantum computing?

After this research, I understand that it is the process of placing a quantum system into a specific state. Manipulating the system can be achieved through several means, including using quantum gates, quantum algorithms, applying electromagnetic fields, and controlling particle interactions.

I found the subject matter a little overwhelming, particularly regarding what route I would need to take to show the communication between classical and quantum computers effectively. I believe the next step is to go back to the basics of understanding the fundamentals of the quantum gates and read the relevant literature.

I have also attended Carlo's COMP8220 lectures and learnt the Deutsch algorithm. I need to take the same approach as learning classical computers and understand the difference between low-level (gates) and high-level (algorithms/code using an HLL) required in the communication diagram.

I still need to read up on quantum measurement. However, from lectures and previous reading into quantum computing, I can understand why this works: By measuring a qubit, it collapses the quantum state and will be fixed at a 0 or a 1.

I need to review linear algebra, as I’m still not confident about it, to better understand quantum gates. It would also help me grasp the algorithms better. I can continue to learn standard UML communication diagrams, but it will be easier to hold the QC aspect when understanding the fundamentals.

I researched the low-level aspects that I didn’t fully understand before, particularly notations such as why the square root of ½ is used.

Links:

Quantum Pies, Waves, and Grandma’s Kitchen: A Deep Dive into 1/sqrt(2) Why do I see 12√12 a lot in QM?

To summarise what I’ve learned, in quantum mechanics, probabilities are represented as probability amplitudes, which are complex numbers due to the wave-like nature of quantum particles. Magnitude squared by the probability amplitude gives the likelihood of a particular outcome (quantum state). ½ represents the probability, and the square root ½ represents the probability amplitude. Probability amplitudes can be negative or imaginary which reflects superposition principles.

I’ve also made a note to look into Euler's formula.

13/03/2024

Third group meeting. I discussed my concerns with Carlos, who advised me to avoid getting bogged down in the details. I don’t need to understand the fundamentals of linear algebra and everything involved with quantum to complete the project; this would be impossible anyway. I left the meeting with some “breathing space” to collect myself and work out the best first steps.

We discussed the properties of a good research paper when writing our own. The requirements are:

- abstract
- intro
- conclusion
- good title

It needs to leave the reader wanting more, each sentence adding further value, and avoid overselling what's written (good abstract, bad content). Original content should be 60%/70% with background information being 40%/30%.

I’ve opted to base the critical review on Carlos’ QUML paper as this is the basis for my project. We discussed what makes a good critical review:

- walk away understanding the reviewed paper
- understand what the results of the reviewed paper are about
- The reader doesn't need to read the reviewed paper; they can go off yours, and yours should be “better.”
- provide additional contributions to the reviewed paper
- contribute something meaningful
- something no one has done before; useful, novel and with value
- results, strengths and weaknesses

Carlos mentioned that literature reviews are valid scientific output. They typically condense 100 papers into 1, so you don't have to read 100! He also advised us to spend half of our time on the main project and the other half on the modules due to their significance to our final marks.

I’ve also made a note to look into Quantum Polarising Filters.

Reading recommendations:

- Systemisation of Knowledge (SoK) papers
- The New Scientist
- Quantum Computing for Computer Scientists by Mirco A. Mannucci and Noson S. Yanofsky

- Quantum Computation and Quantum Information by Michael A. Nielsen and Isaac L. Chuang

03/04/2024

Group meeting. We discussed the benefits of using Overleaf for the critical review and dissertation due to its professional formatting and ability to utilise and create a bibliography of Bibtex files for referencing. I need to confirm whether the dissertation and corpus are one unified document or separate. It would be helpful to obtain books on scientific writing/academic writing to understand better how to write a good dissertation. Carlos has recommended keeping it clear and concise as a general rule.

Carlos advised me to use an assertive voice instead of a passive voice. I should not use the royal “we” or “I” but include the reader. For example, “We will study…” So we as in myself and the reader, not the royal we.

Regarding critical review references, it would be helpful to compare other papers related to the subject matter during the evaluation that may have achieved what the reviewed paper didn’t. However, this is not essential.

04/04/2024

First project methods workshop. Notes were taken to consider as the project is research-based as opposed to developing a piece of software:

- Research; not known or new perspective
- Must be able to replicate the results you’ve discovered
- Fit your work in the context of what others have done
- Challenge well-established existing ideas
- The right question, not the right answer (this has also been discussed in our group meetings)
- Find ways your finding does not work to further discovery and improvements
- Utilise DBLP for sourcing research papers

Could QUML only apply in a particular context? Is it restricted in its design? My research:

- Define specific research questions with Carlos.
- How does it interact with the broader field of knowledge?
- Ask Carlos about prominent publications on quantum computing.

The organisation of research:

- Set targets
- Identify milestones

Don’t write your dissertation like a chronological logbook. Report in the order that makes the most sense to the user:

- Introduce the concepts and base information that’s required to understand the paper
- Establish what the base knowledge of the reader is
- Write so it gives the reader base knowledge of the topic in the introduction

05/04/2024

I spoke with Carlos during the final lecture after having a “lightbulb” moment regarding UML communication diagrams and quantum. I asked whether a good question to ask is how UML communication diagrams can be used to illustrate the communication between classical and quantum computers, seeing as classical computers are needed to interact with quantum computers at present. Carlos advised that it was a worthwhile question to ask.

18/04/2024

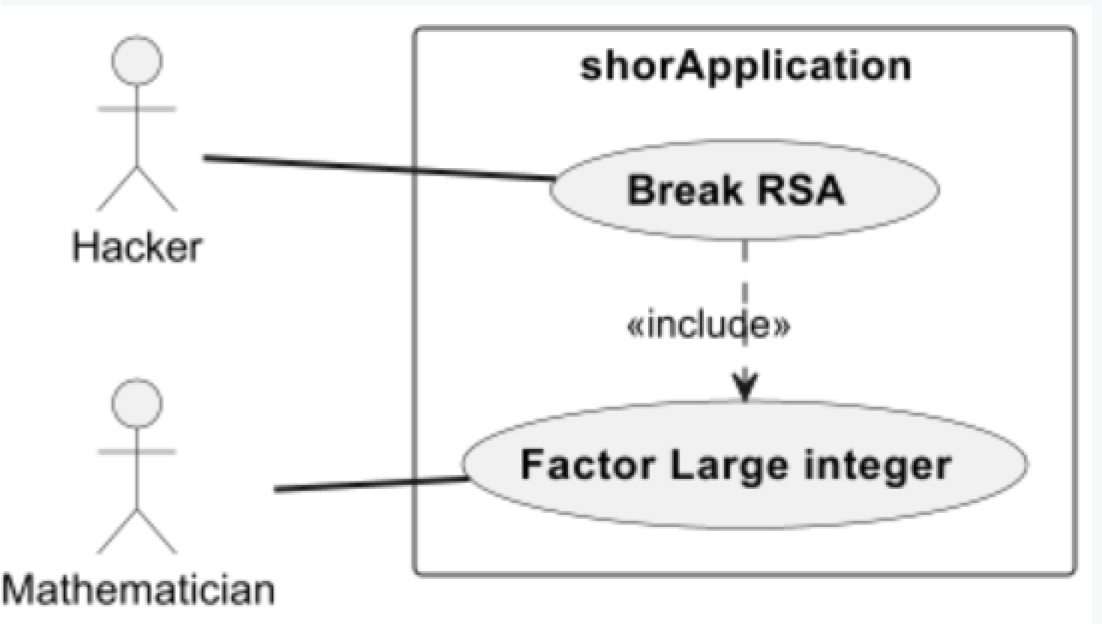
I began preparing for the critical review. I referred to the critical review lecture slides, specification document and Fong’s paper on structuring a critical review. In preparation for writing the review, I utilised the following GitHub repository sort-google-scholar and searched papers under the term “Quantum UML”. It brought up the following papers. I read the latter two in full and about ten pages of the former:

Quantum Software Engineering Modelling Quantum Circuits with UML Design of classical-quantum systems with UML

The Quantum Software Engineering (QSE) paper is a fantastic resource that gives an overview of all aspects of the field and expands my understanding beyond just QUML and how it fits into the grand scheme of things relating to the domain of QSE. The other two papers (one being an extension of work to the other) give a different perspective on how to apply quantum to UML through UML profiles. This is a resource I can return to analyse more in-depth when applying quantum concepts to UML diagrams.

26/04/2024

Completed and submitted the critical review. I was able to lay a timeline of where QUML fits into the grand scheme of things in the development of QSE:



I now better understand how new QSE and QUML fit into the grand scheme of things. Combined with the research papers relating to QSE, it feels more manageable, as there are many different applications within this, such as quantum programming languages, with the QUML principles and axioms that can also be applied. I don’t need to focus on the minute details, such as linear algebra, to understand how software engineering principles can be applied to quantum computing.

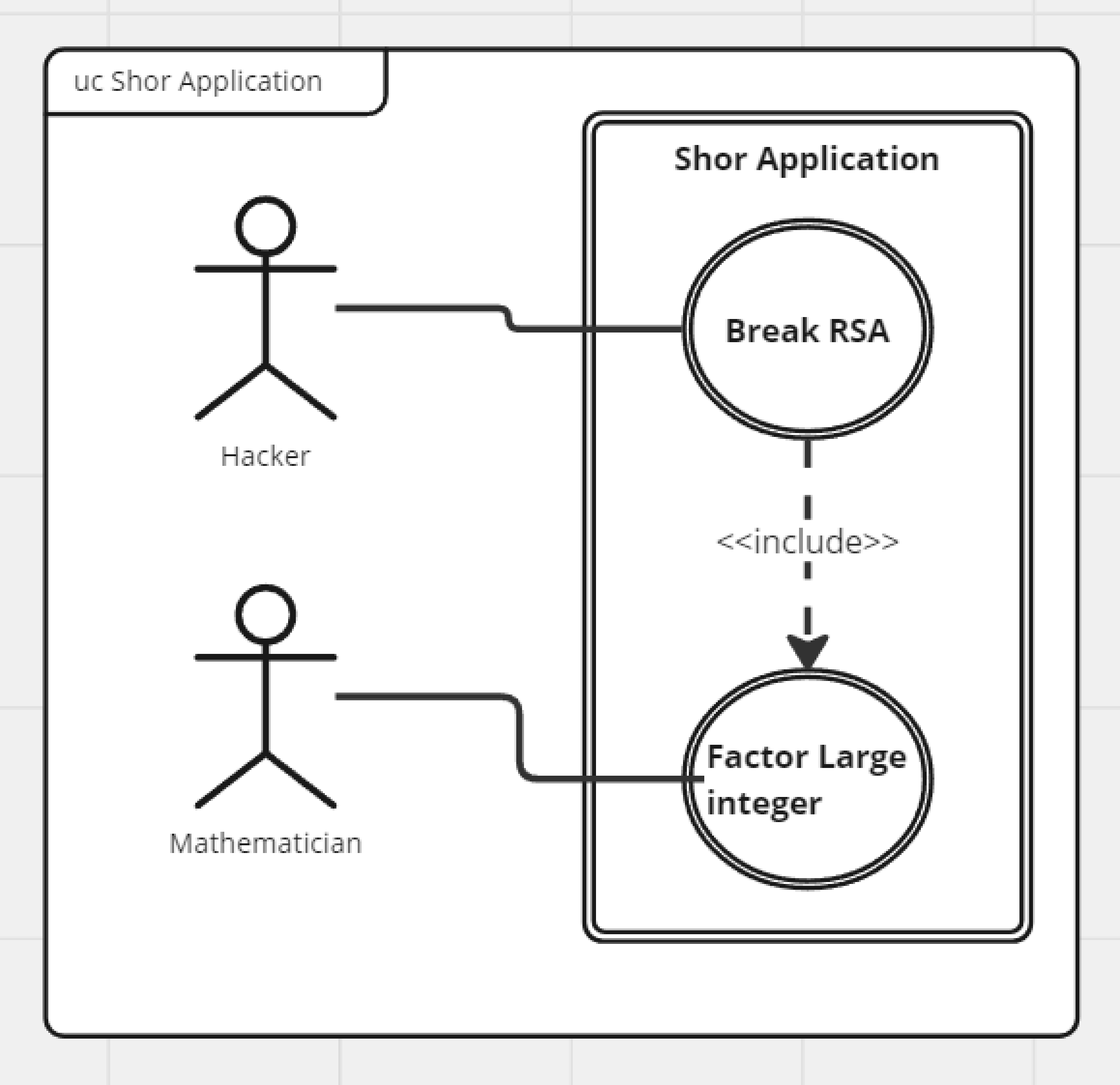
After completing the critical review, I can implement the design choices of bold text for textual information and double-line pictorial information and how to apply this to a new UML design. My next step in constructing UML designs is to recreate the examples in Carlos' paper. This should give me an understanding of both Shor's algorithm and how the examples utilise the rules of UML and their specific diagrams. I need to move away from the theoretical and apply the practical to solidify my understanding.

02/05/2024

I looked into recreating the use case diagram in the QUML paper. A software engineer I know had previously sent me information on PlantUML) (PUML) as an open-source Java library where you create UML diagrams through code. I was drawn to using this as I would enjoy the ability to implement UML through code and work on some practical coding work for the project.

I had also looked into Miro; however, the number of designs that could be created was limited.

I began learning how to construct a use case diagram in PUML and copying the example from the QUML paper. I used the inbuilt Creole markup to implement the bold text. However, there was no inbuilt functionality to create double dashed lines, which is necessary to indicate the quantum <> relationship between the two use cases and for the use cases themselves:



I would like to access the PUML GitHub repository and make my adjustments to include this functionality. Due to its open-source nature, PUML could be a good option for creating a quantum extension for UML. However, as this process is to solidify my understanding of both quantum algorithms and UML diagram structures through practical experience, I believe it would be better to spend my time finding a UML package where I can include these design functions immediately, as it would take me time to learn how to modify PUML.

