Quantum Circuits and Compilation
Grover's Algorithm

Alex Blackwell, and Anonymous

April 10, 2023

1 Introduction

Quantum computing is a novel high-tech field in computer science that has, in recent years, gained extreme popularity for its potential uses in computing speedup. By the nature of qubits, one can create a superposition of an exponential number of states, which then run through the quantum algorithm and return all possible outputs for said states.

However, this speedup is counterbalanced by the complexity of said algorithms. For the most part, classical algorithms simply have better Big-O complexity, resulting in no improvements when converting into quantum. This has been one of the largest issues preventing mainstream adoption of quantum computers.

But, there are a handful of niche algorithms that are capable of outperforming their classical counterparts. One such algorithm is Grover's algorithm. And despite its niche use cases, it has been utilized in many different experiments to create quantum solutions to traditionally classic problems.

We provide an overview of Grover's Algorithm at a high level and discuss three papers in detail involving different applications of Grover's Algorithm. We present advantages and disadvantages of Grover's algorithm in both the general sense and with regards to the discussed papers.

2 Basics of Grover's Algorithm

Grover's algorithm is a search algorithm for randomly ordered elements of size N. On a classical computer, an unstructured search has a time complexity of O(N). Due to the nature of the elements being unstructured, a classical algorithm must query—in the worst case—N elements to find the

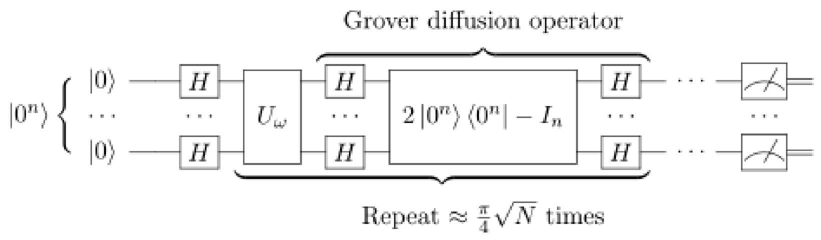
Grover diffusion operator
$|0^n\rangle$
$|0\rangle$
H
U_ω
$2|0^n\rangle\langle 0^n| - I_n$
Repeat $\approx \frac{\pi}{4}\sqrt{N}$ times

Figure 1: A Deconstruction of Grover's Algorithm [12]

desired element. In the average case, the classical algorithm can expect to stumble across the targeted element after N/2 queries. However, due to the nature of quantum mechanics, it has been shown that Grover's algorithm can be done with only $O(\sqrt{n})$ steps—a quadratic speed-up from classical search [7]! At a high level, Grover's algorithm has four steps:

1. **Initialization.** Set equal likelihoods of being the target for all elements.
2. **Oracle.** *Verifies* if an element is the target for each element.
3. **Amplitude amplification.** Amplify the likelihood of the target being found correctly.
4. **Measurement.** Select one of the elements (hopefully the target that we amplified).

2.1 Initialization

Because of the benefit superposition has to the quantum speedup, this is a relatively simple but vital step. Simply apply the Hadamard onto all n qubits [7]. In most literature, this is indicated as $H^{\otimes n}$

This creates an equal superposition of all computational states on each qubit, which represents an item in the search space [11]. This illustrates the uncertainty of the qubit's real state or what it represents in a given scenario [7]. See Figure 1.

2.2 Oracle

Sometimes a solution can be verified asymptotically faster than it can be found. Imagine the simple example where there is one "target" element in

N unstructured elements. To find the target element each of the N elements must be queried (in the worst case); to verify if an element is the target, however, only one query must be performed. The *oracle* realises the idea of replacing "finding the solution" with "verifying the solution." [7]

The function of an oracle can be represented as a matrix. The matrix is the NxN identity except for element u_{ii}, which is negative for target i. The following matrix shows the general structure for such a matrix [7].

$$U = \begin{bmatrix} -1^{f(0)} & 0 & \cdots \\ \vdots & \ddots & \\ 0 & & -1^{f(N-1)} \end{bmatrix}$$

If the function $f(u_i)$ is 1 for the target element and 0 otherwise, then the matrix U will be in the form of an oracle. So, the function $f(u_i)$ does not need to solve the entire search problem; instead, $f(u_i)$ only needs to *verify* if u_i is the target for each element u_i in U [7].

2.3 Amplitude Amplification

The purpose of Amplitude amplification is to *amplify* the likelihood of measuring the target state. In this context, the "amplitude" of a state refers to the probability of the state being measured.

