Design and Analysis of Quantum Algorithms

Keshan
Department of Computer Science and Engineering
Jaypee Institute of Information and Technology, Noida, Uttar Pradesh, India
Email: 21303009@mail.jiit.ac.in

Prof. Vikas Saxena
Head, Department of Computer Science and Engineering
Jaypee Institute of Information and Technology, Noida, Uttar Pradesh, India
Email: vikas.saxena@mail.jiit.ac.in

Abstract – Many researchers and industry experts are succeeding to create innovative software frameworks to automate the generation of the desired quantum algorithms but this requires an expert to give inputs in the form of high level codes. Some experts are developing automated software frameworks that accept user inputs in the form of text, image, voice etc. to generate programming codes and novel designs but these software frameworks don't solve complex problems that require the generation of appropriate quantum algorithms to act automatically on the user inputs.

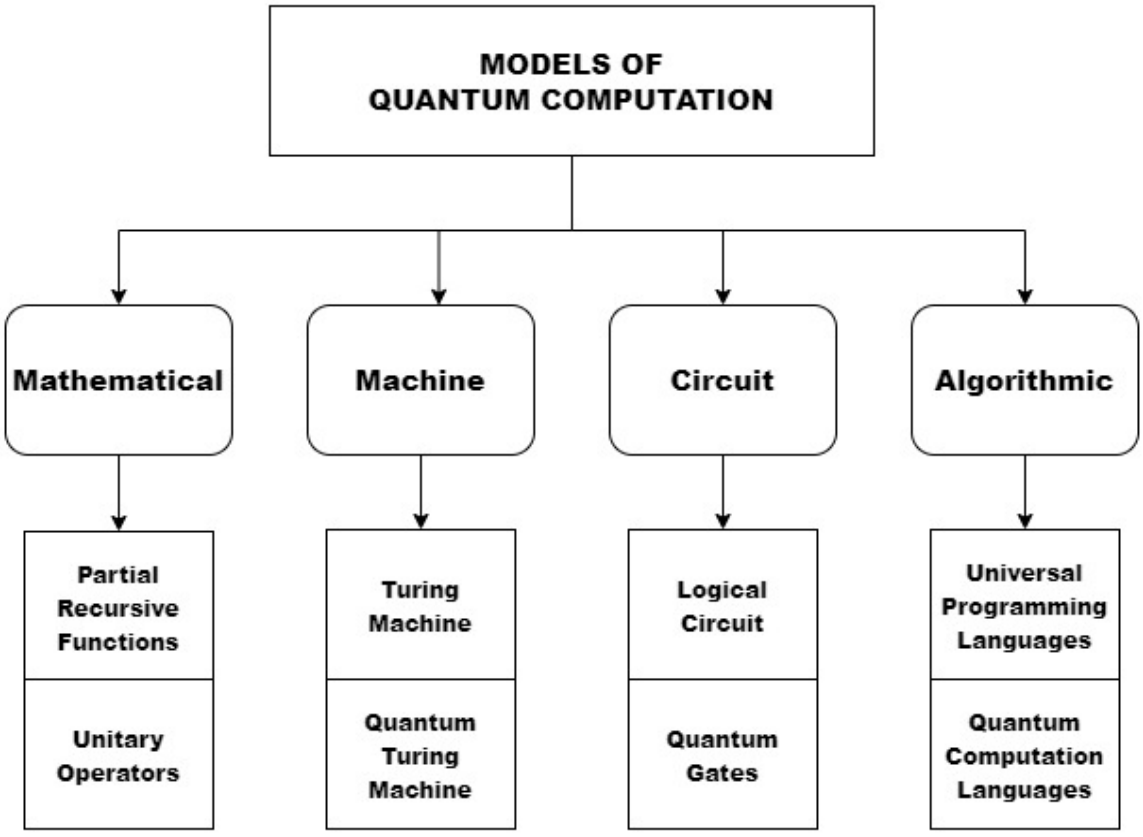
My goal is to identify new ways to create an automated software framework so that any user may give her/his inputs in the form of voice, video, image, text, touch, thought etc. that will also generate the required quantum algorithms automatically to solve complex problems. This requires a fundamental change in our understanding of data driven automated generation of quantum algorithms for **user** centric automation of everything. To achieve such goals I need to **(1)** identify the role of models, paradigms, classes, hardware specifications etc. for the evolution of quantum algorithms **(2)** analyze the concepts of existing quantum algorithms and **(3)** compare the technologies of emerging quantum software companies.

Keywords – Quantum Algorithms, QRL, QAO, QNN, QGAN

1. Introduction – Evolution of Quantum Algorithms

1.1 Limitations of Classical Computing: Classical algorithms based on Turing model of computation could not solve complex problems in real time due to the limitations of complexity theory. Classical algorithms based on functional and logic programming models of computation could not be developed effectively till now, so we don't know the efficacy of these models. The experts focused to harness the principles of quantum mechanics like superposition, entanglement and measurement to explore the possibility of quantum computation and tried to create quantum algorithms faster than all the classical algorithms.