I investigated recommended applications and got mixed reviews for PUML. Some people enjoy using it, while others say it does not scale well as diagrams become more complex. There were also recommendations to use Mermaid (as previously suggested) as PUML is quite old now, and Mermaid is also open-source. However, as it's older, could it also mean it's more popular/has greater support for modifications?

Links:

Is there any open source software for doing UML diagrams? PlantUML turns text into diagrams Hacker News. Plant UML in VS 2022 community edition and push to Github?

I looked into the applications of PUML for communication diagrams via the forum and was linked to the following type of diagram:

Component Diagram

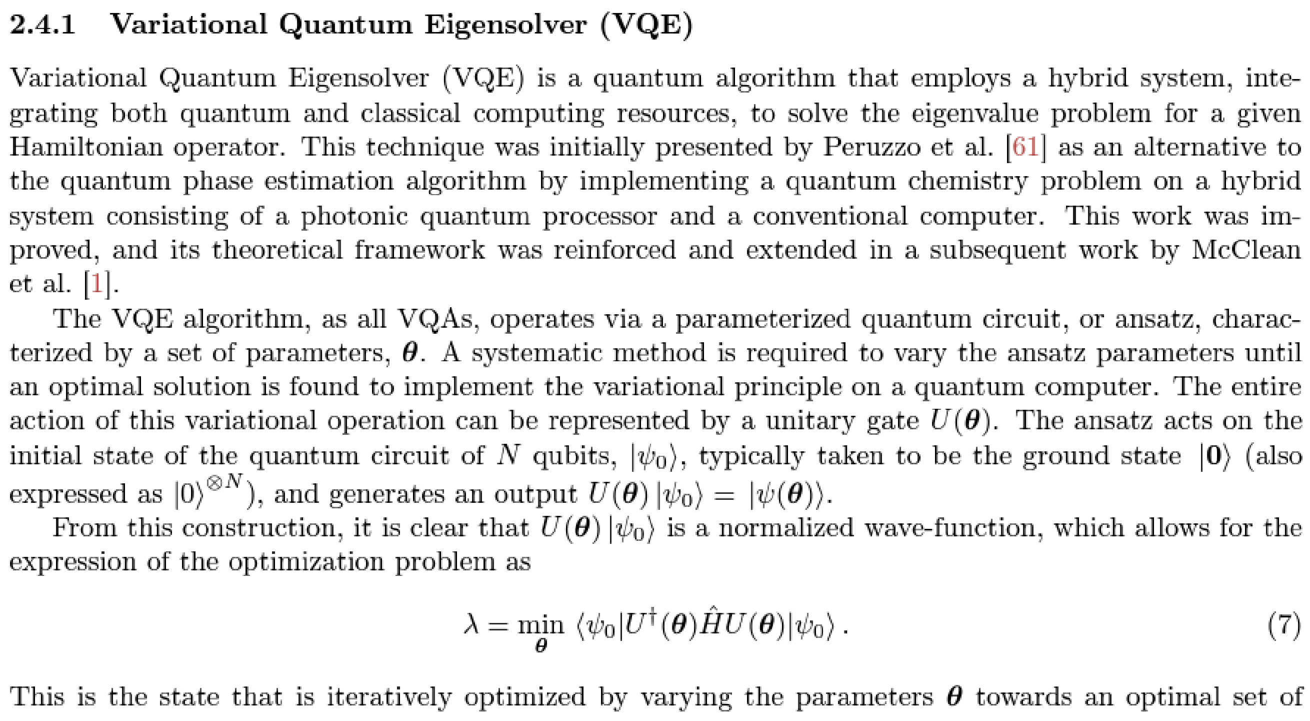
I will return to this task and Miro (even if I have to pay for a subscription) as this seems a popular choice with software engineers already and hopefully has the versatility to incorporate the QUML design choices. Once I can fully replicate them, I will provide textual explanations of what they represent in the context of Shor's algorithm and the specific UML diagram.

I also spent the day transferring diary notes into a Logbook following recommendations from Les Johnson

28/05/2024

I returned to the project after finishing exams. I used Miro to recreate the use case diagram in Carlos' paper. During this process, I created a Word document, "Analysing QUML Diagrams", to break down the UML aspects and Q-UML; my thought process is to apply this to all the diagrams in the paper to get a sound understanding of how to use UML in this process, I'll then move on to learning how classical systems interact with quantum systems and can create my own QUML diagrams of these processes.

I encountered the same roadblock as before: there isn't a straightforward answer to creating double-lined connections. I emailed Carlos to ask how he made the diagrams for his paper. However, this shouldn't hinder me from analysing the diagrams in the paper. I've completed the use case diagram and will now move on to the class diagram.



2.4.1 Variational Quantum Eigensolver (VQE)

Variational Quantum Eigensolver (VQE) is a quantum algorithm that employs a hybrid system, integrating both quantum and classical computing resources, to solve the eigenvalue problem for a given Hamiltonian operator. This technique was initially presented by Peruzzo et al. [61] as an alternative to the quantum phase estimation algorithm by implementing a quantum chemistry problem on a hybrid system consisting of a photonic quantum processor and a conventional computer. This work was improved, and its theoretical framework was reinforced and extended in a subsequent work by McClean et al. [1].

The VQE algorithm, as all VQAs, operates via a parameterized quantum circuit, or ansatz, characterized by a set of parameters, $\boldsymbol{\theta}$. A systematic method is required to vary the ansatz parameters until an optimal solution is found to implement the variational principle on a quantum computer. The entire action of this variational operation can be represented by a unitary gate $U(\boldsymbol{\theta})$. The ansatz acts on the initial state of the quantum circuit of N qubits, $|\psi_0\rangle$, typically taken to be the ground state $|\mathbf{0}\rangle$ (also expressed as $|0\rangle^{\otimes N}$), and generates an output $U(\boldsymbol{\theta})\,|\psi_0\rangle = |\psi(\boldsymbol{\theta})\rangle$.

From this construction, it is clear that $U(\boldsymbol{\theta})\,|\psi_0\rangle$ is a normalized wave-function, which allows for the expression of the optimization problem as

$$\lambda = \min_{\boldsymbol{\theta}} \langle \psi_0 | U^\dagger(\boldsymbol{\theta}) \hat{H} U(\boldsymbol{\theta}) | \psi_0 \rangle \,. \tag{7}$$

This is the state that is iteratively optimized by varying the parameters $\boldsymbol{\theta}$ towards an optimal set of

An issue noted when going through class diagrams is that the book UML @ Classroom states that class names are centred and written in bold font. In QUML, quantum class names are bold, whereas classical names are not bold. This is a change to the standardisation of UML, so it’s worth investigating what the common practice is and whether a design change needs to be made for QUML, for example, using underlining rather than bold to denote quantum.

A minor note is that “m” was used instead of “*” to denote multiple in the aggregation between ShorApplication and ShorFactor.

29/05/2024

Continuing with class diagram analysis. I finished running through the diagram and understanding the UML notations. I did not have a complete understanding of how this example represents Shor’s algorithm in code, so I looked at the following videos to piece together my understanding:

Shor's Algorithm — Programming on Quantum Computers — Coding with Qiskit S2E7 The Story of Shor's Algorithm, Straight From the Source | Peter Shor

It would be nice to have, but not a must-have, to understand this and fully summarise the diagram's operation. However, I will only spend a little time on this and continue with the following UML diagram. As my work will be based not on Shor’s algorithm but on the communication between classical and quantum computers, it’s not essential to understand Shor’s algorithm for now; this process is more to understand the UML notations through deciphering working examples.

I also re-watched the following to refresh my understanding of Shor’s algorithm - How Quantum Computers Break Encryption | Shor's Algorithm Explained

30/05/2024

I started the morning by quickly analysing the state diagram in QUML. This one is relatively straightforward (negating the fact I haven't delved into the actual context of what the diagram represents, i.e. how Quantum Fourier Transforms work); however, it is worth bearing in mind when investigating how classical computers interact with quantum as there will be a state change when moving from classical to quantum information and visa versa. The initial plan had been to focus on communication diagrams and possibly others not depicted in the QUML paper; however, once I conduct my research into classical/quantum interaction, it should become more apparent which UML diagrams are required. So, rather than extending the Shor algorithm example and applying it to other UML diagrams to transform to QUML, I would consider another aspect of quantum software (classical/quantum interaction) and how this can be depicted through design.

Before moving on to Activity Diagrams, I wanted a better understanding of how we interact with Quantum Computers, so I looked at the following:

Decoded: How Does a Quantum Computer Work? Quantum computing: Facts, fiction and the future

This led me to look further into Sycamore and Google Quantum AIand, following the roadmap, looked further into NISQ - Noisy intermediate-scale quantum era and Beyond quantum supremacy: the hunt for useful quantum computers What is NISQ

I intend to use UML to model existing quantum architecture instead of a theoretical QC software algorithm.

A potential topic to investigate could be error correction algorithms and quantum-based simulators.

I finished my reading session today and am interested in further exploring classical/quantum hybrid algorithms such as VQE, QAOA, and VQF. Tomorrow, I will investigate QAOA to see if it can be a good focus point for understanding a quantum/classical hybrid algorithm and how this can be illustrated via QUML.

I will return to the topic tomorrow and watch this:

Lecture 5.2 - Introduction to the Quantum Approximate Optimization Algorithm and Applications

Also, it would be worth looking into Qiksit and Cirq.

04/06/2024

I spent this period researching the QAOA algorithm. Some general links during these investigations were:

Solve utility-scale quantum optimization problems Farhi, J Goldstone, S Gutmann - A quantum approximate optimisation algorithm Quantum Machine Learning - 18 - Quantum Approximate Optimization Algorithm (QAOA) What is a Hamiltonian? Quantum Jargon Explained - A video understanding Hamiltonians. This was necessary as Hamiltonians represent a system, and the algorithm seeks to find the ground state of a system. A tutorial on Quantum Approximate Optimization Algorithm (Oct 2020). Part 1: Theory - This was the critical video used to attempt to understand the algorithm, with a link to a related GitHub repository: QAOA_tutorial

10/06/2024

I returned to researching the QAOA algorithm after having a week off. Relevant links include: Quantum optimization algorithms BQP - roughly the equivalent of classical P class (stated in YT video), determining the hardness of algorithms. Hermitian matrix Quantum vacuum state Maximum cut Combinatorial optimization Optimization problem A tutorial on Quantum Approximate Optimization Algorithm (Oct 2020). Part 2: Hands-on

I started writing informal notes on QAOA in a separate Word document to formulate better and understand the algorithm in my head.

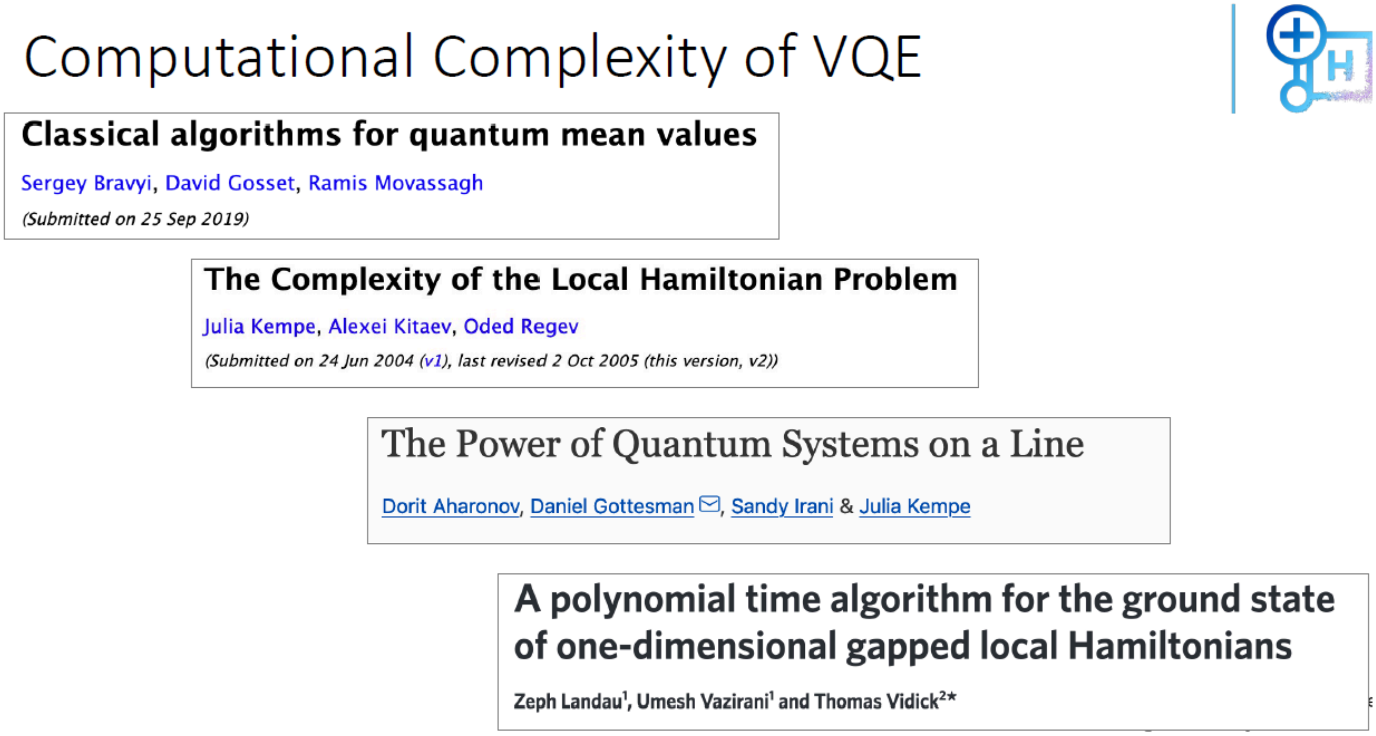
One aspect that could be represented in QUML is translating the classical problem formula to a Hamiltonian space.