The initialization of states in Grover's algorithm creates a superposition where the likelihood of measuring any one state is equal. That is: the element u_i will be measured with probability 1/N for each ui in U. After the oracle is applied to the states, the target amplitude is negative while all other state amplitudes remain the same. The target state can now be amplified by applying the Grover diffusion operator G (see Figure 1). The Grover operator intuitively flips each amplitude around the average. Recall that since the negative target state brings the average amplitude down, all non-target states are reduced and the target state is increased by about 3 times its original value [7]. The result of applying the Grover operator is the target state is amplified.

Each time the Grover operator is applied, the target state is amplified more. It turns out that applying the Grover operator O($\sqrt{N}$) times—or O($\sqrt{N/M}$) times for M targets—is sufficient amplification of the target state to obtain the correct answer at measurement time with high certainty [7].

2.4 Measurement

Measurement collapses the superposition of qubits into a single state. Due to the repeated application of the Grover diffusion operator, the amplitude of the target state is much more likely to measure than any other state. So, the measured state in the final step of applying Grover's algorithm is the target state with arbitrarily high certainty [7].

3 Paper Summaries

3.1 Quantum Secret Sharing Based on Quantum Search Algorithm [10]

Quantum Secret Sharing (QSS) combines quantum mechanics with sharing of secrets between agents [10]. For the sake of simplicity, we will consider "secrets" to be a single target qubit. To fix ideas, say Alice wants to send Bob her favourite secret number from zero to three. That is: Alice must send Bob one of $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. QSS keeps Alice's answer a secret by encoding her response and splitting it into many parts before sending the parts to different agents. The agents alone—with only a fraction of Alice's response—cannot determine the message's contents; however, by combining the pieces and decoding the result Alice's message can be understood [10].

The Hsu QSS protocol uses three different states (1) the carried state, (2) the encoded state, and (3) the key state [10]. The carried state $|S_i\rangle$ is randomly chosen and will "carry" the transmitted information. To extend the example of Alice and Bob, the transmitted information is Alice's secret number, and the carried state will (eventually) contain this information. The encoded state $|S_i\rangle_w$ encodes the transmitted information into the carry state. Encoding helps ensure Alice's secret number is kept a secret. The key state $a|w\rangle$ is obtained after applying decoding to $|S_i\rangle_w$. The key state consists of phase a and the target qubit $|w\rangle$—Alice's secret number.

To illustrate the encoding and decoding process, let Alice's favourite secret number from 0 to 3 be 2; so, Alice wants to send Bob the state $|10\rangle$. Initial states are constructed similarly to how initialization is done in Grover's Algorithm. Each initial state (size 4 qubits) is a tensor product of individual states (2 qubits). Put explicitly for our example, there are 16 initial states each of which is a tensor product of the following individual states: $1/\sqrt{2}(|0\rangle + |1\rangle)$, $1/\sqrt{2}(|0\rangle - |1\rangle)$, $1/\sqrt{2}(|0\rangle + i|1\rangle)$, $1/\sqrt{2}(|0\rangle - i|1\rangle)$. After initialization is complete, a carrier state must be selected at random. Let our initial state be $|S_1\rangle = 1/2(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$. With the carrier state selected, Alice's secret number of 2 can be encoded therefore making the encoded state. To encode Alice's secret number 2, apply the encoding operator $U_{S_1} = I - 2|S_1\rangle\langle S_1|$ to $|S_1\rangle$ to get the encoded state $|S_i\rangle_{10}$. To decode the message and obtain Alice's secret number, the decoding operator $-U_{S_1} = 2|S_1\rangle\langle S_1| - 1$ must be applied to $|S_i\rangle_{10}$.

With an understanding of how encoding and decoding work, the Hsu protocol can be introduced. Consider a 2-qubit example involving 2 additional agents to transmit Alice's secret: Bob and Charles. The Hsu protocol works as follows [10].

1. Alice randomly prepares some carrier state $|S_i\rangle$ and applies encoding.
2. Alice sends one of the encoded qubits to Bob and the other to Charles who both announce publicly what they got.
3. Alice confirms they have received the qubits.
4. Alice announces her $|S_i\rangle$ publicly.
5. Bob and Charles combine their qubits and perform $-U_{S_i}$ to determine the secret $|w\rangle$.

The Hsu QSS is related to Grover's Algorithm because the protocol is fundamentally one iteration of Grover's Algorithm [10]. Initialization is done by Alice, the encoding operator U_{S_i} is the oracle, $-U_{S_i}$ is the diffusion operator, and measurement is conducted by Bob and Charles.