1.2 Competing models of quantum computation paved ways for creating new quantum algorithms:

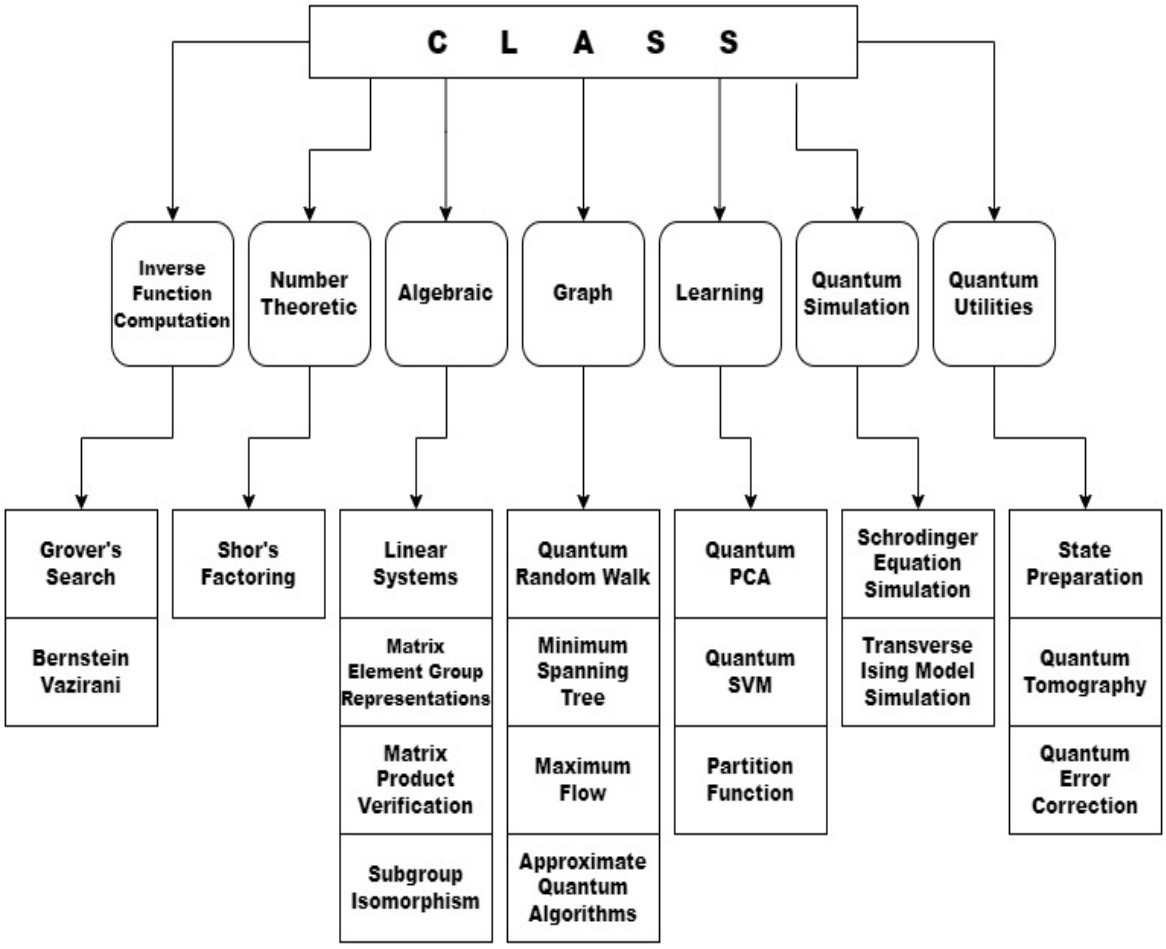


1.3 Paradigms of Quantum Algorithms: The majority of well-known quantum algorithms combine several quantum computing specific algorithmic paradigms.

- Quantum Fourier Transform (QFT)
- Grover Operator (GO)
- Harrow-Hassidim-Lloyd (HHL) method for linear systems
- Variational Quantum Eigenvalue Solver (VQE)
- Direct Hamiltonian Simulation (SIM)

Since it is challenging to create quantum paradigms from existing conventional paradigms due to the unitarity constraint on quantum processes and the inability of non-intrusive measurement, so innovations in philosophical and conceptual foundations will pave a way for holistic paradigms to create better and better quantum algorithms.

1.4 Classes of Quantum Algorithms:



1.5 Criteria to Implement Quantum Algorithms – DiVincenzo's Criteria:

I. Physical system that scales well and has well-defined qubits: It requires a physical system with a set of qubits. Generic state – $x|0\rangle + y|1\rangle$; x, y are complex numbers and $|0\rangle$, $|1\rangle$ are two states. A "well characterized" qubit means different things like: physical parameters ought to be precisely understood; the existence of alternative qubit states and couplings to them; interactions between qubits; and the connections to outside field that could be used to change a qubit's state.

II. Initializing the qubit's state to a trustworthy state: There are two reasons for this computing requirement: right before the start of computation the register initialization should be done and there is a requirement for fresh & continuous qubit supply in a concentrated state for quantum error correction. Two approaches for standardizing the state of qubits are: when the system's Hamiltonian's ground state is in the desired condition, it may either be 'naturally' cooled, or a measurement has the ability to rotate the system into the intended condition or into another state.

III. Long relevant decoherence times: Decoherence is essential for comprehending the fundamentals of quantum physics because it is the main mechanism for the emergence of classical behavior. For the same reason, decoherence poses a serious threat to quantum computing as its capabilities won't be any different from classical systems if it lasts for a very long time. Decoherence time needs to be sufficient for the distinctively quantum characteristics of this type of computation to potentially take effect. The outcomes of quantum error correction also provide information about how long is 'long enough.'

IV. A 'universal' collection of quantum gates: It is impossible to fully implement quantum gates. Systematic and random faults are expected in the application of the required Hamiltonians. If the mistakes are small enough, error correction techniques can be used to create trustworthy calculations from unstable gates. The errors indicated above can be seen as another form of decoherence. Gate operations must be performed on coded qubits in order to repair errors, raising the concern that the base-level qubits that make up the code will need a new set of simple gate operations.

V. Qubit specific measuring ability: The capability to measure specific qubits is a prerequisite for a quantum system.

The above mentioned requirements suffice for computation alone. But to gain the advantage of quantum information processing we need to look at two more requirements:

VI. Inter conversion of flying and stationary qubits.

VII. Reliably transmitting flying qubits between predetermined positions.

Quantum Key Distribution is a quantum communication method that includes the transfer of entangled qubits or coherent quantum states where interconverting and transmitting are required. A significant issue is the flying qubit's ability to be transmitted without decoherence.

The photons decoherence when interacting with airborne particles is now the main problem. Similar attempts have been made to employ optical fibers, however this hasn't been successful due to signal attenuation.

These two specifications are interdependent, yet they should be considered separately because some tasks need one but not the other. There are very few fully produced proposals that include requirements VI and VII. **Ex** – The **VII** requirement is sufficient for Quantum Cryptography.

2. Literature Review – A Critical Analysis of Existing Quantum Algorithms

2.1 David Deutsch created the 1st quantum algorithm in 1985 and later modified it with Richard Jozsa in 1992.

Problem: A hidden Boolean function is given that takes a bit string as input and either outputs 0 or 1. It is certain that this Boolean function will be either constant or balanced. We want to determine whether the given function is constant or balanced. For any input:

A function that outputs all zeroes or all ones is known as a constant function.	A function that outputs zeroes for exactly one half and ones for the other half is known as a balanced function.

Classical Solution: This solution deals with two cases, the best case and the worst case.

