Adaptive Grover for Quantum Search

Weiheng Su, Jianfeng Lin, Gayatri Susarla

Abstract

The Adaptive Grover Algorithm is designed for weighted data search problems, it is essential for quantum subroutines in various complex algorithms. In this paper, we addressed briefly about Grover's search approach on weighted databases. Our analysis and implementation reveal the adaptive quantum search nature compared to linear search. We addressed the key challenges in designing an adaptive Grover search compared it with original Grover Algorithm, and explained about results for distributions in real-world weighted data. All implementation code, data and results are available in the GitHub repository at https://github.com/griefinglol/adaptive_grover/blob/master. In conclusion, our project provides a few enhancements to the original Grover's search, implementation strategies, and scalability of the algorithm.

1 Introduction

With the rapid exponential growth of data, efficient search algorithms have become crucial for extracting meaningful information from large datasets. Compared with traditional search techniques, which utilize linear search which resulted in less efficient search time, struggling to handle the sheer volume and variety of data, faster search algorithms became one of the main fields for data scientists to explore. Quantum search algorithms offer a significant advantage by reducing the time complexity of search tasks. In Quantum search algorithms, Grover's algorithm stands out as it offers a drastic speed up compared to classic search, achieving a search time complexity of $O(\sqrt{N})$. However, Grover's algorithm originally applies to unstructured data. To adapt Grover's algorithm to real world data, this project will study the obstacles in traditional Grover's algorithm in the application to weighted data. Simultaneously, evolution of the Grover's algorithm will be explored by developing a variant of the diffusion operator in the traditional Grover's algorithm, returning a generalized form of Grover's algorithm, which finally allows the algorithm to adapt to data that follows any distribution. To mimic the real-world data and perform validation on the Adaptive Grover's algorithm, this project utilized data that follows four types of different distributions, including normal, heavy-tail, uniform, and quasi-uniform distribution. By adapting Grover's algorithms, this project seeks to optimize search performance and accuracy in the real world.

2 Grover Search On Weighted Database

2.1 Background

Consider a database x_1 , . . . , x_M, where M is an integer representing the total number of elements. Each element in the database is a real number that characterizes certain properties of objects. In traditional unweighted search problems, all elements are distinct. However, in many real-world databases, elements may repeat, representing shared characteristics. Let there be N unique types of elements in the database, denoted as y_1,...,y_N . The database can therefore be reorganized as (y_1, p_1), . . . , (y_N, p_N), where p_1, . . . , p_M represent the proportions of the distinct characteristics y_1 , . . . , y_N in the total database.

Directly applying Grover's search algorithm designed for unweighted databases x_1 , . . . , x_M to a weighted database y_1 , . . . , y_N is unsuitable and requires modifications to account for the weights. According to Grover's search based on unweighted data, the evolution operator consists of two components: the oracle U_O and the diffusion U_D. By repeatedly applying these operators, the amplitude of the target state is amplified. This iterative process increases the probability of measuring

the target state, enabling the search to be completed with high probability after a relatively small number of iterations. As a result, the search can be effectively finalized in a single measurement step. Now, we need to develop a more generalized approach to extend this idea and thus relax the constraint imposed by unweighted data.

To perform the quantum search task on y_1 , . . . , y_N, the first step is to consider representing the distribution of a database using a quantum state $|W\rangle$.

$$|W\rangle = \sum_{i=1}^{N} P(i)|i\rangle \tag{1}$$

where $|W\rangle$ is a superposition of the orthonormal basis $\{|1\rangle, |2\rangle, \cdots, |N\rangle\}$ and it is clear that $\sum_{i=1}^{N} P(i)^2 = 1$. Then, we keep the oracle U_O same and change the structure of diffusion U_D. The new U_D is defined by

$$U_D = 2|W\rangle\langle W| - I \tag{2}$$

The amplification operator required by the Grover search is defined by G:= $U_D U_O$. Suppose that the target state is $|t\rangle$ and the whole state evolves to this state after performing G for k times. In this case, the step number for searching $|t\rangle$ is k.

We refer to this new variant of Grover's algorithm as the Adaptive-Grover algorithm because it can adapt to any distribution of the database. Additionally, the Adaptive-Grover algorithm serves as the generalized version of Grover. You can imagine if the input database is uniformly distributed, the state$|W\rangle$ becomes $|+^{\otimes n}\rangle$ making it equivalent to the original Grover's search algorithm.