3.2 Solving Binary $\mathcal{MQ}$ with Grover's Algorithm [8]

The multivariate-quadratic or $\mathcal{MQ}$ problem is a hard problem within cryptosystems and is considered NP-Complete. However, solutions can be brute-forced using enumeration. For classical computing, this takes $\Theta(2^n)$, where n is the number of variables. Grover's algorithm improves the runtime of classical computation to $\Theta(2^{n/2})$ given that these equations are over $\mathbb{F}_2$. To accomplish this task, an arbitrary Oracle is required; Peter Schwabe Bas Westerbaan offer two such oracles [8]. Henceforth, their research will be referred to as $\mathcal{MQ}$ *Grover*. Additionally, $\mathcal{MQ}$ *Grover* illustrates that far less than 1000 logical qubits are needed to break cryptographic schemes [8].

Before discussing the oracles, the $\mathcal{MQ}$ problem will be briefly defined to establish background knowledge. The $\mathcal{MQ}$ problem's goal is to find values $(x_1, x_2, ..., x_n)$ such that a system of m equations has a solution, with each equation $1 \leq k \leq m$ defined as

$$\sum_{1 \leq i,j \leq n} \lambda_{i,j} x_i x_j = v_k, \quad \lambda_{i,j} = "cube" \in \mathbb{F}_2 \quad [8]$$

However, these equations can be rewritten into "convenient form." This form still holds a regular system, but with a new restriction that $\lambda_{i,j} = 0$ in all m equations when $i > j$. This transformation can be done to any system by adding a new equation E_{m-1} and variable x_{n+1}, and redefining the "cubes" as follows, where $1 \leq i, j \leq n+1$ and $1 \leq k \leq m$ [8]:

$$\lambda^{'}_{i,j} \in E_k = \begin{cases} \lambda_{i,j} & i = j \leq n \\ \lambda_{i,j} + \lambda_{j,i} & i < j \leq n \\ 1 + v_k & i = j = n+1 \\ 0 & otherwise \end{cases}$$

$$\lambda^{'}_{i,j} \in E_{m+1} = \begin{cases} 1 & i = j = n+1 \\ 0 & otherwise \end{cases}$$

Notice that $x_{n+1} = 1$, thus $\lambda^{'}_{n+1,n+1} = v_k + 1$ to preserve the original equality in the system [8]. Using these convenient forms allows for a simpler translation into quantum circuit solutions, as is therefore the system used henceforth when discussing circuits solving systems of quadratics.

The first oracle is simple yet effective. It requires $n + m + 2$ qubits, where n = number of variables, and m = number of equations, in convenient form. The n bits are the input variables, the m bits hold the results of the equations, $|t\rangle$ is used for intermittent storage and, $|r\rangle$, holds $|1\rangle$ iff a solution is found. The m + 2 bits are all initialized to $|0\rangle$. It takes, at most, $n^2 + 2n$ gates to calculate an equation and put the results into its respective qubit [8]. As there are m such qubits, equation calculation takes $m(n^2 + 2n)$. Then to confirm if all equations are satisfied, a multiply controlled Toffoli gate is used to send $E_1 * E_2 * ... * E_m$ to $|r\rangle$. Notice that $|r\rangle$ can only be 1 if every equation is satisfied. As this is a quantum circuit, the gates and calculations must be reversed. This takes $m(n^2 + 2n)$ gates [8]. Thus, the maximal number of gates used for this oracle is $2m(n^2 + 2n) + 1$. As neither the equations nor the components of the equations are calculated in parallel, it is clear that the circuit depth is $O(n^2 m)$ [8]. By adding more qubits to allow for more parallelization, the depth can be reduced to $O(n + m)$ using $n^2 + m$ bits [8].

The second oracle is more complex and thus has a more abstract definition. To minimize the logical qubits used, a counter and increment function are utilized. This reduces the register usage down to $n + \lceil log_2 m \rceil + 3$. The $\lceil log_2 m \rceil$ registers act as a counter and are incremented if an equation is satisfied. This means that each equation will have to be computed and uncomputed twice—leading to double the number of gates [8]. $\mathcal{MQ}$ *Grover* notes that increment circuits are costly without ancillas, and thus uses the cyclical nature of qubits upon certain circuits to create their own increment circuit [8]. The researchers illustrate that this can be accomplished by picking a primitive polynomial $p(x)$ over $\mathbb{F}_2$ of degree c, where $c = \lceil log_2 m \rceil$ and building the corresponding circuit made of up, at most, $2c - 3$ gates [8]. Finally, the oracle checks whether the value in the counter is m, done by a single multiply-controlled Toffoli and up to m X gates. Thus, the total gates used for this oracle are $4m(n^2 + 2n) + m(2c - 3) + m + 1$ [8]. Note that due to the focus on minimal register usage, rather than complexity, this oracle's circuit depth is not focused upon or discussed.