Only 2 queries can determine that f (x) is balanced. This is the best case. f (1, 1, 1 …) -> 1 and f (0, 1, 1 …) -> 0 will give two different outputs thus confirming that the given f (x) is balanced.
Total no. of possible inputs = 2^n, so for f (x) to be constant, $2^{n-1} + 1$ test inputs are needed. This is the worst case.

Quantum Solution: The mentioned problem can be resolved with a quantum computer in a single invocation to the function f (x). This necessitates the implementation of the function f as a quantum oracle. This algorithm involves 5 steps:

1. Get 2 quantum registers ready. 1^{st} register is initialized to $|0\rangle$ and 2^{nd} register is initialized to $|1\rangle$. The 1^{st} register is a n-qubit register while the 2^{nd} register is a one-qubit register.
2. A Hadamard gate is applied to all the qubits.
3. Quantum oracle is applied.
4. A Hadamard gate is applied to each qubit in the 1^{st} register. The 2^{nd} single qubit register is ignored for this activity.
5. Measure the 1^{st} register.

Deutsch-Jozsa algorithm is one of the earliest instances of a quantum algorithm which is exponentially quicker than any potential classical algorithm (deterministic). While the best classical algorithm solves the Deutsch-Jozsa problem in $2^{n-1} + 1$ steps (worst case), its quantum counterpart solves the same problem in only one call to the function. Although this algorithm still has very little practical use yet it has motivated to create better & better quantum algorithms.

2.2 The 2^{nd} quantum algorithm was created by Bernstein-Vazirani in 1993.

Problem: A black-box function f is given that takes a bit string as input and either outputs 0 or 1. It is certain that this black-box function will return some string s. We can say that f (x) = sx % 2. We want to determine s.

Classical Solution: The oracle will return f_s (x) = sx % 2 on input x. For revealing the hidden bit string s we need to query the oracle with following sequence of inputs: 100…0 then 010…0 then 001…0 and so on, where each search will reveal a distinct bit of s.
Ex – If x = 100…0 then we can get the s bit that is least significant. This way the next least significant bit can be found and so on. So, the given function f_s (x) is called n times.

Quantum Solution: The mentioned problem can be resolved with a quantum computer in a single invocation to the function f (x). This algorithm involves 5 steps:

1. The input qubits are initialized to the $|0\rangle^{\otimes n}$ state and the output qubit to $|-\rangle$.
2. Hadamard gates are applied to the n-qubit input register.
3. The Oracle is queried.
4. Again apply Hadamard gates to the n-qubit input register.
5. Measure the 1st register.

Bernstein-Vazirani algorithm is an expansion of the Deutsch-Jozsa algorithm. While the best classical algorithm solves the Bernstein-Vazirani problem in n steps, its quantum counterpart solves the same problem in only one call to the function. One major difference between Bernstein-Vazirani and Deutsch-Jozsa lies in the application of oracle function. Deutsch-Jozsa focuses on determining whether the function is balanced or constant while Bernstein-Vazirani focuses on determining hidden string s. All other steps are same in both the algorithms.

2.3 The 3rd quantum algorithm was created by Simon in 1994.

Problem: A black-box function f is given. It is certain that this black-box function is either 1 to 1 or 2 to 1. We want to find whether f is 1:1 or 2:1.

1:1 – maps exactly 1 unique output for every input. Ex – f (5) -> 5, f (7) -> 7, f (9) -> 9.
2:1 – maps exactly 2 inputs to every unique output. Ex – f (3) -> 1, f (5) -> 2, f (7) -> 1, f (9) -> 2.

If f is 2:1 then how soon b can be determined? So, both the cases lead to the same issue of determining b. Mapping of 2:1 is according to a hidden bit string b where x_1, x_2 are given such that f (x_1) = f (x_2), so it is certain that $x_1 \oplus x_2 = b$.

Classical Solution: For finding b we need to check up to $2^{n-1} + 1$ inputs until we locate two instances with the same result. It is a matter of chance that the problem is solved with our first two tries (like Deutsch-Jozsa problem).

Quantum Solution: This algorithm involves 6 steps:

1. The zero state is used to initialize two n-qubit input registers.
2. A Hadamard gate is applied to the 1st n-qubit register.
3. Query function is applied.
4. Measure the 2nd register. A specific f (x) value is noted.
5. A Hadamard gate is applied to the 1st n-qubit register.
6. Measure the 1st register.

This algorithm is based on the model of query complexity (outcome of previous tests influencing next tests). While the best classical algorithm solves the Simon's problem in $2^{n-1} + 1$ steps (worst case), its quantum counterpart solves the same problem in only one call to the function. Simon's algorithm differs from Deutsch-Jozsa and Bernstein-Vazirani in one aspect: Simon's algorithm measures the second register and then measures the first register while Deutsch-Jozsa and Bernstein-Vazirani don't measure the second register at all they only measure the first register.

The above 3 '**Proof of Concept**' quantum algorithms paved a way for establishing quantum supremacy. This quantum advantage encouraged everyone to build new and better quantum algorithms for different applications to create the idea of quantum simulation and to build cloud based quantum hardware.

2.4 Peter Shor created a quantum algorithm in 1994 to determine an integer's prime factors. The basic idea of this algorithm is as follows:

Problem: Prime factors of an integer. The periodic function is defined as: $f(x) = a^x \% n$ (a less than n), with no common factors. The period (r) is defined as: $a^r \% n = 1$.

Solution: It consists of 2 parts – 1st part changes the issue from finding prime factors of an integer to period finding of the function. 2nd part deals with finding the period using Quantum Fourier Transform. This algorithm involves 5 steps:

1. The gcd (j, k) of an arbitrarily chosen positive integer $j < k$ is determined.
2. The sequence's unidentified period R is deduced using a quantum computer:
$$x \% n, x^2 \% n, x^3 \% n \ldots$$
3. If the number R is odd, then repeat 1st step. Move on to the following step if R is even.
4. Check $(j^{R/2} - 1)(j^{R/2} + 1) = (j^R - 1) = 0 \% n$. If $(j^{R/2} + 1) = 0 \% n$ then repeat 1st step. If $(j^{R/2} + 1) \neq 0 \% n$ then move to next step.
5. Finally $d = [\gcd(j^{R/2} - 1, n), \gcd(j^{R/2} + 1, n)]$ is calculated. This will give us the prime factors of n.