Links to Pennylane, an open-source framework for Quantum ML, has videos and tutorials on writing the QAOA algorithm in PennyLane: QAOA: A different perspective | PennyLane Tutorial Intro to QAOA

To do: Look into Fourier expansion.

11/06/2024

Whilst reading a paper on QAOA (A Review on Quantum Approximate Optimization Algorithm and its Variants), I came across a section which discusses the VQE algorithm:



Computational Complexity of VQE

Classical algorithms for quantum mean values

Sergey Bravyi, David Gosset, Ramis Movassagh

(Submitted on 25 Sep 2019)

The Complexity of the Local Hamiltonian Problem

Julia Kempe, Alexei Kitaev, Oded Regev

(Submitted on 24 Jun 2004 (v1), last revised 2 Oct 2005 (this version, v2))

The Power of Quantum Systems on a Line

Dorit Aharonov, Daniel Gottesman, Sandy Irani & Julia Kempe

A polynomial time algorithm for the ground state of one-dimensional gapped local Hamiltonians

Zeph Landau[1], Umesh Vazirani[1] and Thomas Vidick[2*]

I was particularly drawn to implementing VQE on a hybrid classical/quantum system. My initial goal had been to source information on classical/quantum systems interacting with each other to represent via a UML communication diagram.

I started to look into VQE using the following links:

A variational eigenvalue solver on a photonic quantum processor - This is the original paper, which has information on the hardware used in the experiment, which could be used for a UML diagram

The theory of variational hybrid quantum-classical algorithms

Variational Quantum Eigensolver | Qiskit Global Summer School 2023—This video was the primary source for deepening my understanding of VQE. It also mentions how QAOA and VQE are essentially the same algorithms from different fields. QAOA is frequently demonstrated with the Maxcut problem; they both seek to find a global minimum, the system's ground state. The link to the related GitHub is qgss-2023.

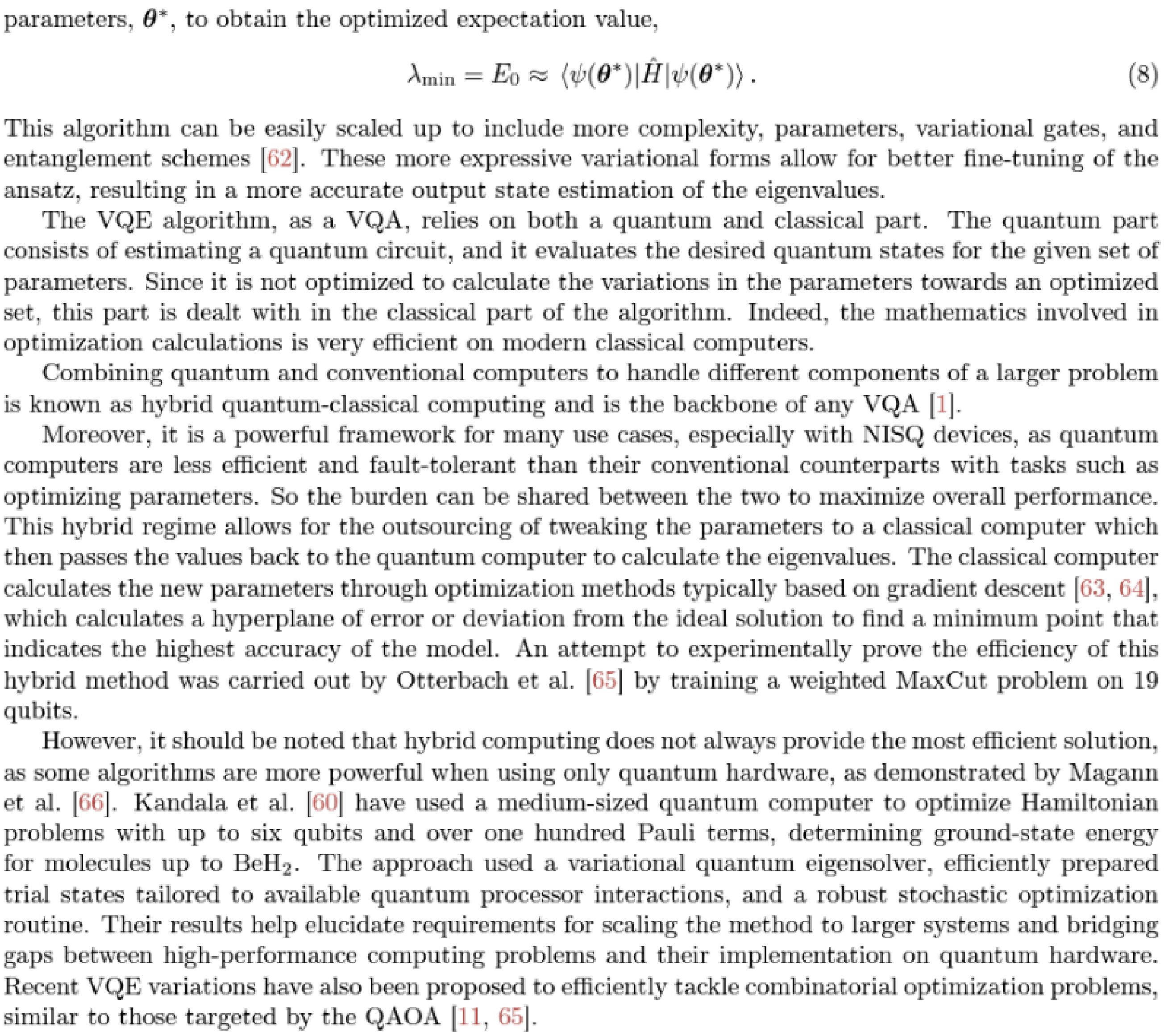
I decided to switch from QAOA to VQE as there would be no loss in prior work understanding QAOA. They are essentially the same algorithm as VQE, which has further information on the classical/quantum hardware and an IBM tutorial on how to code it in Qiskit. Although I could have used Pennylane, coding the algorithm to further my understanding and having some experience writing in Qiskit would be beneficial. Also, whereas QAOA is typically demonstrated via the Maxcut problem, VQE has the potential of being simpler, whereby tutorials just discuss how it seeks the ground state as opposed to applying it to a problem.

I also looked into different software for UML diagrams, which would allow me to implement the graphical choices of Q-UML. The links were:

drawio github drawio

I received a reply from Carlos regarding what he used in the Q-UML paper: Lucidchart.

Links presented in the IBM YT VQE video which could be relevant:



parameters, $\boldsymbol{\theta}^*$, to obtain the optimized expectation value,

$$\lambda_{\min} = E_0 \approx \langle\psi(\boldsymbol{\theta}^*)|\hat{H}|\psi(\boldsymbol{\theta}^*)\rangle \,. \tag{8}$$

This algorithm can be easily scaled up to include more complexity, parameters, variational gates, and entanglement schemes [62]. These more expressive variational forms allow for better fine-tuning of the ansatz, resulting in a more accurate output state estimation of the eigenvalues.

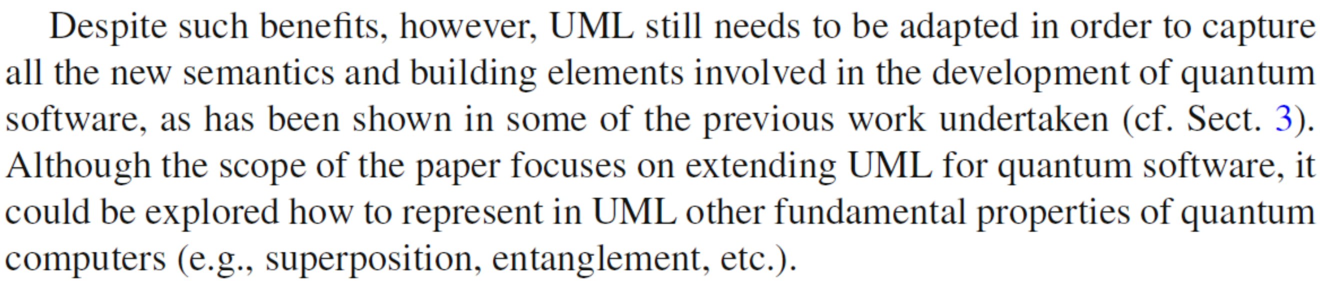
The VQE algorithm, as a VQA, relies on both a quantum and classical part. The quantum part consists of estimating a quantum circuit, and it evaluates the desired quantum states for the given set of parameters. Since it is not optimized to calculate the variations in the parameters towards an optimized set, this part is dealt with in the classical part of the algorithm. Indeed, the mathematics involved in optimization calculations is very efficient on modern classical computers.

Combining quantum and conventional computers to handle different components of a larger problem is known as hybrid quantum-classical computing and is the backbone of any VQA [1].

Moreover, it is a powerful framework for many use cases, especially with NISQ devices, as quantum computers are less efficient and fault-tolerant than their conventional counterparts with tasks such as optimizing parameters. So the burden can be shared between the two to maximize overall performance. This hybrid regime allows for the outsourcing of tweaking the parameters to a classical computer which then passes the values back to the quantum computer to calculate the eigenvalues. The classical computer calculates the new parameters through optimization methods typically based on gradient descent [63, 64], which calculates a hyperplane of error or deviation from the ideal solution to find a minimum point that indicates the highest accuracy of the model. An attempt to experimentally prove the efficiency of this hybrid method was carried out by Otterbach et al. [65] by training a weighted MaxCut problem on 19 qubits.

However, it should be noted that hybrid computing does not always provide the most efficient solution, as some algorithms are more powerful when using only quantum hardware, as demonstrated by Magann et al. [66]. Kandala et al. [60] have used a medium-sized quantum computer to optimize Hamiltonian problems with up to six qubits and over one hundred Pauli terms, determining ground-state energy for molecules up to BeH_2. The approach used a variational quantum eigensolver, efficiently prepared trial states tailored to available quantum processor interactions, and a robust stochastic optimization routine. Their results help elucidate requirements for scaling the method to larger systems and bridging gaps between high-performance computing problems and their implementation on quantum hardware. Recent VQE variations have also been proposed to efficiently tackle combinatorial optimization problems, similar to those targeted by the QAOA [11, 65].

Extract from the A Review on Quantum Approximate Optimization Algorithm and its Variants paper, going into detail on hybrid quantum/classical systems:



Despite such benefits, however, UML still needs to be adapted in order to capture all the new semantics and building elements involved in the development of quantum software, as has been shown in some of the previous work undertaken (cf. Sect. 3). Although the scope of the paper focuses on extending UML for quantum software, it could be explored how to represent in UML other fundamental properties of quantum computers (e.g., superposition, entanglement, etc.).

11/06/2024

I attended workshops during this period.

After deciding on VQE as the algorithm to base my UML designs, I returned to the following paper (Modelling quantum circuits with UML), which was included in my critical review. This was another approach to adapting UML for quantum systems, and I wanted to re-read it to understand better what it does differently to Carlos. Essentially, they use functionality in UML diagrams called profile diagrams, which can be adapted for niche fields, such as adapting quantum technologies.

The authors followed up this paper (Design of classical-quantum systems with UML), which I will look into after.

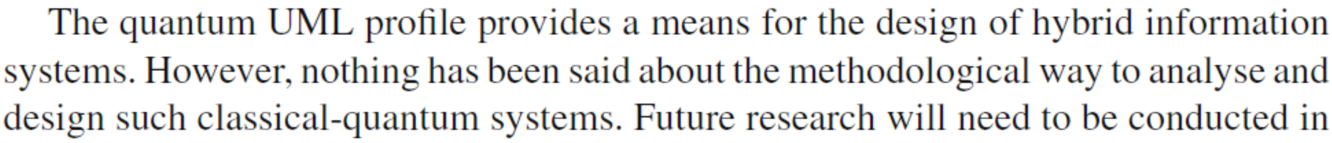
Links from today:

Profile_(UML) UML Profile What is Profile Diagram in UML?

My current thought process focuses on communication diagrams representing classical and quantum interaction. I could apply different UML adaptations to this interaction, such as QUML, UML quantum profile diagram, and potentially a third one I constructed myself. I want to represent communication, the interaction between the physical systems, and a class diagram for the VQE algorithm. Another thought is whether to speak to Rogério about the best way to measure the effectiveness of these diagrams within the software engineering community.

I started watching lecture recordings on modelling from software engineering. I am stuck on how to proceed, so I will email Carlos.