To prove the validity of their findings, $\mathcal{MQ}$ *Grover* examined their oracles' resources upon 84 equations in 80 variables with a single solution, known as the "80-bit secure" parameters. The basic oracle used 168 qubits with approximately $1.4 * 10^{18}$ gates, while the minimal register oracle used only 90 qubits and $2.8 * 10^{18}$ gates [8].

3.3 Quantum Fuzzy K-Means Algorithm Based on Fuzzy Theory [3]

The K-Means algorithm is a classical algorithm used to cluster similar data. Consider n objects where each object x_i is a d-dimensional vector on a euclidean vector space. In this context, "similarity" is defined as the distance between objects, the idea being: similar objects are close together. The K-Means algorithm finds k clusters where objects are closer to the cluster they belong to, than any other cluster. Each of the k clusters has a representative object called the centroid. The centroid is updated at each iteration of the K-means algorithm as the mean of all the objects in the centroid's cluster. The K-means algorithm is as follows.

1. Randomly select k objects as the initial centroids.
2. For each data point x_i for i = 0 to n, calculate the similarity to each k centroids and assign x_i to the cluster of the most similar centroid.
3. Update centroids based on the potentially new cluster memberships.
4. If the end condition is met then stop, otherwise go back to step 2.

Grover's algorithm can be applied to step 2 to find the most similar centroid [3]. The reassignment of objects to clusters without using Grover's algorithm is O(nk); however, Grover's Algorithm can reduce the computational complexity to O($n\sqrt{k}$) [3].

Grover's algorithm can be introduced into K-means by augmenting the algorithm as follows [3].

1. Randomly select k objects as the initial centroids.
2. For each data point x_i for $i = 0$ to n, calculate the similarity to each k centroids using a control-SWAP circuit.
3. For each similarity use phase estimation to store similarities in qubits.
4. Use Grover's Algorithm to find the most similar cluster for each object.
5. If the end condition is met then stop, otherwise go back to step 2.

The control-SWAP circuit calculates the similarity between x_i and each k centroids $|c\rangle$. Phase estimation is used to store the degree of similarity between x_i and $|c\rangle$ in the qubit $|a\rangle$. The result of constructing $|a\rangle$ is the oracle since Grover's Algorithm is given a hint of what the target is. Applying amplitude amplification $\sqrt{k}$ times is sufficient to find the most similar centroid to x_i [3].

4 General Advantages of Grover's

The most significant advantage of Grover's Algorithm over classical unstructured searching algorithms is the computational speed-up. Grover's Algorithm provides a method in which to reduce the asymptotic complexity of classical searching algorithms quadratically. The algorithm creates a superposition that can find a target element in an unstructured sequence of elements in O($\sqrt{n}$)—compared to the classical complexity of O(n).

A major reason for the popularity of Grover's algorithm is how common it is required to search an unstructured set of data. Quantum Secret Sharing [10], binary $\mathcal{MQ}$ [8] and fuzzy K-Means [3] differ drastically as algorithms; however, they all share the same quadratic speed-up from applying Grover's Algorithm to an unstructured search subroutine. The three problems have or construct the property of having a solution that is much easier to verify than it is to solve. Such a property is common in many algorithms and is another reason for the popularity of Grover's Algorithm.

Grover's Algorithm allows unstructured search problems to be expressed as verifying a solution rather than solving the problem directly. As an intuitive example, consider solving a Sudoku puzzle [6]. A provided puzzle can be verified as correct if it follows the rules of a valid Sudoku puzzle. That is: the row and column constraints are met for the puzzle. Verifying the correct solution given a puzzle is simpler than finding the solution since finding such a solution may require searching through all the possible states of the puzzle. For cases such as the Sudoku puzzle, being able to express a search problem in terms of verifying many possible solutions is beneficial. An otherwise difficult problem is intuitively simplified and, in the process, sped up computationally. However, not all problems can be easily expressed as verifying a solution. In such cases—such as the ones discussed in section 5.3.—developing the Oracle proves more challenging than intuitive. Regarding Quantum Secret Sharing, binary MQ, and fuzzy K-Means no algorithm was simplified due to the intuition of the Oracle.