This algorithm is a part of number-theoretic applications class and Quantum Fourier Transform (QFT) paradigm. This algorithm's effectiveness is a result of the effectiveness of the QFT and modular exponentiation. Shor's algorithm outperforms the most effective classical factoring algorithm by being almost exponentially faster. For long it was assumed that RSA is computationally intractable but Shor's algorithm instilled a belief to design and construct powerful quantum computers which may be able to break RSA encryption. This prompted researchers to examine new quantum algorithms. Finally, this created a foundation for post-quantum cryptography (a new generation of quantum computer-resistant cryptosystems).

2.5 Lov Grover created a quantum algorithm in 1996 to solve the problem of unstructured search. The basic idea of this algorithm is as follows:

Problem: Unstructured Search. Given a large list of N items. This problem can be expressed with an abstract function f (x). We want to locate the winner w.

Solution: Total possible inputs $N = 2^n$. This algorithm involves 5 steps:

1. Input registers are first initialized to the state $|0\rangle$.
2. A Hadamard gate is applied to every qubit register.
3. Apply following operations to the register $N_{optimal}$ time:
   a. The oracle applies a conditional phase change of -1.
   b. A Hadamard gate is applied to every qubit register.
   c. In every state a phase change of -1 is done. The state $|0\rangle$ is excluded from this activity.
   d. A Hadamard gate is applied to each qubit register.
4. The register is measured. This will give us the sign of a solution that has a very high probability.
5. Checking solution validity.

This algorithm is part of inverse function computation class and Grover Operator paradigm. Most of the quantum algorithms provide an exponential speedup compared to their traditional equivalents while Grover's approach only accelerates by a quadratic factor. However, when N is big, this quadratic acceleration is quite good. Grover's algorithm deals with the task of function inversion. Grover's algorithm sets apart from other algorithms in a way that – for a number of different algorithms it can produce quadratic run-time improvements.

2.6 Quantum Reinforcement Learning (QRL) vs Quantum Approximate Optimization (QAO):

QRL is a combination of traditional reinforcement learning and quantum theory. Traditional reinforcement learning is based on three fundamental principles, namely, 'a policy', 'a reward function' and 'a model of the environment'. So, these three principles can be established for QRL too but QRL differs from traditional RL on the basis of following four characteristics: 'representation', 'policy', 'parallelism' and 'updating operation'.

QAO is an instance of a heuristic algorithm that solves combinatorial optimization problems on NISQ devices. Much like its classical counterpart QAOA tries to find a classical bit string (with high probability) that is expected to have a good approximation ratio.

There are two major problems in the traditional reinforcement learning algorithm and traditional approximate optimization algorithm:

- **Exploration strategy:** balancing the trade-off between exploration and exploitation.
- **Slow learning speed**

Quantum version of these algorithms overcomes above two problems in the following manner:

- In addition to accelerating learning and ensuring searching across the whole state-action space, the action selection policy achieves a good trade-off between exploration and exploitation by making use of probability.
- The superposition principle underlies representation using eigen-states or eigen-actions. State value updation and probability amplitude updation is made possible through quantum parallelism. This parallelism results in enhanced performance of QRL in simulated experiments when state space is large.

2.7 Quantum Neural Networks (QNN) vs Quantum Generative Adversarial Networks (QGAN):

QNN is a union of artificial neural networks and quantum computing. The difficulty of training traditional neural networks is one of the main motivation behind the development of QNN.

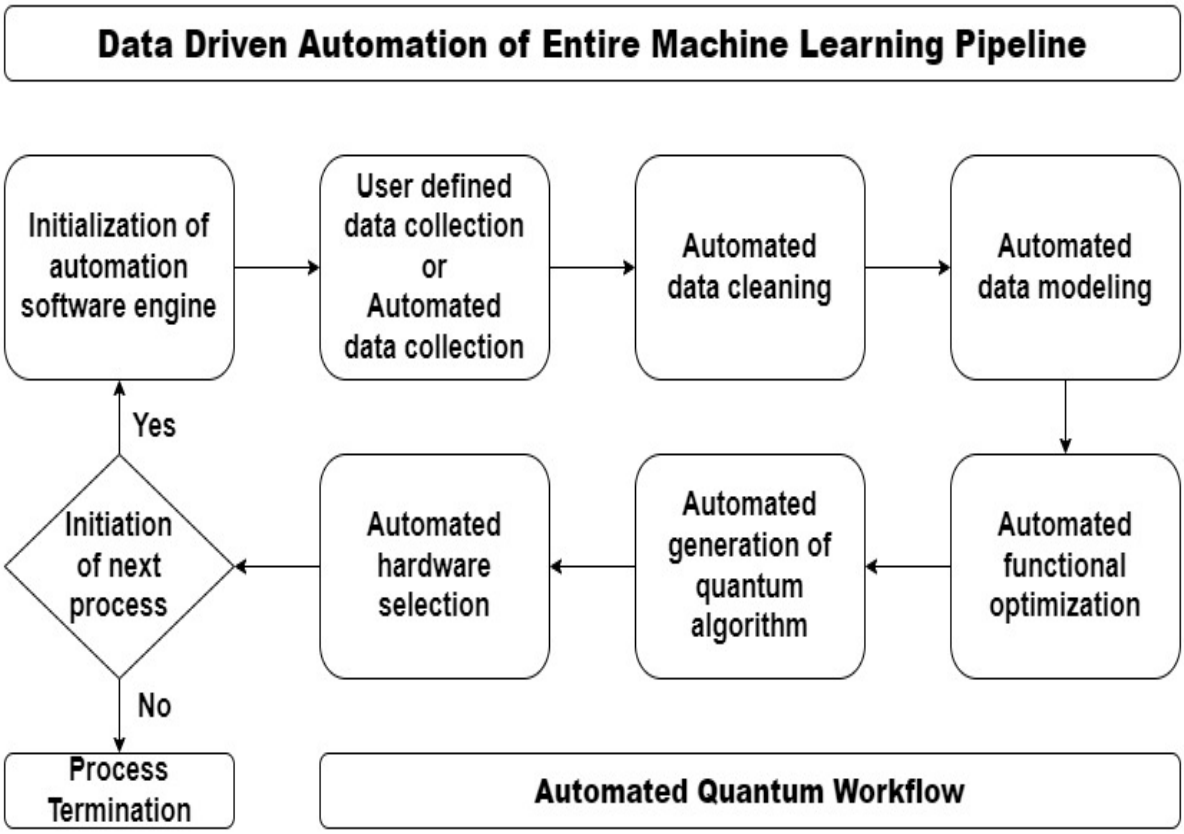
QGAN is a combination of quantum computing and generative adversarial networks. A generator and a discriminator are the two neural networks that are trained at the same time. This training is done via some learning strategy. GANs make use of a classical generator and a classical discriminator while QGANs make use of a quantum generator and a classical discriminator.