Extract from the Design of classical-quantum systems with UML refers to it being a good idea to explore UML in its representation of fundamental QC properties:



The quantum UML profile provides a means for the design of hybrid information systems. However, nothing has been said about the methodological way to analyse and design such classical-quantum systems. Future research will need to be conducted in

12/06/2024

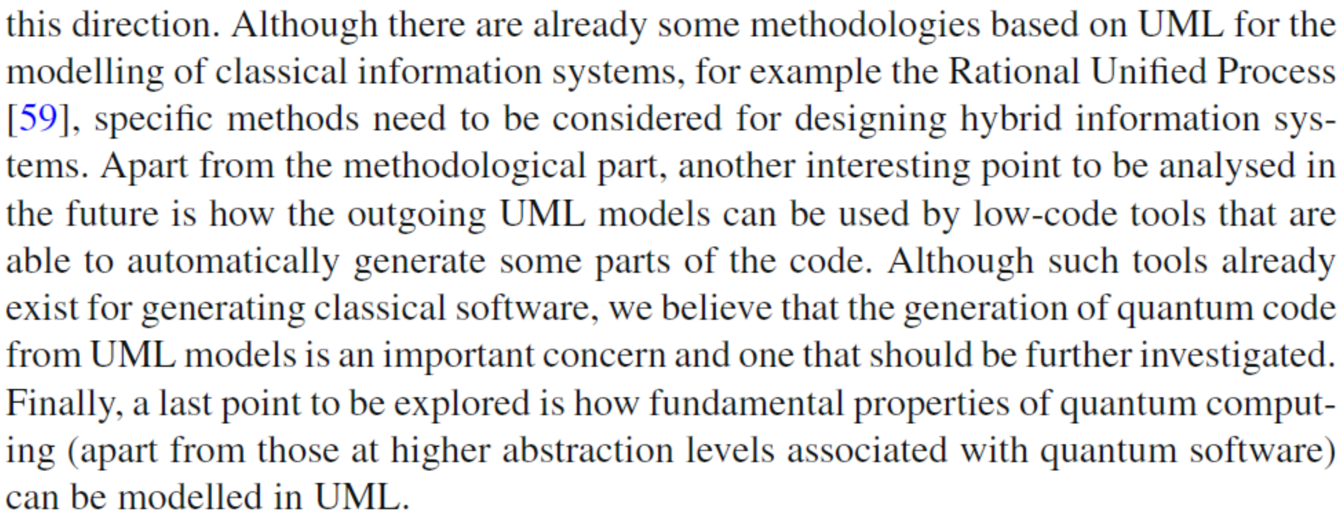
Wrote the following notes on Variational Quantum Eigensolver:

- A near term similar to QAOA, algorithms that are able to run on current NISQ systems with a hybrid design where certain parts of the algorithm are dedicated to either quantum or classical systems.
- Designed to take advantage of shallow quantum circuit and noise level. - yt video
- Wiki Link

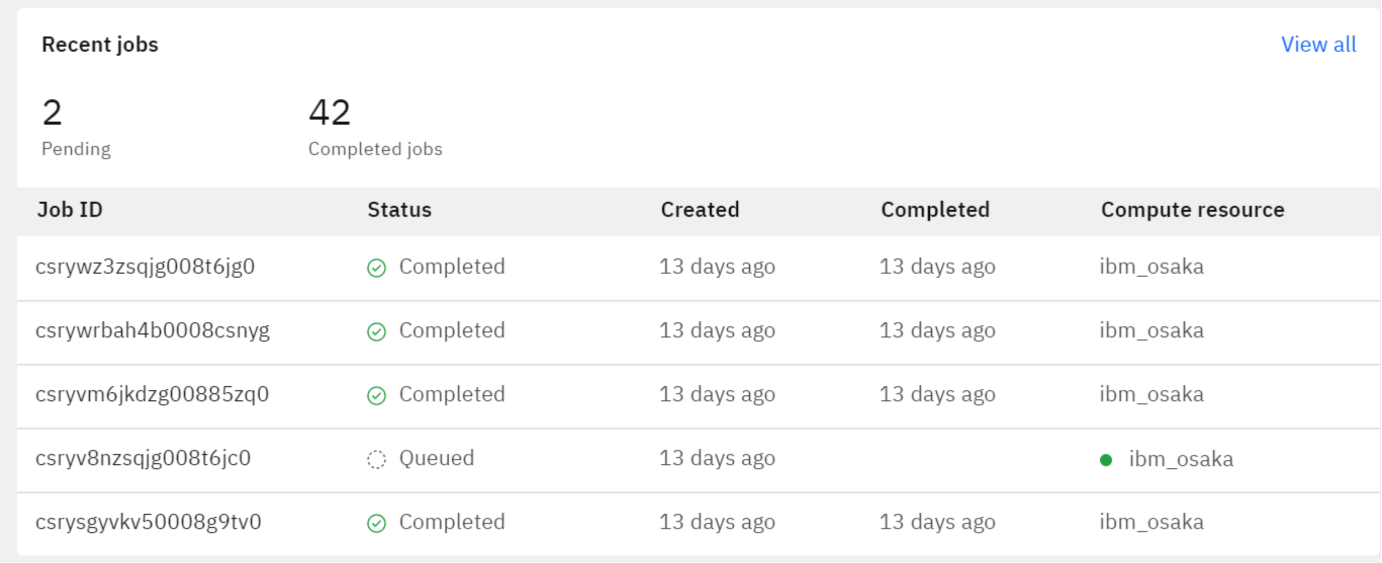
17/06/2024

I finished reading Design of classical-quantum Systems with UML. Links from today:
UML Profile Diagram

The conclusion of the paper has some good jumping-off points in directions of where the quantum UML profile could be applied next, which is worth investigating:



this direction. Although there are already some methodologies based on UML for the modelling of classical information systems, for example the Rational Unified Process [59], specific methods need to be considered for designing hybrid information systems. Apart from the methodological part, another interesting point to be analysed in the future is how the outgoing UML models can be used by low-code tools that are able to automatically generate some parts of the code. Although such tools already exist for generating classical software, we believe that the generation of quantum code from UML models is an important concern and one that should be further investigated. Finally, a last point to be explored is how fundamental properties of quantum computing (apart from those at higher abstraction levels associated with quantum software) can be modelled in UML.



Recent jobs

View all

2
Pending

42
Completed jobs

Job ID	Status	Created	Completed	Compute resource
csrywz3zsqjg008t6jg0	✓ Completed	13 days ago	13 days ago	ibm_osaka
csrywrbah4b0008csnyg	✓ Completed	13 days ago	13 days ago	ibm_osaka
csryvm6jkdzg00885zq0	✓ Completed	13 days ago	13 days ago	ibm_osaka
csryv8nzsqjg008t6jc0	◌ Queued	13 days ago		● ibm_osaka
csrysgyvkv50008g9tv0	✓ Completed	13 days ago	13 days ago	ibm_osaka

I spoke with Carlos on Teams to discuss where I am now and the best steps going forward. Carlos thinks it is interesting to apply the two UML approaches, his and Perez-Castillo’s, and this is worth exploring. This could be used for both NISQ and standard QC systems. As I have spent time learning about QAOA and VQE, this would be a good place to start.

So, I will apply the two UML approaches to VQE rather than focusing on communication diagrams. I will start with a class diagram and break down how the algorithm operates; the next best step will be to code the algorithm in Qiskit following the IBM tutorial. Once I’ve had a go at working with it practically, I will start creating UML diagrams.

Carlos advised that this would allow me to examine the strengths and weaknesses of the two UML approaches. I could also investigate other aspects beyond VQE, such as superposition and entanglement, as discussed in the paper, Grover’s algorithm, and how the UML approaches handle both NISQ and complete QC systems. However, having one complete piece of work is better than many half-baked pieces, so I will just focus on VQE for now. I sent Carlos a link to the UML profile and VQE paper.

I will start drafting the thesis layout's bare bones, and once this is done, I'll follow the IBM tutorial on how to write VQE in Qiskit.

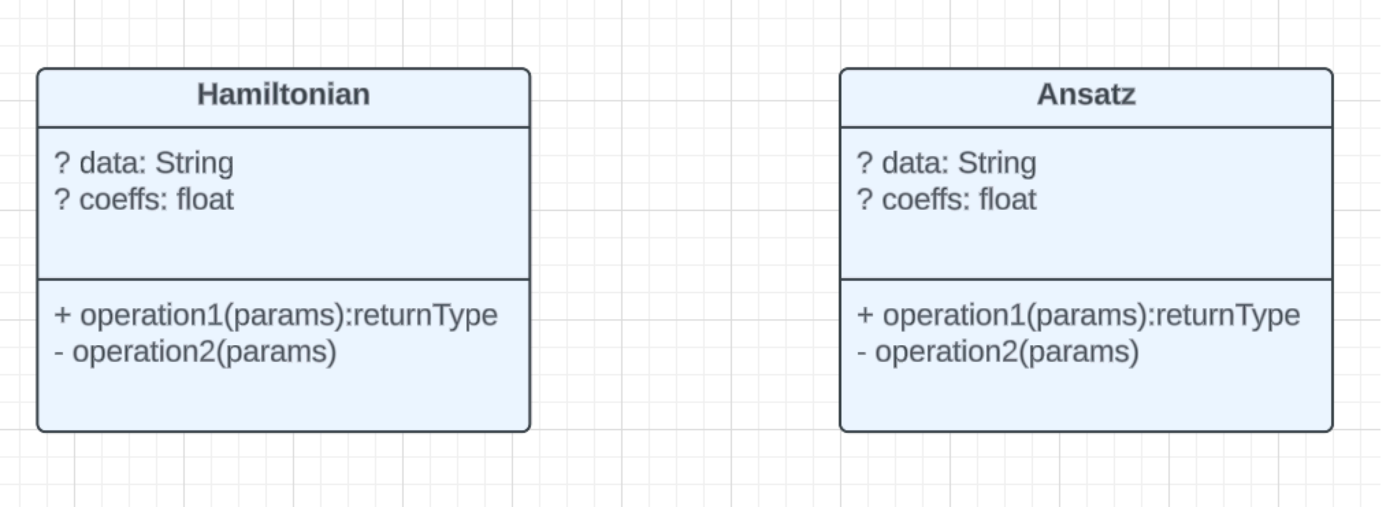
Links: Variational Quantum Eigensolver | Qiskit Global Summer School 2023 Eigenvectors and eigenvalues | Chapter 14, Essence of linear algebra Sean Carroll: Hilbert Space and Infinity

I have started a summer internship with Giftease, so I’ll have less time to focus on the dissertation for now; however, I have been granted a 3-month extension to my deadlines. I will attempt to work on the dissertation as and when I can during the internship.

18/06/2024

Links related to VGE algorithms: vqe-molecules.ipynb Instances and Extensions Variational quantum eigensolver - started with this tutorial

I looked into following one of the above examples and coding in Qiskit. I used the following tutorial on installing Qiskit and ultimately created a new environment via Pycharm to install and code in Qiskit. I then copied the code in the example, which uses real Quantum hardware to execute.



At this point, I didn’t necessarily understand what was happening and became quite conscious that there’s only a 10-minute allowance monthly to use this hardware. I thought it could have been used on a quantum simulator rather than hardware.

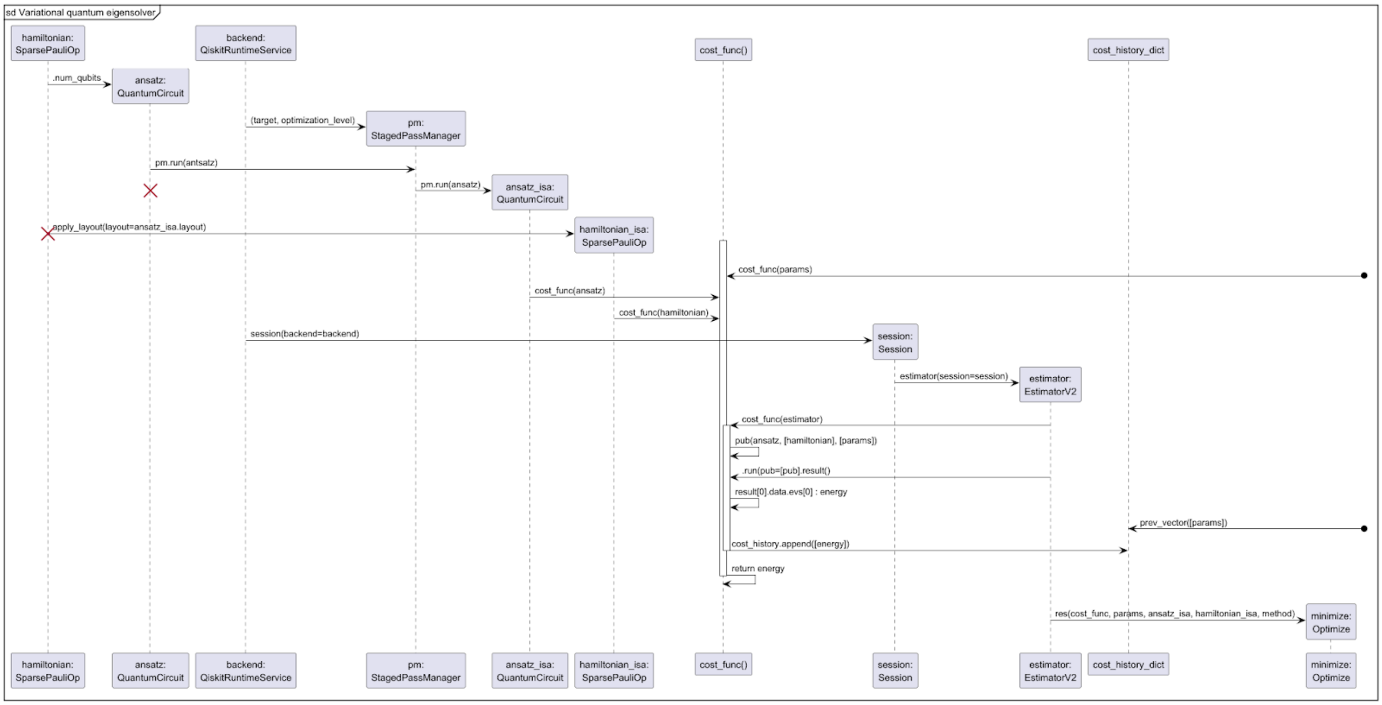
My friend pointed out that I didn't need to write code for what I wanted to do, which is true. So, I will return to reviewing the IBM articles on VGE and select one to translate into a QUML class diagram.