Binary $\mathcal{MQ}$ and fuzzy K-Means share the advantage of a quadratic speed up from applying Grover's Algorithm to a subroutine involving unstructured searching. For example, to improve the computational complexity of Binary $\mathcal{MQ}$ from $\Theta(2^n)$ to $\Theta(2^{n/2})$, Grover's algorithm decreases the run time of an unstructured search from $\mathrm{O}(n)$ to $\mathrm{O}(\sqrt{n})$. The speed-up of K-Means from the application of Grover's Algorithm is similar to that of Binary $\mathcal{MQ}$. K-Means requires searching for the most similar centroid for each object in order to converge on clusters. This unstructured search was able to be performed quadratically faster by using Grover's Algorithm than is possible with classical computation.

While the asymptotic time complexities improved for Binary $\mathcal{MQ}$ and fuzzy K-Means with the application of Grover's Algorithm, the same is not true for Quantum Secret Sharing. Instead of speeding up the algorithm compared to its classical counterpart, Grover's Algorithm provides a framework in which secrets can be shared in a quantum space. So, the advantage of Grover's algorithm is unique for Quantum Secret Sharing since it did not speed up or provide intuition for solving a subroutine. Instead, Grover's Algorithm is the foundation that is adapted to the context of sharing secrets to produce the Hsu protocol.

5 General Disadvantages of Grover's

Since Grover's Algorithm is a quantum algorithm it is limited by the shortcomings of quantum computing. Such limitations include a high error rate driving the necessity of error correction and fault tolerant gates. A disadvantage of the papers is they suffer from an inaccurate—or lack of—gate and ancilla cost analysis. Finally, a valid Oracle must be constructed and this is not always trivial.

5.1 Error-Correcting and Fault Tolerance

Fault Tolerance and Error-Correcting are very frequently discussed topics within Quantum Computing. This is because, where classical computing has an error rate of 10^{-17} per operation, quantum computing has an error rate of 10^{-2} per gate—which is an extremely large increase [1, Lec. 7]. Considering the number of operations required within a circuit to apply a non-trivial algorithm or function, this indicates that many if not all quantum circuits will fail due to noise and error, giving incorrect results.

However, this can be mitigated by using a specific gate set that is fault-tolerant and/or can be implemented transversally. Fault-tolerant gates are those gates that do not propagate error qubits [1, Lec. 7]. Transversal gates are those that can be broken down simply into single-qubit unitary gates, and is thus a series of Fault-Tolerant gates [1, Lec. 7]. By using such gate sets, alongside quantum error correcting codes that can guarantee to correct singular errors, one can ensure the algorithms runs without issue, by The Threshold Theorem [2]. This theorem proves that if the probability of failure per gate is below a hardware threshold, correcting errors during calculation ensures algorithms run with high levels of success, and that this correction is efficient as only a poly-logarithmic number of gates are added to the circuit [2]. Thus, by using Fault-Tolerant sets we can ensure Grover's runs correctly and continues to have quantum speedup.

One of the most popular universal gate sets with this feature is Clifford + T. The Clifford gates $\{H, CNOT, S\}$ can all be implemented transversally, and T can be implemented fault-tolerantly. However, this requires building a "magic state" ancilla and utilizing teleportation, which is extremely costly [1, Lec. 7]. Due to the universality of this set, it is a common gate set to see in many research papers when examining quantum circuits as this is a well-known flaw.

In the aforementioned papers, it is clear that this flaw within Grover's algorithm and quantum circuits, in general, was considered, despite not being mentioned explicitly. Within Grover's general algorithm, the diffusion operator can be implemented regardless of oracle with a series of X, H, $i = i(|00\rangle + |11\rangle)$, and Toffoli gates [4]. And each of said gates can be decomposed into Clifford + T trivially. Thus, it suffices to examine the oracles themselves.

In *Quantum Secret Sharing Based on Quantum Search Algorithm* [10], there is no mention of circuit construction, but given the definition of U_{S_1} and U_{S_1} for arbitrary $|S_i\rangle$ and $|w\rangle$, it is clear these can be made fault-tolerantly.

Within *$\mathcal{MQ}$ Grover*, only a small subset of possible gates are used: $\{CNOT$, Toffoli, X, n-bit Toffoli, $SWAP\}$ [8]. Each gate in said subset can be defined within Clifford + T trivially or is explicitly a gate in Clifford + T. $CNOT$ is an element of Clifford, Toffoli and n-bit Toffolis can be written with a series of H, T, and $CNOT$ gates, $X = HZH = HSSH$, where H and S are gates within Clifford, and $SWAP$ is equivalent to 3 $CNOTs$ sequentially. Thus, the authors clearly recognized the requirement of fault-tolerance within their gate set and worked around it.