Major problems in neural networks and generative adversarial networks:

- Vanishing Gradient
- Local Minima
- Learning inefficiencies

Variational Quantum Circuits (VQC) carry out numerous numerical tasks like classification, approximation and optimization that assist in overcoming the above mentioned limitations. VQC depends on free parameters. When it comes to approximating functions via parameter learning, VQC is comparable to ANN but it also differs in some ways. Entanglement layers are used in quantum circuits rather than activation functions for multilayer architectures because we now understand that quantum gate operations are reversible linear processes.

3. Data Driven Automation of Entire Machine Learning Pipeline



3.1 Automated Data Collection:

Quantum Sensors work on the principle of quantum entanglement, quantum interference and quantum state squeezing with optimized precision and beat the current limits of sensor technology. Quantum sensing is a technological field which deals with the design and engineering of quantum sources and quantum measurements that can beat any classical strategy in various applications. Quantum Sensors are often built on continuously variable systems (quantum systems characterized by continuous degrees of freedom such as position & momentum). Basic working mechanism of quantum sensor typically relies on optical states of light (involving mechanical properties such as squeezing or two-mode entanglement).

3.2 Automated Data Cleaning:

Using machine learning, the data is modified or removed if it is incorrectly formatted, incorrect, incomplete, duplicated or flawed. On an estimate, manual data cleaning consumes almost 90% of the time spent in the machine learning life cycle. This is a serious cause of concern as all the focus is fixated on making the data conducive rather than on maximizing the potential of other parts of machine learning i.e. algorithm selection / generation, hyper parameter optimization / tuning, algorithm evaluation, data specific hardware selection etc.
Now we need to answer an important question: Which steps of data cleaning can be actually standardized and automated? The answer to this question is: the steps which are performed frequently (time and again) are most likely to be automated or standardized.
Next, we want to ensure that our pipeline is generic and can adapt to various types of datasets.

1. CSV / JSON – There are five types of building blocks for this type of data. These are processed as follows:

Missing Values (Block 1): There are two ways to deal with missing values, we either delete those observations that contains missing values or we can simply use an imputation technique. An imputation technique replaces missing data by certain values, like mean, median, mode, or

by a value similar to other values of the sample. Classification or Regression models are quite useful in prediction of missing values in our data.

To handle the missing values, we primarily focus on:

- the data type (numerical or categorical)
- count of missing values relative to the count of total samples

We will adopt the strategy to give more preference to imputation than deletion. We will be focusing on following strategies: prediction with linear and logistic regression and imputation & deletion with mean, median and mode as well as K-NN.

Class Structure of Missing Values

```
class MissingValues:
    def handle (miss, missing_num = 'auto', missing_categ = 'auto', _n_neighbors = 5):
        ...
    def_impute (miss, imputer, type):
        ...
    def_lin_regression_impute (miss, model):
        ...
    def_log_regression_impute (miss, model):
        ...
    def_delete (miss, type):
        ...
```

The handle function will handle numerical and categorical missing values in the following way: some imputation techniques are applicable only for numerical data while some are applicable only for categorical data. Handle function firstly checks for the handling method which is by default set to 'auto' indicating:

If the missing data is of numerical type then the data is either deleted or imputed through

- linear regression or
- knn or
- mean, median or mode

If the missing data is of categorical type then the data is either deleted or imputed through

- logistic regression or
- knn or
- mode

It is to be noted that the categorical data will be first label encoded to integers to be able to predict the missing values and finally the labels must be mapped back to their original values.

The impute function will impute the missing values in the data through knn, mean, median and mode while the linear and logistic regression function will impute the missing values through prediction. Lastly, the delete function will delete the missing values in the data.

Outliers (Block 2): Outliers are the extreme values in a dataset. Firstly, we need to finalize a criteria to label a value to be an outlier. We will consider a value or a data point to be an outlier if it does not fall inside the following range:

1^{st} quartile – 1.5 * interquartile range; 3^{rd} quartile + 1.5 * interquartile range

Much like the missing value problem, there are two ways to deal with outliers namely, winsorization and deletion. Winsorization is a statistical technique used to limit extreme values in the data to limit the effect of the outliers. In addition, outliers are replaced with a specified percentile of the data. Outliers will be replaced using above defined range through winsorization:

- upper range value will replace values > upper bound and
- lower range value will replace values < lower bound

Class Structure of Outliers

class Outliers:
def handle (out, outliers = 'winso'):
...
def_winsorization (out):
...
def_delete (out):
...
def_bounds (out, feature):
...

The handle function will handle the outliers in the data in two ways: winsorization or deletion. The winsoriation function will compute the upper bound and lower bound and check for the two conditions mentioned above. Whichever conditions matches, that value will replace the respective outlier with an appropriate value. The delete function will delete the outliers in the data. Lastly, the compute bounds function will compute the upper and lower boundaries for identifying data outliers.

Categorical Encoding (Block 3): There are no problems while handling numerical type data but categorical data first needs to be changed into numerical data to perform any computations. One-hot encoding and label encoding are two popular methods that can be used to get numerical data from categorical data.

Label encoding: replace a categorical value with a numerical value ranging between 0 and total classes – 1. Label encoding assigns a unique integer to each value. One of its drawback is that algorithms may interpret the numerical numbers as having some sort of hierarchical order in them. A contemporary approach of one-hot encoding is present to address this drawback.

One-hot encoding: replace a categorical value with a binary (0 or 1) value. Although, this approach solves the problem of order / hierarchy but it has its own disadvantage of adding more columns to the data set (if many unique values are present).

Categorical encoding will be done keeping in mind following 3 rules (with the default strategy set to 'auto') that are:

1. If the feature set has less than 10 distinct values then one-hot encoding is used.	2. If the feature set has less than 20 distinct values then label encoding is used.
3. If the feature set has greater than 20 distinct values then no encoding is done.	

Class Structure of Categorical Encoding

```
class EncodeCateg:
    def handle (cat, encode_categ = ['auto']):
        ...
    def_to_onehot (cat, feature, limit ≤ 10):
        ...
    def_to_label (cat, feature):
        ...
```

Extraction of DateTime features (Block 4): Extraction of features from the dataset that have datetime values becomes easier to handle while processing or visualizing them later. Timestamps or dates are some of these datetime values. This can be done automatically by searching through the features with the help of pipeline and checking whether conversion into the datetime type is possible for any one of them.