01/07/2024

Whilst looking at the IBM learning page, I started looking into this - Single systems

This was a helpful resource, but it gets down to the minute details when they may not be necessary for UML/VGE.

I started building the VGE algorithm using [this tutorial] (https://learning.quantum.ibm.com/tutorial/variational-quantum-eigensolver) in Lucidchart. I will continue with this. Links from today after the picture:


sd Variational quantum eigensolver
hamiltonian: SparsePauliOp
backend: QiskitRuntimeService
cost_func()
cost_history_dict
.num_qubits
ansatz: QuantumCircuit
(target, optimization_level)
pm: StagedPassManager
pm.run(antsatz)
pm.run(ansatz)
ansatz_isa: QuantumCircuit
apply_layout(layout=ansatz_isa.layout)
hamiltonian_isa: SparsePauliOp
cost_func(params)
cost_func(ansatz)
cost_func(hamiltonian)
session(backend=backend)
session: Session
estimator(session=session)
estimator: EstimatorV2
cost_func(estimator)
pub(ansatz, [hamiltonian], [params])
.run(pub=[pub].result()
result[0].data.evs[0] : energy
prev_vector([params])
cost_history.append([energy])
return energy
res(cost_func, params, ansatz_isa, hamiltonian_isa, method)
minimize: Optimize
hamiltonian: SparsePauliOp
ansatz: QuantumCircuit
backend: QiskitRuntimeService
pm: StagedPassManager
ansatz_isa: QuantumCircuit
hamiltonian_isa: SparsePauliOp
cost_func()
session: Session
estimator: EstimatorV2
cost_history_dict
minimize: Optimize

Links: Variational Quantum Eigensolver SparsePauliOp EfficientSU2 Operator UML Class Diagram Tutorial Build and deploy quantum programs with Qiskit Runtime Instances and Extensions Variational Algorithms Single Systems The Pauli matrices Lucid Chart

13/07/2024

Created a “VGE- Layman terms” Google doc to break down the elements of the algorithm I’m following (Variational Quantum Eigensolver) into simpler terms so I could fully understand what the code was doing.

I also watched some videos to refresh myself on the algorithm, as I had struggled to explain it simply when discussing dissertation progress with peers.

Links: The Variational Quantum Eigensolver (VQE) in a Nutshell The Variational Quantum Eigensolver — Programming on Quantum Computers — Coding with Qiskit S2E4

25/07/2024

I spent the past few weeks working on the “layman's terms” document, which I completed tonight. This document thoroughly investigated the quantum aspects of the problem and used Gemini/documentation for the more general coding aspects later in the algorithm.

Now that this document is completed, I am more confident in understanding what is happening in the code. I will now return to creating a diagram based on this algorithm. My process will start with a rough diagram and then implement

UML standardisation. I will then adapt to Carlos' quantum extension of UML and implement the quantum UML profile diagram. Possibly, right after Carlos' diagram, I will start writing the dissertation as I can begin discussing aspects like the background before having both diagrams completed.

I have struggled to balance dissertation and internship work. I lose my memory regarding topics for the dissertation and have to spend time familiarising myself with the domain when returning to this work, a two-step forward, one-step back situation.

19/09/2024

I used a Pycharm plugin to translate the code from this tutorial (written as a Python file instead of a Jupyter notebook) into a class diagram, which I could use to understand class diagrams and then translate into the Q-UML format. It could not translate it into a class diagram, whether or not because it can't recognise the Qiskit library or because the plugin needs internal classes within the file if it does not look for the external classes from libraries used in the code (i.e. SparsePauli0p, EfficentSU2).

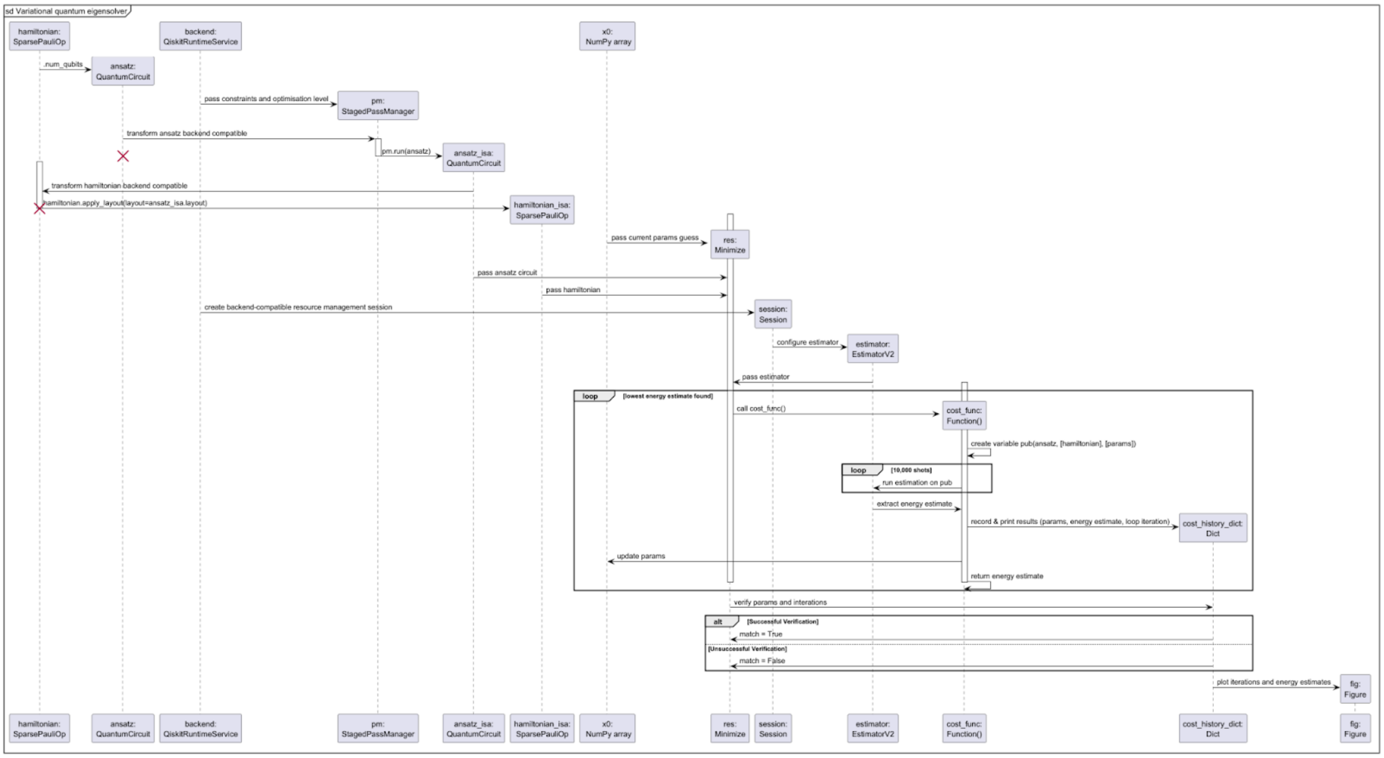
This then raised the question of the most suitable UML diagram for constructing the first UML diagram of the VQE Qiskit tutorial code. I had initially considered constructing communication diagrams. However, a sequence diagram may be more suitable. It is similar to communication diagrams, whereby both are behavioural diagrams that demonstrate the dynamic interactions of a system; however, sequence diagrams are more commonly used.

The VQE algorithm consists of interactions between different classical and quantum modules to solve the problem; therefore, a UML diagram that illustrates a series of messages may be the most suitable initial diagram. There is also the advantage that sequence diagrams have already been constructed in the QUML and quantum UML profile texts, as opposed to communication diagrams, which would mean I would have direct references during their construction.

I began looking into sequence diagrams and their fundamental aspects, such as lifecycle dependency. I will now start translating the VQE Qiskit algorithm into a sequence diagram.

24/09/2024

I returned to the dissertation after the end of the internship last week and started creating the VQE Qiskit algorithm using Plant UML as a Pycharm plugin. A few months prior, I had some experience writing in PUML and felt this was less labour-intensive than constructing the diagram using a GUI, as PUML uses plain text code to construct the diagrams.

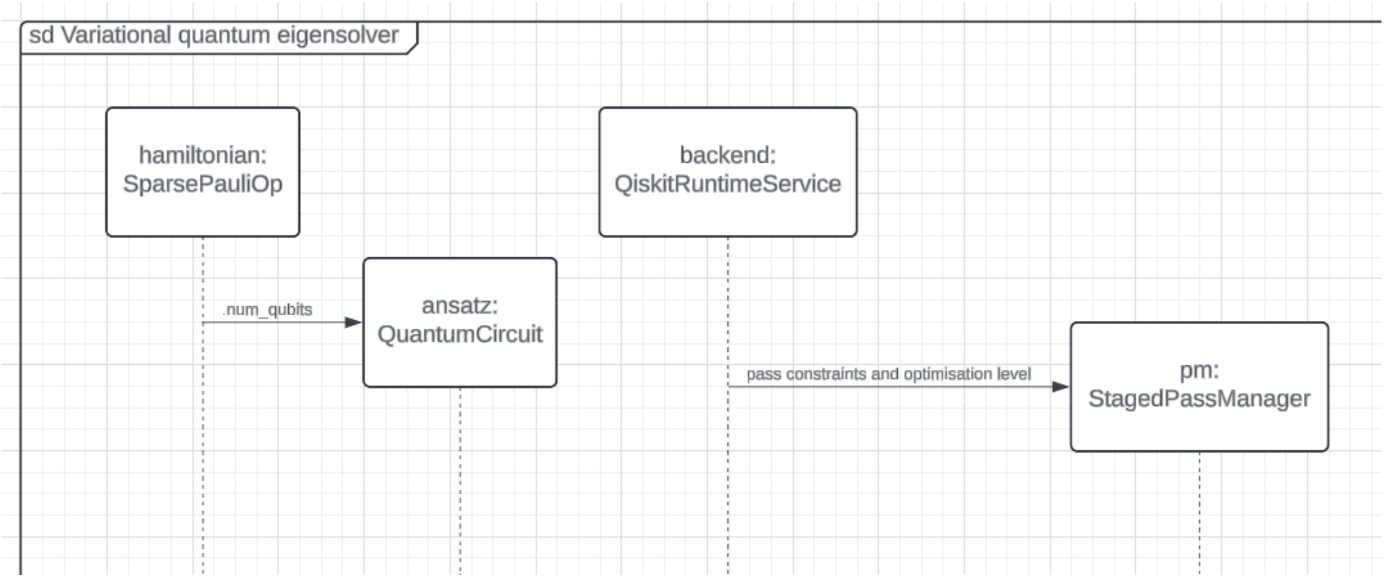

sd Variational quantum eigensolver
hamiltonian: SparsePauliOp
backend: QiskitRuntimeService
x0: NumPy array
.num_qubits
ansatz: QuantumCircuit
pass constraints and optimisation level
pm: StagedPassManager
transform ansatz backend compatible
pm.run(ansatz)
ansatz_isa: QuantumCircuit
transform hamiltonian backend compatible
hamiltonian.apply_layout(layout=ansatz_isa.layout)
hamiltonian_isa: SparsePauliOp
pass current params guess
res: Minimize
pass ansatz circuit
pass hamiltonian
create backend-compatible resource management session
session: Session
configure estimator
estimator: EstimatorV2
pass estimator
loop
[lowest energy estimate found]
call cost_func()
cost_func: Function()
create variable pub(ansatz, [hamiltonian], [params])
loop
[10,000 shots]
run estimation on pub
extract energy estimate
record & print results (params, energy estimate, loop iteration)
cost_history_dict: Dict
update params
return energy estimate
verify params and interations
alt
[Successful Verification]
match = True
[Unsuccessful Verification]
match = False
plot iterations and energy estimates
fig: Figure
hamiltonian: SparsePauliOp
ansatz: QuantumCircuit
backend: QiskitRuntimeService
pm: StagedPassManager
ansatz_isa: QuantumCircuit
hamiltonian_isa: SparsePauliOp
x0: NumPy array
res: Minimize
session: Session
estimator: EstimatorV2
cost_func: Function()
cost_history_dict: Dict
fig: Figure

Links: Variational Quantum Eigensolver Overview of Sequence Diagrams. Sequence Diagram SparsePauliOp Target UML sequence diagram how to draw instance passed to another instance Preset Passmanagers StagedPassManager PassManager How to destroy a participant? Operator EfficientSU2 Explore the UML sequence diagram Methods in a Sequence Diagram UML Sequence Diagram Sequence Diagram: Background Loop minimize

25/09/2024

I completed the first draft of VQE SD using PUML. I had to reconsider the order of the “Minimize” and “Cost Function” methods as Min is essentially an outer loop implementing the Cost Func within it. This, in turn, meant I’d included the loop and alt fragments with the alt depicting the verification process at the end of the search.