Finally, within *Quantum Fuzzy K-Means Algorithm Based on Fuzzy Theory* [3], Phase Estimation is used as the oracle, and involves H and $SWAP$ gates, a controlled global-phase gate U, which adds a phase of $e^{2\pi i\theta}$, and QFT^{-1}. U, as a controlled global-phase gate, can be implemented through a series of S and T gates, based on θ. QFT^{-1} is a more complex calculation, but is well-known to use n H gates and $n(n-1)/2$ phase gates, where n is the number of input bits [1] [9]. Again, these phases can be recreated with T and S gates, allowing for fault-tolerant gates, but comes at a significant cost as these phases do not directly translate and involve some error estimation of said phase gates—which, depending on hardware, may still result in error-prone results [1]. In this regard, *Quantum Fuzzy K-Means Algorithm Based on Fuzzy Theory* [3] seems cognizant of this issue within Grover's algorithm but either chose to dismiss the issue in favour of focusing on results, or determined the target audience would already be domain experts and therefore explaining the faults of such a well-known operation redundant.

5.2 Lack of Accurate Gate/Ancilla Cost Analysis in Usage

As mentioned briefly in other sections, this cost and understanding of gates and ancillas within these papers was either severely under-discussed or missed altogether. Due to this lack of discussion, many of the gate and ancilla calculations can be considered faulty at best, or incorrect at worst. This issue primarily occurs when Toffoli gates are used within circuits. As *Quantum Secret Sharing Based on Quantum Search Algorithm* [10] lacks any discussion of gates or ancillas entirely, it cannot be analyzed within this section.

$\mathcal{MQ}$ Grover's first oracle uses nm standard Toffoli gates to calculate

and stores all equations into their respective qubits, an m-qubit Toffoli to evaluate the equations for a solution, and then nm standard Toffoli gates to decompute the equations [8]. The issue here lies within the expensive nature of Toffoli gates; a Toffoli gate can be rewritten as Clifford + T exactly, using 7 T-gates, and 2 H gates alongside 6 $CNOT$ gates as seen in Figure 2. Additionally, in the examination of n-controlled bit Toffoli cost, $\mathcal{MQ}$ *Grover* explicitly states "If one allows one ancillary qubit, one only needs O(n) many 2-qubit gates to construct a n-qubit Tofolli gate" [8]. In the referenced paper [5], controlled-V gates were used, which can be constructed by Clifford + T as seen in Figure 3.

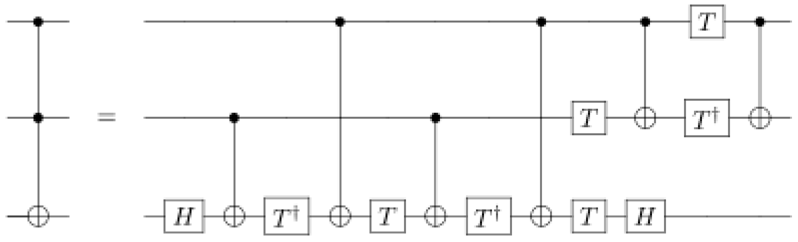

Figure 2: Toffoli Deconstruction into Clifford + T Gateset [13]

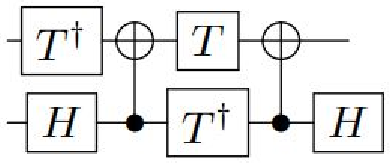

Figure 3: Controlled-V Deconstruction into Clifford + T Gateset [5]

In total, $32n - 96$ elementary (being single-qubit or controlled-V) gates with a single garbage ancilla are required for an n-controlled bit Toffoli construction [5]. To break it down further, $8n - 24$ of said elementary gates are $CNOTs$, meaning $24n - 72$ are controlled-V [5]. As each controlled-V gate has 7 Clifford + T gates within it, the final breakdown for n-qubit Toffoli would be: $8n - 24 + 168n - 504 = 176n - 528$ Clifford + T gates. Through this gate-deconstruction, it is clear the simple gate estimation of $2m(n^2 + 2n) + 1$ with $n + m + 2$ bits does not illustrate the true cost; Toffoli gates are more complex, and an n-qubit Toffoli even more so, so cannot be treated as equal cost to less-complex gates. This is also true of intermediate ancillas used. A more accurate value for Oracle 1 would be: $n + m + 3$ registers, and $2m(n^2 + \mathbf{17}n) + \mathbf{176n - 528}$. The same breakdown can be done for Oracle 2. However, because the final Toffoli in Oracle 2 is c-bit, the cost is far lower. Thus, the original estimation of $n + \lceil log_2 m \rceil + 3$ bits and $4m(n^2 + 2n) + m(2c - 3) + m + 1$ gates is also inaccurate. $n + \lceil log_2 m \rceil + \mathbf{4}$ bits and $4m(n^2 + \mathbf{17}n) + m(2c - 3) + m + \mathbf{176c - 528}$, where $c = \lceil log_2 m \rceil$ is more precise.