Class Structure of DateTime Features

```
def handle (cat, encode_categ = ['auto']):
    ...
def_to_onehot (cat, feature, limit ≤ 10):
    ...
```

By default, the granularity is set to ‘s’ for seconds. The granularity can be customized too to extract the datetime features. After the extraction, the date and time entries are checked for correctness by the function. If the extracted columns namely ‘Day’, ‘Month’ and ‘Year’ contain 0’s then all three of them will be deleted. Same goes for ‘Hour’, ‘Minute’ and ‘Sec’.

Data frame Polishing (Block 5): Although, the dataset is processed, some more modifications are still needed to make the data frame ‘look good’. Firstly, some features that were initially of type integer may have been transformed to floats through imputation techniques or other processing procedures, so before outputting the final data frame these values will be converted back to integers. Secondly, all float values will be rounded to same number of decimals. This will ensure that float decimals are free from unnecessary trailing 0’s and the values are not rounded greater than the original values.

2. Text Data – It is necessary to clean raw text before using it for Natural Language Processing. Clean text is just plain old human speech that computers can understand. Text can be cleaned up with a few lines of straightforward Python code that eliminates stop words and separates complex words into their component parts.

Normalizing text: The method of normalizing text involves standardizing text so that, through NLP, human input is better understandable to computer models. This will result in effectively performing sentiment analysis. Text normalization through NLTK library involves standardizing capitalization so that capitalized words are not grouped together by machine models. This way lowercase and uppercase words will be segregated.

```
In [1]: text = "DESIGN AND ANALYSIS OF QUANTUM ALGORITHMS"

        text = text.lower()

        print(text)

        design and analysis of quantum algorithms
```

Removing Unicode characters: URL, Emoji, Punctuation and @ symbols are distinctive signatures that are either unique or result in incorrect Unicode translation, which causes AI models to become confused. Punctuation adds noise and hinders NLP comprehension because instead of just affecting the term it is related with, it changes the sentence tone as a whole.

```
In [2]: import re

        text = "Design and Analysis of Quantum Algorithms! https://qiskit.org/textbook/ch-algorithms/"

        text = re.sub(r"(@\[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)|^rt|http.+?", "", text)

        print(text)

Design and Analysis of Quantum Algorithms
```

Removing Stop words: Stop words are recurrent within sentences that don't provide any value and can be eliminated during NLP cleaning before analysis. In English and numerous other widely spoken languages, stop words are frequently seen.

```
In [3]: import nltk.corpus
        nltk.download('stopwords')
        from nltk.corpus import stopwords

        stop = stopwords.words('english')
        text = "Stop words are recurrent within sentences that don't provide any value"
        text = " ".join(\
                        [word for word in text.split() if word not in (stop)])

        print(text)

Stop words recurrent within sentences don't provide value
```

Stemming and Lemmatization: It involves breaking down words to expose their underlying meanings and connections. We can more accurately gauge purpose if we do this. It is crucial to choose the right technique for the analysis we want to undertake because, despite the similarities between the two techniques, they yield very different outcomes. **Stemming** arranges words according to their root stem. **Lemmatization** allows us to discern between the past, present and indefinite by organizing words based on the root definition.

In [4]:
```
import nltk
from nltk.stem.porter import PorterStemmer
from nltk.stem import WordNetLemmatizer
words = \
["play", "played", "plays", "playing"]
stemmer = PorterStemmer()
for word in words:

    print(word + " = " + stemmer.stem(word))
```

```
play = play
played = play
plays = play
playing = play
```

In [5]:
```
import nltk
from nltk.stem.porter import PorterStemmer
from nltk.stem import WordNetLemmatizer
words = \
["play", "played", "plays", "playing"]
lemmatizer = WordNetLemmatizer()
for word in words:

    print(word + " = " + lemmatizer.lemmatize(word))
```

```
play = play
played = played
plays = play
playing = playing
```

3.3 Automated Data Modeling:

PYOMO stands for Python Optimization Modeling Objects. It is a modeling tool based on Python programming language. It is used for model optimization. It comes under the umbrella of Algebraic Modeling Languages (AMLs).

An abstract (or symbolic) mathematical model is defined using symbols representing data values. Abstract models rely on unspecified parameter values. A model instance can be specified using data values.

A concrete mathematical model is defined directly with data values that were assigned when the model was defined.

Variables: A numerical value determined during optimization. The search space for optimization is defined by the variables.

Pyomo variables are managed with the Var class, which can denote a single, independent value, or an array of values. Variables can have initial values, and the value of a variable can be retrieved and set.

Declarartion of a single variable: model.x = Var ()

Declaration of multi-dimensional variable arrays:

```
J = [1.7, 2.7, 3.7]
model.u = Var (J)
model.K = Set ()
model.t = Var (J, model.K)
```

Objectives: An objective is a function. This function involves data and variables which is either maximized or minimized through a solver. The solver looks for the variable's values

that provide the best possible value an objective function can have. Although, constraints may limit the possible values for the variables, the objective function dictates how the variable's values will be assessed.

Declaration of a single objective: model.m = Obj ()

Declaration of multiple objectives:

model.j = Obj ()
model.k = Obj ([1,2,3])

Constraints: Expressions that place a limit on the values of variables are defined by constraints. Variable values can also be limited using the bounds option, however constraints broaden this concept by enabling expressions that limit a number of interacting variables at once.

Constraints expressions are declared just like the objective function expressions. However, there is one stark difference between constrains and objectives, that is, for specifying the equalities or inequalities some auxiliary information needs to be augmented to the expressions.

Another difference between constraints and objectives is in the terms of indexing. Constraints are frequently indexed, enabling access to variables and indexed parameters while creating the constraint expression. Although objectives can be indexed, this feature isn't often used.

Model Data: The two components namely set and param are used to construct constraints and objectives (for defining the data). By default, abstract data declarations are defined through set and param components.