Modifications will likely be made to the message types (request and response); however, this can be reviewed when applying the QUML standard. I will now copy the PUML SD onto Lucidchart and apply QUML standards. I will re-read Carlos's paper to familiarise myself with the modifications.


sd Variational quantum eigensolver
hamiltonian: SparsePauliOp
backend: QiskitRuntimeService
.num_qubits
ansatz: QuantumCircuit
pass constraints and optimisation level
pm: StagedPassManager

Links: Target EstimatorV2 minimize EstimatorOptions

26/09/2024

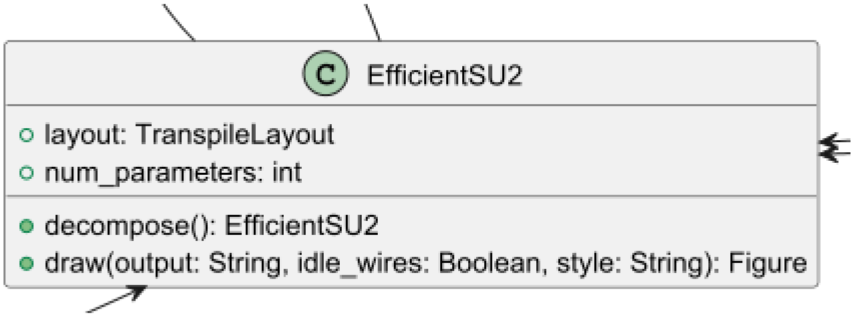
I’ve started drawing up the SD as an SD using QUML in Lucid Chart

So far, I am confident with everything up to Hamiltonian (the first one!), but I need to assess whether ansatz would be considered a classical or quantum object.

I’ve tried to refer to the documentation regarding the classes used to construct the instances. However, there doesn’t appear to be any information regarding whether they access classical or quantum hardware. I’ve had to use a combination of common sense and ChatGPT to make my assessment, which is that the information will be held classically until there is a need to access quantum hardware to search the parameter space.

Links:

SparsePauliOp: num_qubits numpy's complex128 conversion QiskitRuntimeService IBMBackend Target NotebookLM


C EfficientSU2
○ layout: TranspileLayout
○ num_parameters: int
● decompose(): EfficientSU2
● draw(output: String, idle_wires: Boolean, style: String): Figure

27/09/2024

I finished transferring the SD to Lucidchart and implementing QUML notation on it. I now need to start drafting the early deliverable, which should be an overview of the dissertation's contents and a walkthrough and explanation of the QUML VQE SD.

Links:

Circuit Library QuantumCircuit PassManager Add and customize shapes in Lucidchart

02/10/2024

I started working on the dissertation plan to submit for the early deliverable due on the 5th alongside the Q-UML VQE SD. I completed the problem description/goals and objectives.

03/10/2024

Continuing with the early deliverable, keeping a record of resources used whilst drafting it. Information about Q-SE 2020

Links: ICSE 2020 First International Workshop on Quantum Software Engineering (Q-SE 2020) Quantum software engineering A Quantum Software Modeling Language Design of classical-quantum systems with UML Towards a Quantum Software Modeling Language ICSEW'20: Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops Modelling Quantum Circuits with UML Computing Noisy intermediate-scale quantum era Variational quantum eigensolver A brief overview of VQE What is Profile Diagram in UML?.

04/10/2024

I’m worried about how much time I will have to complete all objectives listed in the early deliverable. I should not do the polling and prioritise cresting multiple UML diagrams. Links: Improving information system design: Using UML and axiomatic design

05/10/2024

I finished the early deliverable and submitted it.

09/10/2024

Resumed face-to-face meetings with Carlos and his peers. This was very helpful; I spoke with Chris and Lee, establishing a core question: what is an Eigensolver? We talked to Carlos, who advised that it would be a way of finding all the eigenvalues (energy states) in a Hamiltonian.

I spoke with Jorge about when to determine if a Qiskit object is classical or quantum. He confirmed that my assumption was correct and that there would be some sort of translation process from the quantum object to a classical one for it to receive the classical information. Jorge also explained how increasing a circuit by the number of qubits increases the complexity in polynomial time, whereas increasing the Hamiltonian by adding more terms increases the complexity exponentially.

Carlos also reassured me that focusing on VQE and the two SD adaptations would be enough for the project. I mentioned that one of the objectives of the early deliverable was to create multiple diagrams and model fundamental quantum concepts. There won’t be enough time to do that effectively. It means there should be much to discuss for future work and when reflecting on the project.

Qiskit club meeting notes:

Questions:

- Qiskit documentation confirming when the quantum hardware is accessed

- UML for quantum hardware and fundamental quantum properties
- What is an eigensolver? (finding the eigenvalues of a system) eigensolver can be quantum and classical

- Matlab
- Lanczos algorithm
- Quantum alternative to factoring, easy to verify, difficult to know, bqb complete Sparse matrix - shallow circuit - depth (qubit count) breadth (circuit chain, has lack of coherency)

11/10/2024

I have moved on to translating the VQE SD to the quantum UML profile format. I have re-read the paper and have found it quite challenging to understand. I read the UML book on class diagrams because I thought it would help me understand the UML profile diagram.

UML profiles are lightweight extensions of UML, which is referred to as a meta-model. A class diagram shows how the UML diagrams are constructed, including stereotypes generated for the UML profile. The UML profile can be seen as a package applied to the diagram.

Links:

UML Profile Diagram What is a UML profile diagram and when is it used? [duplicate] Profile Diagram - Unified Modeling Language(UML) Profile (UML) What is Profile Diagram in UML? Creating a profile diagram Profile Diagram A Comprehensive Guide to PlantUML Activity Diagrams: Everything You Need to Know

12/10/2024

Quantum UML profile paper refers to Carlos's first paper as a domain-specific language: Domain-Specific Languages Guide

"A Domain-Specific Language (DSL) is a computer language that's targeted to a particular kind of problem, rather than a general purpose language that's aimed at any kind of software problem. Domain-specific languages have been talked about, and used for almost as long as computing has been done."

During this project, I thought there was an issue with the fact that some software developers don’t even use UML and that the extent of their use of it would have just been during education. It made me think there should be some consideration for having a quantum UML adaptation, which can also be used more informally. I have come to realise that this possibly doesn’t matter; even if people don’t use UML, they will choose how they want to depict their informal diagrams, whether through colour, shape or text. The important thing is to consider how UML can be adapted effectively for quantum, as UML is the standard modelling language for system design.

I have had a crack at the UML Profile diagram. Having a clear explanation of the different stereotypes will be crucial. I will create a separate Word doc to define these stereotypes. I don’t think constructing the diagram afterwards would be too difficult. I think it will be good to start on the dissertation and start writing about the VQE algorithm first. Getting a clear understanding of this and referring to the meeting notes where we discussed eigensolvers will be beneficial in clearly understanding how each type of diagram should depict it.

Links:

A Two-tiered Methodology to Extend the UML Metamodel Modelling Quantum Circuits with UML 24765-2017 - ISO/IEC/IEEE International Standard - Systems and software engineering--Vocabulary A Proposal for a Formal Definition of the Design Concept Design Patterns — Ten Years Later Towards model-driven quantum software engineering Toward a standardized methodology for constructing quantum computing use cases

13/10/2024

I wrote a Word document to understand the different stereotypes. I will then go through the VQE algorithm written in Qiskit to identify which stereotype should be applied to each element to construct the UML profile SD diagram. The dissertation should distinguish between the VQE algorithm and, when discussing methodologies, discuss Qiskit and Qiskit's implementation of the VQE algorithm.

14/10/2024

I've tidied up the logbook, included the early deliverable in my overleaf dissertation draft, and updated the bibliography.

15/10/2024

I briefly reviewed the Lanczos algorithm, which finds the minimum and maximum eigenvalues and eigenvectors in hermitian matrices.

Whilst watching the video, I used Chat GPT and google to clarify terms that came up:

Eigenvector and Eigenvalue Eigenvector: A vector that stays in the same direction after a transformation. Eigenvalue: The factor by which the eigenvector is scaled during the transformation.

Why Matrices Have Eigenvalues and Eigenvectors They show how the matrix (transformation) stretches or compresses along specific directions (eigenvectors).

Orthogonal vs. Orthonormal Orthogonal: Vectors are perpendicular (90 degrees to each other). Orthonormal: Vectors are perpendicular and have a magnitude of 1.

Orthogonal Matrix It contains vectors that are both orthogonal and normalised (orthonormal). Its rows and columns are perpendicular and have a length of 1.

Hermitian Matrix A complex version of a symmetric matrix where A = A dagger (equal to its conjugate transpose). Every symmetric matrix with real entries is also Hermitian.

Transpose of a Matrix Switches rows and columns. A^T flips the matrix over its diagonal.

Conjugate Transpose Transpose the matrix and take the complex conjugate of each element.

Lanczos Algorithm An iterative method for finding a Hermitian matrix's smallest and largest eigenvalues/eigenvectors. It is efficient for large matrices as it reduces the matrix to a simpler tridiagonal form.

Links: Lanczos algorithm Hermitian matrix The Lanczos Algorithm, Part 1/2

16/10/2024

Qiskit Club 2nd Meeting Notes:

- Rendevous Games - find each other in a search space without communicating
- Artemis Moon Landing
- What is a wave function? A wave function describes a particle's quantum state, providing information about the probability of finding the particle in a particular position or state when measured.
- What is Born's rule? Born's rule states that the probability of finding a particle is the square of the wave function’s absolute value.
- What is Bayes rule? Bayes' rule calculates the probability of an event based on prior knowledge of conditions related to the event. It updates the probability as more evidence becomes available.

The topic discussed was Random Number Generation using QC.

- Evolutionary algorithms (work done by Jorge). Fitness function = variation of the randomness
- Fundamental entropy (ignorance) - non-randomness
- Qubit to represent a bit, 8 bits for 256 number values (0 to 255)
- Hadimard gates and CNOTS to randomise
- Evolutionary algorithm to find the sequence of gates to pass randomness test (RNG test, ball/bull test?)
- Different tests for different randomness
- How do we prove randomness? Is this even possible?
- Examples of binary sequences given with no information on how they were produced:
    - 101101011
    - 111111111
    - Which one is more random? If we keep getting 1’s in our 1111 sequence, the probability of it not being random increases and the probability of getting 0 increases exponentially (probability of 1 * probability of 1 *, etc.)
- RNG would have health tests that would not allow sequences such as 1111 or 0000
- Kolmogorov complexity - shortest program complexity to output a sequence
    - Complexity/Randomness/irreducible
    - Programmes to write the sequence examples
        - 111111111 (print 1, repeat) less complex
        - 101101011(write a strong, print 1, print 0, print 1 etc.) more complex

18/10/2024

I started working on a Word document, “Defining VQE Qiskit Objects”, so I can label each element of the UML VQE diagram with its stereotype, if it has one, and write down a general explanation of each object's datatype. Once I can establish this for all elements in the VQE Qiskit algorithm, it will be easy enough to construct it as the UML diagram, and I can move on to writing the dissertation. It should also help clarify if I have done the QUML VQE SD correctly.

One thing to note is that Scipy imported the minimise function, and it takes the cost function as an argument. I’ve made it a quantum object in QUML VQE as it contains a quantum object, but I need to nail down whether or not this is the case.

Links:

IBM Upcoming sunset of backend.run sparse_pauli_op.py PauliList pauli_list.py efficient_su2.py Qiskit Runtime Github Qiskit QiskitRuntimeService qiskit_runtime_service.py IBMBackend Target target.py

19/10/2024

Continued with “Defining VQE Qiskit Objects”. I was unsure if the pass manager should be assigned the stereotype <> as it involves some communication with the classical ansatz circuit to be backend compatible with a quantum computer; however, the information held is classical, and the transformations made remain classical, there is currently no interaction with quantum hardware, so I have decided that this remains a classical element. A note from the “The five Quantum UML Stereotypes” document:

The <> lifeline is responsible for creating an instance of a <> object. When communication is made to quantum hardware, this stereotype will be applicable.

Links: StagedPassManager passmanager.py Transpile with pass managers Transforming Quantum Circuits using Qiskit's Transpiler with Matthew Treinish: Qiskit Summer School Quantum processing units

21/10/2024

Continued with “Defining VQE Qiskit Objects”. I have changed the backend from classical to <>. Although information regarding quantum hardware is stored classically, I did not consider the broader scope of what this object does. It will facilitate communication between the classical and quantum software when executed using the cost_func method.

Links: EstimatorV2 estimator.py RuntimeJobV2 estimator.py Session session.py Qiskit Runtime minimize

Completed all elements in “Defining VQE Qiskit Objects” and updated the VQE Quantum UML Profile Plantuml file. I will still need to review the messages to confirm they’ve all been assigned the correct stereotypes.

I’m unsure if I have defined cost_func correctly and have changed it from <> to <>. I’ll continue with the messages for now and maybe come back to it in the future, but I think I can also create the diagrams and either move on with making class diagrams or start to write up the dissertation.

22/10/2024

I continued with “Defining VQE Qiskit Objects” to define the message types. I need to make a note regarding primitives and backend, which are two types: estimator and something else. I think this is relevant to discuss in the dissertation.

I’ve been using Notebook LM to bounce ideas off during this process. I’ve been able to upload the research papers, my document about quantum stereotypes and website links to the Qiskit documentation I’ve been looking into. It’s not foolproof; it has given me incorrect information if I convince it, but it’s helpful as a sounding board.

Notes from Notebook LM:

You are correct in stating that access to the quantum hardware occurs when a primitive, like the Estimator, is executed via the .run() method.

Primitive Execution: The execution of a primitive triggers the submission of a job to the quantum backend.

Job Allocation: Both the backend and the session object play a role in managing the job allocation to the quantum hardware. The session helps prioritise jobs submitted within its context, while the backend manages the overall job queue and execution.

I have determined for now only to include <> and not <> in my sequence diagram. This would probably be helpful to discuss as a Qiskit club. I have abstracted away from some of the requests/replies in the diagram (for example, one instance making a request to the other and the other replying with the information it obtains, just keeping it to the request). In the shots loop, 10,000 iterations of the circuit are run, and a quantum reply is not received for all of them. Only the result will be obtained, which will be transformed into classical information when received; therefore, I have left the depiction as a classical reply.

I will keep it as I've done it, complete my UML Quantum Profile diagram, have it as the LucidChart version, and move on to transforming them into class diagrams of both quantum styles. My goal is to get this done before the end of the month, which will allow me to focus purely on the dissertation throughout November.