2.2 Analysis

In this section, we will compare the diffusion U_D between Grover's algorithm and the Adaptive-Grover algorithm. The diffusion U_D used in Grover's algorithm can be decomposed as:

$$H^{\otimes n} Z_{or} H^{\otimes n} \tag{3}$$

where Z_{or} is defined as follows:

$$Z_{OR}|x\rangle = \begin{cases} |x\rangle & x = 0^n \\ -|x\rangle & x \neq 0^n \end{cases} \tag{4}$$

To be clear, we can confirm that $Z_{or} = 2|0^{\otimes n}\rangle\langle 0^{\otimes n}| - I$ in unitary matrix form. Now the diffusion U_D can be constructed as unitary gate Z_{or} and a bench layer of Hadamard gates and written in this form:

$$U_D = H^{\otimes n}(2|0^{\otimes n}\rangle\langle 0^{\otimes n}| - I)H^{\otimes n} = 2|+^{\otimes n}\rangle\langle +^{\otimes n}| - I \tag{5}$$

In contrast, new diffusion U_D is defined as :

$$U_W Z_{or} U_W \tag{6}$$

where U_W is a unitary gate which transforms $|0^{\otimes n}\rangle$ to $|W\rangle$, that is $U_W|0^{\otimes n}\rangle = |W\rangle$. Following the same approach, we can extend our analysis to the next step:

$$U_D = U_W(2|0^{\otimes n}\rangle\langle 0^{\otimes n}| - I)U_W^{\dagger} = 2|W\rangle\langle W| - I \tag{7}$$

Now, we will demonstrate how this new diffusion form, U_D, effectively increases the amplitude of the target state. Let us try giving them an arbitrary input $|\phi\rangle = \sum_{x=0}^{N-1} \alpha(x)|x\rangle$, we get

$$\begin{aligned} U_D|\phi\rangle &= 2|W\rangle\langle W| - I)|\phi\rangle \\ &= 2|W\rangle\langle W|\phi\rangle - |\phi\rangle \\ &= 2(\sum_{x=0}^{N-1} P(x)\alpha(x))|W\rangle - |\phi\rangle \\ &= \sum_{y=0}^{N-1} (2CP(y) - \alpha(y))|y\rangle \end{aligned} \tag{8}$$

where C = $\sum_{x=0}^{N-1} P(x)\alpha(x)$. Given C > 0, it is easy to verify that the amplitude of the state $|y\rangle$ increases if $\alpha(y) < 0$ and decreases otherwise. Fortunately, the oracle U_O reflects the sign of the target state $|t\rangle$'s amplitude $\alpha(t)$ while leaving other states unchanged, so the amplitude of the target state $|t\rangle$ increases while the amplitude of non-target states decreases. However, when $\alpha(t)$ is sufficiently large, C can become negative and reverse the previous conclusion. This is not a concern, as if $\alpha(t)$ is large enough, we can get the target state by directly measuring $|\phi\rangle$ with high probability. As a result, the new diffusion U_D can increase the amplitude of the target state $|t\rangle$, thereby generalizing the database from a uniform distribution to an arbitrary distribution .

2.3 Implementation

Designing the Adaptive-Grover search circuit was the key challenge we successfully addressed in this project. The Adaptive-Grover algorithm consists of repeated applications of the oracle U_O and diffusion U_D, the overall pipeline is shown in Figure 1. Since Adaptive-Grover and Grover share the same oracle, this component is omitted from further discussion. If you are interested, all implementation details can be found at https://github.com/griefinglol/adaptive_grover/blob/master/grover.py. Figure

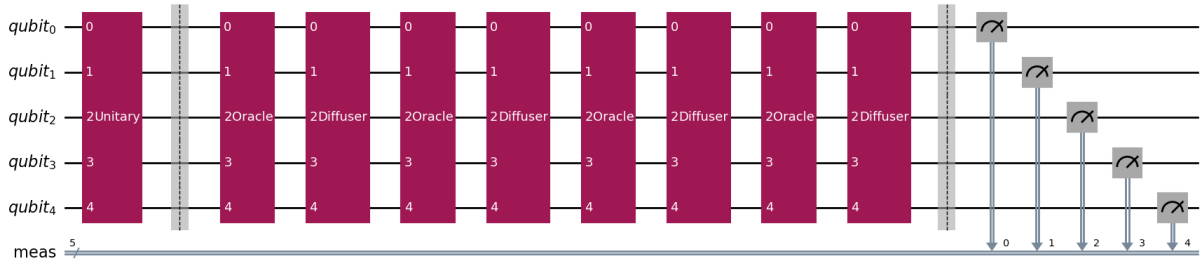

Figure 1: This is an example of implementing an Adaptive-Grover circuit on 5 qubits. The first unitary block is U_W which transforms $|0^{\otimes n}\rangle$ to $|W\rangle$, followed by repeating oracle U_O and diffusion U_D.