Declarations for set X and parameter y (with no data):

model.X = Set (within = Reals)

model.y = Param (model.X, within = Integers)

Set component is not abstract if there is no specification of initialize option	Param component is not abstract if there is no specification of initialize option as well as default option
Set or Param component is abstract if indexing is done through an abstract Set component	

Data command file: A sequence of commands directly specifying set and parameter data or specifying external sources to obtain such data. Commands for data declaration:

- set command – set data.
- param command – parameter data table. It can also include the declaration of the set data used to index parameter data.
- table command – 2-dimensinal table of parameter data.
- import command – importing set and parameter data through external data sources. It includes ASCII table files, CSV files, ranges in spreadsheets and database tables.

Some more commands used in data command files:

- include command – specifying a data command file that needs to be processed immediately.
- data and end commands – they perform no action. Both the commands provide compatibility with AMPL (A Mathematical Programming Language) scripts to define data commands.
- namespace keyword – data commands are organized into named groups and during model construction these groups can be enabled or disabled.

Data commands:

- are terminated with a semicolon
- does not depend on whitespace

- can be broken over multiple lines
- newlines and tab characters are ignored
- formatted with whitespace (involving few restrictions)

The **set** command

Simple sets: set command defines the members of an indexed set, which might be an array of sets or a single set, explicitly. A list of the data values that make up this set are supplied as a single set. Syntax: set <setname> := [<value >] ... ;

Data values in a set:

- Numeric values – Python evaluated strings that take a numeric value (int, float, bool).
- Simple strings – alpha numeric character sequence.
- Quoted strings – strings inside a pair of ‘ ’ or “ ” quotes. Within the quoted string, quotations may be used.

A set declaration has no restrictions regarding values in a set. A set can be empty and it can contain different string combinations of numeric and non-numeric values. While constructing a model, set data validation is performed. When analyzing a data command file, set data is not validated.

The **param** command

There are two types of parameter data: one-dimensional and multi-dimensional.

1-dimensional: single set indexing.

2-dimensional: multiple set indexing or multi-dimensional indexing.

The **table** command

This command specifies explicitly a 2-dimensional array of parameter data. This command has some advantages over the param command in terms of flexibility and complete data declaration. Illustration for a single parameter data declaration is as follows:

table J (X) :			
X	Y	J	K
X1	Y1	2.6	3.6
X2	Y2	2.7	3.7
X3	Y3	2.8	3.8

The **import** command

This command allows to import data from various external tabular data sources. In a model, set and parameter data are represented through a relational table. Numeric string values matrix, simple strings and quoted strings are all part of a relational table. All rows and columns have the same lengths. Column data labels is represented by the first row.

External data sources that can be loaded through import command are:

- Tab file: text file.
- Csv file: text file.
- Xml file: tabular data.
- Excel file: spreadsheet data.
- Database: relational database.

The import command makes use of a data manager to extract data from a specified data source.

In this way, the import command provides a generic mechanism that enables the models to interact with standard data repositories that are maintained in an application-specific manner.

4. EMERGING APPLICATIONS

A. **Automated Full Stack Design And Development**
B. **Automated Education And Creative Development**
C. **Automated Healthcare and Human Engineering**
D. **Automated Social Engineering**
E. **Automated Nature Engineering**
F. **Automated Resource Generation And Planning**
G. **Automated Infrastructure and Services**
H. **Automated Agriculture and Regional Development**
I. **Automated Industrialization, Ecology and Biodiversity**
J. **Automated Space Habitat and Floating Cities**

Assuming the new automation pipeline:

How User Centric Automation Will Radically Transform Healthcare:

24 x 7 Online Monitoring: Now quantum sensors are succeeding to take the most accurate data from the remotest areas including the processes inside a human cell. These exploit the quantum principles of superposition and entanglement to collect minute changes in the electric and magnetic field of any quantum particle that has motivated the researchers to observe changes in the cellular processes. The idea is that the quantum sensor will capture even a small change in the cell when it gets infected or inflamed or its cellular process disrupts. This leads to important practical applications to observe changes in any single cell or organ or whole body just by wearing a ring made of quantum sensors that will continuously observe the changes in the electromagnetic field and biochemical changes in the body and will notify as soon as there is abrupt change in any parameter like Blood Pressure, SpO2, Heart Rate / Pulse, Stroke, Epilepsy, Behavioral Disorders, Infection (CoVid-19, HIV, STDs etc.), Radiation, Poisoning etc. This way any one will be continuously automatically monitored without any worries of succumbing to deadly diseases or any unforeseen circumstances etc. This technology is leading to revolution in

healthcare as immediate cure will stop the development of any disease at the very beginning.

Automated First-Aid: When there is an abrupt change in any health parameter or a sudden change in the biochemical composition of any cell, the quantum sensors in the wearable ring will immediately sense the problem at the root level (i.e. changes in the electric field, magnetic impulse, mechanical properties etc.) and will administer the required precautionary medicine from the ingestible chip besides sending notifications to the family members and the healthcare expert(s) e.g. when the first cell gets infected from any deadly virus like Corona or HIV or STD then the quantum sensors in the wearable ring will immediately detect abrupt changes in the cell and will instruct the ingestible chip to release the medicine to create a shield outside the infected cell to stop spread of the virus. Similarly, when any person gets exposed to radiation or poisoning or when (s)he faces abrupt changes in blood pressure, heart rate, sugar level etc. then the precautionary medicine will get released from the ingestible chip that will save her / him for 6-8 hours, so (s)he will find enough time to get proper healthcare. This way neither any disease will develop nor the patient will suddenly die.

On demand Diagnosis: When a person wants a detailed diagnosis of any health parameter or organ or whole body, then (s)he will swallow a capsule made of biomolecules, plants, and herbs etc. This capsule will contain the quantum sensor that will bind to the particular cell / organ to be diagnosed. Then, the patient will select the diagnose button from the ring that will connect the quantum states of the ring with that of the quantum sensor bound with the cell. Now the quantum sensor will send the visual data of that particular cell / organ to be diagnosed to the pipeline through the ring. The new inputs will generate the required quantum algorithm to produce the most accurate diagnosis.

Personalized Prescription and Production: Since continuous monitoring and on demand diagnosis will signal the development of any disease at the very beginning, there will be almost no need of medicines. Everybody will remain healthy and disease free by enjoying personalized food, lifestyle, nature care etc. Recommendations for detox & cellular

regeneration, personalized nutrition, medicine, yoga, aerobics, dance, sound sleep, electromagnetic field energy, hydro therapy etc. is a multi-objective optimization problem which involves various permutations & combinations in the pipeline to suggest according to the personalized needs and goals of the user. The portable health machines will produce only those items and services that will be finalized by the healthcare expert.