Finally, because T gates require preparation alongside magic states and teleportation to be fault-tolerant, it would be preferred to separate the Clifford gates from the T gates when discussing total gate cost and gate depth. However, given complex decompositions such as this, it is understandable why they were avoided by $\mathcal{MQ}$ *Grover*.

Quantum Fuzzy K-Means Algorithm Based on Fuzzy Theory [3], unlike $\mathcal{MQ}$ *Grover* puts minimal effort within the paper itself to examine gate or ancilla cost. In fact, the paper omits this category entirely when examining the results of their findings. Instead, a brief illustration of Time and Space complexity is chosen as their evaluation metrics. However, Space complexity can be seen as analogous to bit usage. In this use case, it has been found that a data point $x_i = \{x_{i1}, x_{i2}, ..., x_{id}\}$ needs only $2 + log_2 d$ qubits, while the k clusters require $2 + log_2 d + log_2 k$ qubits [3]. Thus, in total $4 + 2log_2 d + log_2 k$ bits are required to find the cluster for point x_i, and $n(4 + 2log_2 d + log_2 k)$ bits are required to find the clusters for all n data points. Given the lack of additional ancilla use in QFT and gate U, this is indeed accurate. But, as there is no in-depth discussion of any sort about gate cost, it is evident the *Quantum Fuzzy K-Means Algorithm Based on Fuzzy Theory* [3] had little to no concern for this issue.

To summarize, these papers seem to be indicative of a problem when using Grover's algorithm for real-world problems: there is a fundamental lack of deep, accurate analysis of gate and ancilla cost. This may be because the focus of such papers lies primarily within the application. However, application should not come at the expense of efficient, accurate implementation. And when using expensive implementation, it is vital to discuss or even briefly mention this expense if it is not obvious—such as the cost of n-qubit Toffolis being far more than a standard gate, or the extensive use of T gates to approximate certain phase gates. Without this, certain findings may be inappropriate or hide the true cost.

5.3 Oracle Definition

The Oracle within Grover's algorithm is the linchpin of the entire circuit, and must be customized for every new use-case of the algorithm. However, by the requirements of oracle construction being Fault-Tolerant, and of the specific form discussed in 2.2, this can be difficult to construct. As a reminder, Oracle O can be seen as a circuit that transforms $|x\rangle|y\rangle$ to $|x\rangle|y \oplus f(x)\rangle$ [11]. This means knowing the definition of $f(x)$—where $f(x) = 1$ when x is a solution, and $f(x) = 0$ otherwise—a priori. In many scenarios, for an arbitrary x, this may be difficult or impossible to do. Additionally, even if O were to be an oracle for an input size of n, it may need to be reworked entirely if the input size is altered in any way. The best way to ensure O works is to convert $f(x)$ to a binary mathematical problem, and build O according to $f(x)$. Thus, Grover's usage is severely limited to those problems which fit the niche criteria required to build a solid Oracle circuit. In *$\mathcal{MQ}$ Grover* [8] and *Quantum Fuzzy K-Means Algorithm Based on Fuzzy Theory* [3], the oracles were mathematically defined. However, *Quantum Secret Sharing Based on Quantum Search Algorithm* [10] relies on Alice knowing the secret and building her oracle a priori, and informing Bob and Charles of her $|S_i\rangle$. This is one of few problem spaces where the knowledge of the solution is known beforehand, and allows for this build method for O

Another limitation lies in the ability to translate a given problem into n binary bits. For many problem spaces, this translation is non-trivial and requires a precise, thought-out method to convert the search space to n qubits. Compared to classical computing, where there is little to no need to translate search spaces into bits to locate a solution, it may not be worth the effort. Generally speaking, Grover's algorithm may be best for problem spaces where the search space can be easily converted or limited to a binary scope. In *Quantum Secret Sharing Based on Quantum Search Algorithm* [10], this can be done by converting the secret into a binary representation and thusly creating the oracle and diffusion circuits. *$\mathcal{MQ}$ Grover* [8] limits the system of equations to $\mathbb{F}_2$, allowing for only binary x_i and E_i values. And *Quantum Fuzzy K-Means Algorithm Based on Fuzzy Theory* [3]s uses clustering scoring, where scores act as a bit's amplitude, and $\in$[0,1].