I adjusted the text's formatting, which made me think I needed to do this to the QUML version for consistency.

I have now completed the VQE sequence diagrams for both UML versions. I will now move on to creating class diagrams.

Links: EfficientSU2 EstimatorV2 Introduction to Qiskit Runtime execution modes Variational Quantum Eigensolver Introduction to Qiskit Runtime sessions Session

23/10/2024

Qiskit Club. Watched a talk from Joseph about his defence, discussing blockchains and quantum advantage. Notes:

- Hash function, shah 256
- Super conducting quantum device vs other devices
- Grover = quadratic advantage. Shor = exponential advantage.

25/10/2024

Started on the Class Diagram, lots of links for today as having to clarify information for attributes and operations. Have finished the day at the CostFunction method, need to think of a way to represent the cost_history_dictionary attribute as it has several data types, I believe will need to just detail this as attributes in its own element on the diagram and then reference that in the CostFunction element. Link to refer to regarding this: how to draw class diagram that shows a call to a static method of another class & attribute having multiple types in class diagram

Information from ChatGPT:

To model a union of two different data types in PlantUML, you can define an abstract class for the general type and then create two concrete subclasses for the specific types. You would then indicate that the attribute in your main class uses the abstract class as its type. Here's how you can do this:

```
@startuml
abstract class CostHistory {
    +calculate(): float
}

class IterationCount {
    +value: int
}

class EnergyHistory {
    +values: List<float>
}
```

```
CostHistory <|-- IterationCount
CostHistory <|-- EnergyHistory

class SomeClass {
    +history: CostHistory
}
@enduml
```

Explanation:

- CostHistory: This is the abstract class representing the general data type.
- IterationCount: This subclass represents an integer type for the iteration count.
- EnergyHistory: This subclass represents a list of floats for energy values.
- SomeClass: This class uses CostHistory as the type for its history attribute.

Notes:

- Generalization Set: In UML, generalization indicates that subclasses inherit from a superclass, which aligns with the concept of union in type modeling.
- Disjoint and Complete: You can add notes or constraints to clarify that these subclasses are disjoint (no overlap) and complete (cover all possibilities).
- Abstract Type: The abstract class cannot be instantiated, which emphasizes that the attribute must be one of the subclasses.

This representation clearly conveys the structure to a reader and can be used to inform implementation in languages that support such constructs, like C++ or XSD.

For more on UML diagrams and PlantUML, you can check the official PlantUML documentation or UML specifications from the Object Management Group (OMG).

Links:

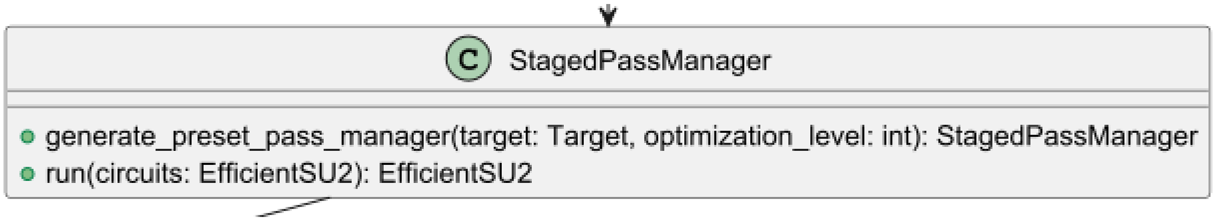
SparsePauliOp EfficientSU2 TwoLocal NLocal What is a 'NoneType' object? UML Class Diagrams Reference. Decompose Circuit Library UML Design class diagram: Class with another class as attribute? UML Best Practice: Attribute or Association Class Diagram Object Diagram Syntax Error: Object name with space characters What is Object Diagram? EfficientSU2 QiskitRuntimeService IBMBackend target.py StagedPassManager Preset Passmanagers Built-in Functions: Int SparsePauliOp Built-in Functions: Complex PauliList NLocal n_local.py Decompose DAGCircuit QuantumCircuit.decompose() should take which gate(s) to decompose Class Diagram SparsePauliOp

28/10/2024

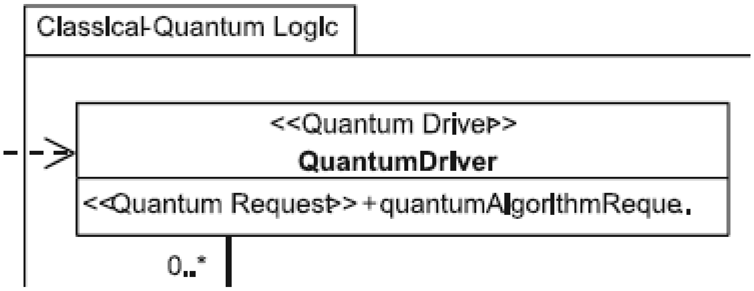
Continued with class diagram, realised I had missed that the ansatz needs to pass the number of params in the creation of x0 when creating SD, helpful to do both a static and dynamic diagram when understanding full scope of the algorithm.

I'm making a design choice to use specific classes in the tutorial such as EfficentSU2 as opposed to general QuantumCircuits, need to think about this when discussing the diagrams in the dissertation. Do I mark down the ansatz as Quantum Circuit, _CircuitsT or EfficentSU2? I'm going to make the decision to use EfficentSU2 as we're speaking strictly about this tutorial, but it's a design choice to consider.

Just remember:


C StagedPassManager
generate_preset_pass_manager(target: Target, optimization_level: int): StagedPassManager
run(circuits: EfficientSU2): EfficientSU2

decompose was originally QuantumCircuit



run circuits: was originally circuits: _CircuitT

I’ve finished the Class diagram in Plant UML, it looks quite messy in all honestly and not as satisfying as when creating the Sequence Diagram. I think I still need to make some revisions in particular to go over muplicities, public/private/protected and make and changes to include notes and improve illegibility. I will do this however when going over it in LucidChart. I’ve set up a repo on GitHub for the project so all the work will automatically update there. I will move my logbook on to there also and have it set up as a Readme file.

Links:

EfficientSU numpy.ndarray PrimitiveResult EstimatorV2 RuntimeJobV2 numpy.random.rand numpy.random.random Session EstimatorOptions QiskitRuntimeService Backend minimize OptimizeResult matplotlib.pyplot.subplots matplotlib.figure matplotlib.axes matplotlib.pyplot.plot

29/10/2024

While using Lucid Chart, I discovered a diagram as a code feature that allows you to write in Mermaid. This may have been the best avenue to explore; Plant UML was suitable for sequence diagrams but looks incredibly messy for a class diagram.

The Dict object depends on cost_function; it can't exist outside of it as that's where it's created. Or either aggregation/composition. I need to go through the book to confirm. This may also need to be the case when considering the cost_func relationship to minimize.

I finished translating the diagram to Lucidchart, but it's still messy. Rupert suggested splitting the diagram; both cost function and minimize take other classes as parameters; it may be better to have cost_function as a separate diagram, which is appended to the main one with minimize. Also, I believe I might need to make QiskitRuntimeWService an abstract class, as it invokes a Backend object (which can't be called directly) and a target object; I've amalgamated these two into the QiskitRuntimeService. Still, it's perhaps lacking in accuracy (I have recorded target as an attribute of QiskitRuntimeService, which isn't necessarily true).

I'm reviewing the UML book and the Class Diagram chapter to make adjustments as I go to the diagram. I've included multiplicities of the attributes so far and tested whether the args for minimise need to be ordered (they do). I confirmed that all parameter prepends can remain default to in; no out parameters exist.

I will continue tomorrow with visibility markers.

30/10/2024

While partway through the visibility markers, I came to a point where I needed to clarify how to depict QiskitRuntimeService. This is used as the main object when interacting with an IBMBackend object, which it creates; the IBMBackend object is what interacts with the quantum hardware. QiskitRuntimeService manages logging in to an IBM cloud account and other matters; it is not an abstract class, as I had thought it might need to be. It is a wrapper class. The IBMBackend object contains the attribute "target" required to generate the pass manager. I have created an IBMBackend object in the class diagram with a target attribute and showed it as a dependency on QiskitRuntimeService.

I was unsure if this also needed to be included in the sequence diagram, but I've decided it doesn't. An instance of IBMBackend is never created; it is interacted with through QiskitRuntimeService. I've shown that QiskitRunTime uses the target attribute by passing constraints to the pass manager.

I'm going to return to visibility after I've spoken to a friend about it. I believe most everything in Python is public, but I may need to consider how things should be depicted in the UML diagram. For example, the least_busy() method for QiskitRuntimeService should only be accessed in this class in the context of this diagram, so should it be shown as private? Or should I omit visibility, as Python and Qiskit should be public attributes and methods?

I can put a pin in this for now. I think the most pressing matter is making the diagrams more legible and whether I need to split them up and combine them.

I spoke to a friend; Python is inherently public anyway. I had given the example of whether least_busy should be considered private in the context of it only being used with QiskitRuntimeService, but this isn't the correct understanding. It could be calling separate private methods to execute the operation, but least_busy remains public. It may be best to remove visibility, as the attributes and methods are all visible.

Discussed _CircuitsT and me using EfficentSU2 instead as datatype, which is the correct design choice as we're not worrying about all circuits, just efficentsu2 in the case of this algorithm

He was concerned that this appeared to have some data flow aspects. I need to revisit how I've written the reading directions. I've considered them more as data flows when they are not.

I've now revised the diagram to adjust the reading directions, as I'd been considering them as data flows instead. I've also separated minimize so it looks clearer. So today, I've covered visibility, reading directions, and made visual adjustments. I will continue reviewing the book and making adjustments. I will try to get this done in the next couple of days, and then I just need to create a Carlos version; diagrams will be done, and I will move on to the dissertation!

I've managed to assign the multiplicities of the attributes correctly. I've gone over the reading direction and association names again, so this has been changed to include general reading direction information, the multiplicities with the attributes where necessary, and anything where one element is passed to another. I've left notation as it should be clear. I am now happy with the diagrams, i feel like I could keep going over the book and over them but concious of time and think it will be best to move on and anything missed can be discussed in the disseration. I will focus the rest of the week on their compliance with the Quantum UML profile and QUML format. I aim to finish this by the end of this week so I can start the dissertation next week.

Links:

Visualize Circuits Runtime Service UML Dependency Class diagram multiple classes uses same class Splitting up a UML Class Diagram Should I use public attributes or public properties in python Python private method for public usage Diagrams dependency relationships UML Dependency Diagrams association relationships UML aggregation vs composition

31/10/2024

I'm cross-referencing the VQE CD with the quantum UML profile paper. The class stereotypes should remain consistent with the SD, as classes are the same as elements in an SD. I'm trying to establish which operations would be considered `<<quantum request>>` . Confident with the estimator.run() being a quantum request. Decided that it is also apt for cost_func and minimize. Going by the diagram in the paper:

I think it's correct, as both methods call the quantum request algorithm.

Also, considering whether or not the diagram should be split into packages, I think I can omit this for now as I'm essentially going through a classical-quantum logic package (referring again to the example diagram in the paper)

I think it's safe to move on to making a copy for the QUML format now. Once that's done, I can spend the rest of the week comparing diagrams to the book chapter, seeing if I can make any additional adjustments before finalising the diagram.

Working on the QUML diagram, the paper notes that anything that contains quantum functionality should be considered quantum. You also need to mention this for attributes and operations and their datatypes. For example, with the estimator object, I have stated that the RuntimeJobV2 is quantum. Even though it is classical, it manages communication with quantum hardware, so I should mark it as a quantum data type.

Another question: Should I mark the parameters for cost_func and minimise as aggregated relationships and not communication? Does it become part of it?

I am also wondering about creating packages for the quantum driver section, the minimise section and the remainder.

Determined a target object (attribute of IBMBackend) will remain classical as it does not interact with quantum hardware. At a point of contention with QRS(QiskitRuntimeService) as it has two operations which return IBMBackend as a datatype; however, the operations themselves don't inherently need to access quantum hardware—determined to have the datatype as bold and the operation as not bold to distinguish classical/quantum.

Primitive Result object: The backend(QRS) will manage communication with quantum hardware. The PR object contains information from quantum operations classically; it would be the backend that communicates this information to it via a classical channel. Therefore, PR is a classical data type.

I've finished the QUML version and, technically, the Quantum UML profile version. I now just need to do some final comparisons with the book to finalise the CD diagrams and then move on to the dissertation. I need to look into aggregation and packages mainly, but I will run through the CD chapter.

03/11/2024

I have included packages and aggregations to the class diagrams. The diagrams are now completed! I am exporting all diagrams, saving them together and will start focusing on writing the disseration!

04/11/2024

Started working on the disseration in Overleaf, finished first draf on section explaining the VQE algorithms implementation in Qiskit.

Links: Add extra level of sections SparsePauliOp Complex Operator Pauli matrices Built-in Types

05/11/2024

Today I was managing how to orgnise the sections related to UML in the disseration and started writing the UML Design Choice section and the VQE Sequence Diagram section. I ended the day with finishing the sequence diagram section and making a start on the class diagram section.

06/11/2024

Today I refined the VQE and Qiskit section and have sent it to Carlose to get feedback. Attended Qiskit Club where I got really helpful information from my peers regarding how the hamiltonian is represented on the ansatz in order to find it's ground state in the algorithm, and also the significance of pi in relation to the parameter arrays given to the ansatz.