Intensive Care through Automated Bed at Home: Today large number of patients die due to delay in healthcare support but since the patient will be continuously monitored online in the new system, the automated bed at home will immediately provide emergency support like CPR, oxygen support, pre-decided medicine etc. to save life. This way the situation will be under control immediately than worsening due to delay in diagnosis and un-availability of the doctors. Since the online monitoring of even a quantum particle inside the cell will be visually analyzed, the impact of holistic medicines can be accessed in time. This will motivate the healthcare experts from various therapies to try and test the impact of combined medicines e.g. a proper combination of allopathic medicine, homeopathic medicine, nutrition, magnetic energy, pranic healing etc. If the administered medicines show any kind of negative effect then the visual data will help the healthcare experts to address those issues with appropriate changes (e.g. giving an anti-dote, improving the combination of various medicines or changing the medicine(s) altogether etc.) immediately.

Intensive Critical Care and Near Death Experience at Automated Health Center: Once the intensive care through automated bed at home fails, the patient will be shifted to the automated healthcare center where the most advanced technologies will try to save her / him. Now, some novel initiatives are being experimented to revive the dead person e.g. Dr Sam Parnia succeeds to bring the dead back to life. He argues that the chances of reviving the dead back to life will be more if we automate the following:

- Cardiopulmonary Resuscitation
- Targeted Temperature Management
- Extracorporeal Membrane Oxygenation

- Brain Oximetry
- Prevention of Reperfusion Injury

The automated healthcare center can also try personalized family therapy, ionized water therapy, vibration therapy, sound therapy, mantra therapy etc. Mrit Sanjeevni Vidya, Maha Mrityunjay Jaap, Tantrik Practices etc. can also be personalized by the pipeline and implemented with the most effective machines.

This new type of monitoring, diagnosis, personalized prescription will empower everyone to automatically keep track of health parameters anytime and reduce their dependence on infrastructure centric healthcare systems.

Similarly, **user centric automation** of all the other problems can be done.

5. Conclusion and Future Directions

Conclusion:

- I have proposed a new automation pipeline that will take inputs in the form of text, image, audio, video etc even from an **illiterate user**; will generate the **most efficient quantum algorithm** according to the requirements of the data; and will show the results **visually**.

- I have conceived to ground this new automation pipeline in a new development model valuing labor, creativity and computation to let any user earn 100 % profits from her / his online work.

- Emerging technology for user centric automation will transform everything and will promote the researchers at the helm of affairs.

Future Directions:

- More research is needed to understand the transitioning from Automated Data Cleaning to Automated Data Modeling.

- **Wearable Ring As Input-Output Device for Every Online Facility As A Service:** Since the ring will be just an input-output device to connect the quantum states for communication and control the instructions by the user, a new type of cloud hardware is required to process data from billions of rings.

- To explore emerging models of computation for a computational theory of everything.

References

[1]. Models of Quantum Computation, http://tph.tuwien.ac.at/~oemer/doc/quprog/node9.html

[2]. Quantum Computation Beyond the Circuit Model by Stephen Paul Jordan, 2008, https://arxiv.org/pdf/0809.2307.pdf

[3]. Quantum Algorithm Implementations for Beginners by Abhijith J. et. al, 2022, ACM Transactions on Quantum Computing, Volume 3, Issue 4, December 2022, Article No.: 18, pp 1–92, https://doi.org/10.1145/3517340

[4]. The Physical Implementation of Quantum Computation by David P. DiVincenzo, 2000, Volume 48, Issue 9-11, September 2000, https://doi.org/10.1002/1521-3978(200009)48:9/11<771::AID-PROP771>3.0.CO;2-E

[5]. A Brief Review of Quantum Algorithms, https://qiskit.org/textbook/preface.html, https://www.allaboutcircuits.com/technical-articles/fundamentals-of-quantum-computing/, https://learn.microsoft.com/en-us/azure/quantum/concepts-grovers

[6]. D. Dong, C. Chen, H. Li and T. -J. Tarn, "Quantum Reinforcement Learning," in *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 5, pp. 1207-1220, Oct. 2008, doi: 10.1109/TSMCB.2008.925743.

[7]. Quantum Approximate Optimization Algorithm, https://qiskit.org/textbook/ch-applications/qaoa.html, https://pennylane.ai/qml/demos/tutorial_qaoa_intro.html

[8]. Quantum Neural Networks by Yunseok Kwak, Won Joon Yun, Soyi Jung, and Joongheon Kim, 2021, Conference: 2021 Twelfth International Conference on Ubiquitous and Future Networks (ICUFN), doi:10.1109/ICUFN49451.2021.9528698

[9]. Quantum GAN, Zoufal, C., Lucchi, A. & Woerner, S. Quantum Generative Adversarial Networks for learning and loading random distributions. npj Quantum Inf **5**, 103 (2019), doi: https://doi.org/10.1038/s41534-019-0223-2;

[10]. Quantum Variational Circuits, https://pennylane.ai/qml/glossary/variational_circuit.html

[11]. Recent Advances for Quantum Neural Networks in Generative Learning, https://arxiv.org/pdf/2206.03066.pdf

[12]. Quantum Algorithms for Reinforcement Learning with a Generative Model, http://proceedings.mlr.press/v139/wang21w/wang21w.pdf

[13]. AutoML and AutoRL, https://www.automl.org/, https://medium.com/swlh/8-automl-libraries-to-automate-machine-learning-pipeline-3da0af08f636, https://arxiv.org/pdf/2201.03916.pdf

[14]. List of Awesome Papers in Quantum Machine Learning, https://github.com/artix41/awesome-quantum-ml

[15]. Classiq, https://www.classiq.io/

[16]. OpenAI and DALL E Mini, https://openai.com/

[17]. Blackbox, https://www.useblackbox.io/search

[18]. Minerva, https://ai.googleblog.com/2022/06/minerva-solving-quantitative-reasoning.html

[19]. Hart, William E., Jean-Paul Watson, and David L. Woodruff. "Pyomo: modeling and solving mathematical programs in Python." Mathematical Programming Computation 3, no. 3 (2011): 219-260.

[20]. S. Jannu *et al*., "Energy Efficient Quantum-Informed Ant Colony Optimization Algorithms for Industrial Internet of Things," in *IEEE Transactions on Artificial Intelligence*, 2022, doi: 10.1109/TAI.2022.3220186.