An example of an unsorted search space that does not have these features and therefore cannot be solved by Grover's but can be by a classical algorithm would be to find if there is an ace of spades within a poker player A's 5-card hand. A classical computer would simply check each of the five cards for an ace of spades and return the index, or -1 if not found. But because there is no way to translate the quality of "ace of spades" to a specific qubit in a cost-efficient manner, no oracle can be constructed. If one were to know the index of the card beforehand, say 3 (4th card), then oracle O could be built. However, if the index is known, there is no need to use Grover's to find the card as it has already been found!

6 Conclusion

Grover's Algorithm is a popular algorithm in quantum computing due to its quadratic speed-up in unstructured search. Algorithms commonly have unstructured search subroutines that are candidates for a speed-up from applying Grover's Algorithm. Three such applications of Grover's Algorithm discussed in detail are Quantum Secret Sharing, binary MQ, and fuzzy K-Means. All three algorithms constructed a valid Oracle and share a quadratic speedup in the unstructured search procedure. While Grover's Algorithm provides a faster alternative to classical unstructured search it also comes with disadvantages. The algorithm is victim to the shortcomings of quantum computing such as the need for fault-tolerant or transversal gates. Further, an accurate number of ancillas and each type of gate necessary for quantum algorithms must be considered to guide efficient implementations. Additionally, Grover's Algorithm comes with the unique complication of constructing a correct Oracle for verifying solutions. So, Grover's algorithm provides an opportunity to speed up algorithms involving unstructured searching; however, when adopting Grover's algorithm into real-life applications the overhead of quantum computing should be considered.

References

[1] Matt Amy. *Lecture notes in CMPT 409/981: Quantum Circuits and Compilation.* 2022. URL: `https://www.cs.sfu.ca/~meamy/f22/cmpt981/`.

[2] Daniel Gottesman. *An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation.* 2009. DOI: `10.48550/ARXIV.0904.2557`. URL: `https://arxiv.org/abs/0904.2557`.

[3] Min Hou, Shibin Zhang, and Jinyue Xia. "Quantum Fuzzy K-Means Algorithm Based on Fuzzy Theory". In: *Artificial Intelligence and Security.* Ed. by Xingming Sun et al. Cham: Springer International Publishing, 2022, pp. 348–356. ISBN: 978-3-031-06794-5.

[4] C. Lavor, L. R. U. Manssur, and R. Portugal. *Grover's Algorithm: Quantum Database Search.* 2003. DOI: `10.48550/ARXIV.QUANT-PH/0301079`. URL: `https://arxiv.org/abs/quant-ph/0301079`.

[5] D. Maslov and G.W. Dueck. "Improved quantum cost for n-bit Toffoli gates". In: *Electronics Letters* 39.25 (2003), p. 1790. DOI: `10.1049/el:20031202`. URL: `https://doi.org/10.1049%2Fel%3A20031202`.

[6] Ankur Pal et al. "Solving Sudoku game using a hybrid classical-quantum algorithm". In: *EPL (Europhysics Letters)* 128.4 (2020), p. 40007.

[7] IBM Quantum. *Grover's algorithm.* URL: `https://quantum-computing.ibm.com/composer/docs/iqx/guide/grovers-algorithm`.

[8] Peter Schwabe and Bas Westerbaan. "Solving Binary $\mathcal{MQ}$ with Grover's Algorithm". In: *Security, Privacy, and Applied Cryptography Engineering.* Ed. by Claude Carlet, M. Anwar Hasan, and Vishal Saraswat. Cham: Springer International Publishing, 2016, pp. 303–322. ISBN: 978-3-319-49445-6.

[9] The Qiskit Team. *Quantum Fourier transform.* URL: `https://qiskit.org/textbook/ch-algorithms/quantum-fourier-transform.html`.

[10] Hsin-Yi Tseng et al. "Quantum Secret Sharing Based on Quantum Search Algorithm". In: *International Journal of Theoretical Physics* 51.10 (Oct. 2012), pp. 3101–3108. ISSN: 1572-9575. DOI: `10.1007/s10773-012-1191-x`. URL: `https://doi.org/10.1007/s10773-012-1191-x`.

[11] George F. Viamontes, Igor L. Markov, and John P. Hayes. *Is quantum search practical?* URL: `https://arxiv.org/abs/quant-ph/0405001`.

[12] Wikipedia, the free encyclopedia. *Grover's algorithm.* 2022. URL: `https://en.wikipedia.org/wiki/File:Grover's_algorithm_circuit.svg`.

[13] Wikipedia, the free encyclopedia. *Toffoli gate.* 2022. URL: `https://en.wikipedia.org/wiki/File:Qcircuit_ToffolifromCNOT.svg`.