Qiskit Club notes:

- Attended talk on quantum super conductors
- Hamiltonian solving shroedinger equation
- Eigenvalue equation
- Opeator matrix multiplied by vector should be equal to eigenvalue multipled by the same vector
- Science/chemistry libre textbooks
- Explain PUB better at this communicated the mapping of Hamiltonian to ansatz
- Hamiltonian as a circuit, circuit as a vector, energy value of the vector, tweak parameters
- Photonics can be in room temperature
- 2 pi is a way to travel to every state in the bloc sphere

Links:

UML Element Frame Getting started with primitives Qiskit Runtime Sequence Diagram Tutorial - Complete Guide with Examples Overleaf - Bold, italics and underlining Standard order for bold + italic? UML Packages within packages What is an SDK?

07/11/2024

Continued working on the class diagram section of the disseration. Needed to make some adjustments to the class diagrams during this so need to update the diagrams and save new versions. I've updated the diagrams and completed the class diagram section. I'll now move on to the QUML section explaining what the format is and then each section discussing the diagrams in particular, I can then move on to doing the same with the Quantum UML profile sections.

Links:

What is the meaning of single and double underscore before an object name? Dependency Relationships UML Aggregation UML Composition

10/11/2024 and 11/11/2024

I've been working on the background and related papers section of the disseration. Spent a long time on UML as had to explain concepts like OOP and go over the book again to write good information.

Links:

7 Modelling Languages for software architecture Modelling Languages Techopedia Modelling Languages Re: [UML Forum] programming language versus modeling language object-oriented programming (OOP) Simula UML Spec Language Modelling Concepts and paradigms of object-oriented programming What is object oriented programming Object-oriented programming : an evolutionary approach Object-Oriented Programming Themes and Variations OOP vs POP Difference between OOP and POP ICSEW'20: Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops Quantum Software Engineering Object-Oriented Programming: Themes and Variations Q-SE 2020 Towards a Quantum Modeling Language Software Engineering for Quantum Computing and Quantum Computing for Software Engineering Towards a Quantum Software Modeling Language YT Rui Maranhao Abreu ICSE 2020 ICSE YT arxiv: what is it? Towards a Quantum Software Modeling Language Towards a Quantum Software Modeling Language Towards a Quantum Software Modeling Language Modelling Quantum Circuits with UML Object-Oriented Programming (OOP) and Procedural-Oriented Programming(POP): Two Paradigms, One Dilemma 2021 IEEE/ACM 2nd International Workshop on Quantum Software Engineering (Q-SE) 2021 IEEE/ACM 2nd International Workshop on Quantum Software Engineering (Q-SE) Table of Contents Q-SE 2021 Modelling Quantum Circuits with UML YT Design of classical-quantum systems with UML Computing

12/11/2024

Writing the UML profile section and started on NISQ today. Finished NISQ.

Links:

Noisy intermedia-scale quantum era Beyond quantum supremacy: the hunt for useful quantum computers NISQ Decoded: How Does a Quantum Computer Work? Quantum Advantage Quantum Computing's Potential for Exponential Growth Quantum Parallelism What is the difference between quantum computing and parallel computing? Non-layperson explanation of why a qubit is more useful than a bit? Advantages and Disadvantages of Quantum Computing: A Complete Guide Why is it hard to build quantum computers? https://www.frontiersin.org/journals/quantum-science-and-technology/articles/10.3389/frqst.2023.1128576/full How Many Qubits Are Needed for Quantum Supremacy? Whether Google achieved quantum supremacy depends on perspective What EXACTLY is an NP-Hard Problem? NP-hard: What is the definition of NP-hardness? What Can a Quantum Computer Actually Do? Quantum advantage or quantum supremacy - what is the difference?

13/11/2024

Attended Qiskit club, I was able to discuss my disseration work with the group and Carlos which was productive. Discussed if there was a need for highlighting quantum technologies in UML if UML is general purpose anyway. Need to think of a better way to answer this, I believe it's still required, especially if quantum devices are to be used commerically in order to depict how much classical/quantum units are needed.

Notes from Carlos feedback from the VQE Qiskit and NISQ sections I sent him:

- QIskit VQE, too dry, why are we doing this? Add a bit in the beginning to give them a warning as to why they need to run through the VQE Qiskit algorithm
- Weave a story, explain what we're focusing on and why in each chapter section
- NISQ
- NP hard solve in P is not nessecarily true. General consesus is there is an overlap, problems known as BQP
- NP hard - if and only if can reduce every problem in np to be solved with the solution of this problem
- Reduction - solve littlest number by problem with sorting solution
- Problems that can be solved in P not NP hard as NP hard can solve all NP. Quantum can solve some NP, not all
- Carlos solution; speed up certain problems such as factoring
- Focus on what you do understaand well

Spent the evening writing the VQE section, created a psuedocode algorithm.

Links:

Quantum Blockchain Miners Provide Massive Energy Savings Near-Term Quantum Computing Techniques: Variational Quantum Algorithms, Error Mitigation, Circuit Compilation, Benchmarking and Classical Simulation A variational eigenvalue solver on a photonic quantum processor https://en.wikipedia.org/wiki/Ansatz What is a Hamiltonian? Quantum Jargon Explained Hamiltonian Ground State Ansatz Generally, could VQE be used to prepare the highest energy state? Notation question about min and max functions Write pseudo code in latex Variational Quantum Eigensolver | Qiskit Global Summer School 2023 Quantum Algorithm (7): Variational Quantum Eigensolver Algorithm for Quantum Chemistry The Variational Quantum Eigensolver (VQE) Pseudocode example A variational eigenvalue solver on a photonic quantum processor LATEX Mathematical Symbols Ansatze and Variational Forms Greek letters used in mathematics, science, and engineering Finding the Laplacian of the deflection potential: I obtain $4\kappa(\vec{\theta})$, not $2\kappa(\vec{\theta})$. Why am I wrong? Extensible \vec instead of \overrightarrow Braket notation in LaTeX

15/11/2024

Tidied up files to submit everything that is not the disseration for the corpus, deadline is tomorrow. Checked with Carlos regarding the corpus who advised not to worry too much as the project is theoretical work:

"The corpus is essential in an engineering project, where you're building something (say a java app). Here it is more about the theoretical results. I really want to hear your conclusions about the pros/cons of both quantum UML approaches."

Links:

How to create Class Diagram using Mermaid-js Full Support of Mermaid Syntax Class diagrams Markdown Support PyCharm's Markdown rendering is beautiful. Is there a way to save it into a pdf or html file as is?

18/11/2024

I've completed the explenations on the two papers and how they are applied to each of the diagrams. I can now move on with dedicating the next week to the evaulation of the project, allowing the final week for final adjustements, edits and inclusion of figures.

20/11/2024

Attended Qiskit Club, discussion regarding peers proposals.

Added figures and introductions to sections. Now need to focus on Results section. Also need to remember to include breakdown of sections around the introduction.

21/22/2024

Fixed the script to compile and publish the disseration on github.

Startes results and analysis, completed "author observations" and tomorrow can focus on applying digrams to a "good modelling language" checklist - sources in the UML @Classroom can provide this.

22/11/2024

Working on results and analysis, realised I'd made a mistake with my classification of the minimize class for QUML, had to go through diss and diagrams to update so correct. Will post emails I sent to Carlos as explanation:

Email 1

Hi Carlos,

I have a question about the Q-UML core principles, and I would like to make sure I've understood them correctly in the context of IBM's VQE Qiskit tutorial.

The classical optimiser takes the user-defined cost function as a parameter. The cost function contains the code that will execute the estimation of the quantum circuit and handle quantum information. I've correctly defined the cost function as a quantum element.

I need to confirm whether the classical optimiser would also be considered quantum. So far, I have classed it as quantum. Following the Q-UML design principles, I believe that because it contains the quantum cost function in its parameters, it is "upgraded" to a quantum element following the Quantum Supremacy design principle. I doubt myself; however, as the input and output for the classical optimiser, all remain as classical information.

I've attached the class diagram using the Q-UML notation, which shows that the cost function has a shared aggregation relationship with the classical optimiser. Even if it might not fall under Quantum Supremacy, possibly Quantum Aggregation as the classical optimiser is composed of a quantum object having the cost function passed as a parameter.

I've defined the design principles of quantum supremacy and quantum aggregation as follows:

Quantum Supremacy: If an object does not use any quantum information for its design, interactions or relationships with other objects, it will always remain classical. It will be upgraded to quantum when it requires even one quantum element.

Quantum Aggregation: An object composed of at least one other quantum object will be labelled as quantum.

I appreciate you may be busy (and it's a Friday evening!), but I would appreciate it if you could advise as soon as possible. If it should not be considered quantum, I must make considerable changes to my work.

Many thanks,

Email 2

Hi again Carlos,

Just to follow this up, I believe I was wrong in classifying the classical optimiser as quantum (which sounds obvious writing it out!). I am following up on the previous email with my own explanation, having gone over things.

The classical optimiser having quantum elements passed to it in its parameters does not constitute it being composed of quantum elements- failing Quantum Aggregation design principle.

The information it received from the cost function is classical and the information it gives to the parameters also remains classical. It does not require any quantum information for its design, interactions or relationships with other elements- failing Quantum Supremacy.

Kind Regards,

Elly

25/11/2024

Working on the model quality section, have some good links regarding other non-UML models for quantum, worth mentioning in conclusion as I dont believe objective of finding out which is best model can be fully achieved without looking at non-UML models.

26/11/2024

Read through the diss and made some edits. Have drafted very rough notes for the conclusion and future work, need to refine this to complete the diss! Then make the video!

Links:

The pragmatics of model-driven development The pragmatics of model-driven development citations TOSCA4QC: Two Modeling Styles for TOSCA to Automate the Deployment and Orchestration of Quantum Applications A Graph-Based Approach for Modelling Quantum Circuits MDD Model-Driven Engineering Chapter 3 - Advances in Model-Driven Security

27/11/2024

Completed the diss! Wahoo! Just need to make the video and then I am done!

28/11/2024

Refined disseration and biblogrpahy, ensure correct formatting and source more suitable references where nessecary.

30/11/2024

Finished the video! Have some links from additional quantum modelling papers, worth going over to make sure correct that there arent code implemenations in UML to compare to. Also authos of q uml profile have already written paper on developing model driven design, need to mention this. KDM to UML Model Transformation for Quantum Software Modernization Generation of Classical-Quantum Code from UML models Transforming Quantum Programs in Kdm to Quantum Design Models in Uml Software modernization to embrace quantum technology Modeling of Parallel Quantum Key Distribution System via UML Software architecture for quantum computing systems — A systematic review QUANTUM SOFTWARE DEVELOPMENT: A SURVEY Quantum Software Engineering Landscapes and Horizons Unraveling quantum computing system architectures: An extensive survey of cutting-edge paradigms Modeling Quantum programs: challenges, initial results, and research directions

02/12/2024

Made some final adjustments to the diss and video, including some detailed investigation into other quantum modelling papers.

Topics to Research

- Mermaid
- NQCC
- Qiskit
- Blockchain
- UML - differentiate string and integer?
- Q-Cosmic
- Quantum State Preparation
- Quantum Measurement
- Grover’s Algorithm
- Shor’s Algorithm
- Quantum Probabilities (Probability Amplitude)
- Euler’s Formula
- Quantum Polarising Filters
- Systemisation of Knowledge (SoK) papers
- The New Scientist
- Quantum Computing for Computer Scientists
- Quantum Computation and Quantum Information
- DBLP
- PUML
- Miro
- Microsoft Quantum
- Cirq
- VQE
- QAOA
- VQF

To-Do List

- ~~Use a UML package to gain practical understanding of how it's implemented. Which classes of UML will QUML be relevant to?~~
- ~~Use Qiskit~~
- ~~Read the references from Carlos papers.~~
- ~~I need to confirm whether the dissertation and corpus are one unified document or separate documents. To better understand the process of writing a good dissertation, it would be useful to obtain books on scientific or academic writing.~~
- ~~Define specific research questions with Carlos.~~

Evaluation

How I approached this piece of work

I started by trying to understand everything related to quantum and UML, which was overwhelming; however, this research and development stage resulted in me finding the quantum UML profile paper. Once I established the purpose of the paper, comparing Q-UML to this other paper, I could simplify and break down the steps needed to do this. Establishing what tool I would use to create the diagrams, what example I would base them on (VQE) and what diagram I would start with, I could focus on completing tasks that would produce a minimum of one diagram type in the two quantum UML adaptions. From here, I was then able to include an additional diagram type and, through the dissertation writing, go over all of the content explored in the R&D stage and expand on relevant portions to create a dissertation with an overall explanation of the historical background of the topics related to the diagrams (papers, Q-SE, UML, NISQ, VQE) then follow this up with my implementation, reasonings behind it and reflections after the fact.

What I found easy

Discussing the details of the UML diagrams.

What I found difficult

I found explaining the VQE algorithm in a manner that I and a reader unfamiliar with the topic could understand was difficult, particularly the pseudocode.

If I were to do it again, I would do the following differently

I would have spent less time trying to learn linear algebra and having a broad but limited understanding of all UML diagram types. I would have picked one type of diagram and sought a generic hybrid or quantum algorithm, not a code excerpt, to model using UML. Then, we move on to either quantum UML adaptations or another type of modelling.

What I learned is

There have been more attempts than just UML to model quantum technologies, and if time permits, these would have been worth exploring. One of the overarching questions during the project was how many professionals utilise UML for their work. I have learnt how to construct a sequence and class diagram in UML and adapt it to the two quantum UML paper methodologies. I gained insight into diagram modelling tools and their strengths and weaknesses. I also have a more solid understanding of quantum computing and NISQ devices; I still need help with mathematical and algebraic information on quantum computing, but I feel more confident in explaining what it is on a basic level.

I would like specific feedback on

I hope to publish this paper, so I'd like feedback specifically on any areas I can improve to achieve this goal.