2 depicts the structure of diffusion U_D. By comparing Figure 1 and Figure 2, we observe that if a unitary gate U_W that transforms $|0^{\otimes n}\rangle$ $|W\rangle$ can be generated, Adaptive-Grover's search algorithm can be ready. This construction builds upon the foundation of Grover's search algorithm, leveraging the transformation enabled U_W to achieve the desired adaptive search functionality.

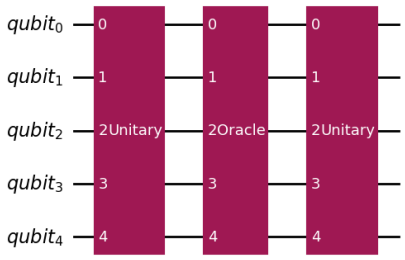

Figure 2: This is the structure of diffusion U_D. The oracle block in the middle is Z_{or} and the unitary gate on sides is U_W .

Algorithm 1 describes the procedure for generating unitary gate U_W based on the distribution of the database. The code uses the Gram-Schmidt process to construct a unitary matrix that maps the initial state $|0^{\otimes n}\rangle$ to the target state. The first column of the unitary matrix will be the target state, and the remaining columns are constructed to ensure unitary.

Algorithm 1 Generate unitary gate U_W

Input : targetAmplitudes (list): List of complex amplitudes for the target state.
Output: unitary gate U_W.

```
state_amp = Normalize the input targetAmplitudes
nitial_state = Create the initial state |0...0⟩
basis = Create standard basis vectors
Q[:, 0] = target_state

for i ← 0 to n − 1 do
    v = basis[:, i]
    for j ← 0 to i − 1 do
        projection = dot_product(conjugate(Q[:, j]), v)
        v = v - projection × Q[:, j]
    end for
    Q[:, i] = v
end for
return Q
```

3 Experiment

3.1 Datasets

The data used to test the adaptive Grover's algorithm in this project generally follows four types of distribution, including heavy tail distribution, Normal distribution, Uniform distribution, and Quasi-uniform distribution. These data are used as it is believed they can best represent real-world data which are weighted. Figure 3 demonstrates 4 datasets each with 1000 samples with values from 1-32.

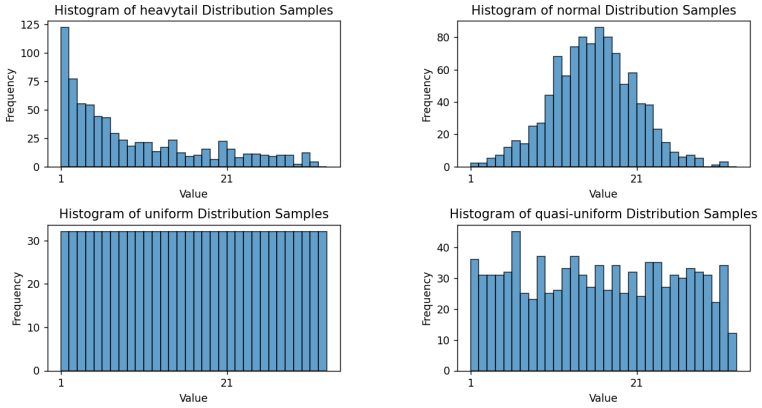

Figure 3: These are the 4 types of data tested on the Adaptive Grover's algorithm, following heavy-tail distribution, normal distribution, uniform distribution, and quasi-uniform distribution, each with 1000 samples, valued from 1-32

3.2 Results on 5 qubits

Figures 4 and 5 depicted a comparison result of Adaptive Grover's algorithm on the real-world datasets that follow heavy-tail, normal, quasi-uniform, and uniform distribution. As demonstrated in Figure 4, adaptive Grover's algorithm performs a slightly faster search for heavy-tail distributed data and results worse for data that are normally distributed. Both algorithms did not have an outstanding result in terms of stability on these two types of data. However, for data that follows a quasi-uniform and uniform distribution, Adaptive Grover's algorithm outperforms the classic algorithm in terms of search time and stability as shown in Figure 5. It is observed that the algorithm performs poorly when it is applied to data that have low likelihood.

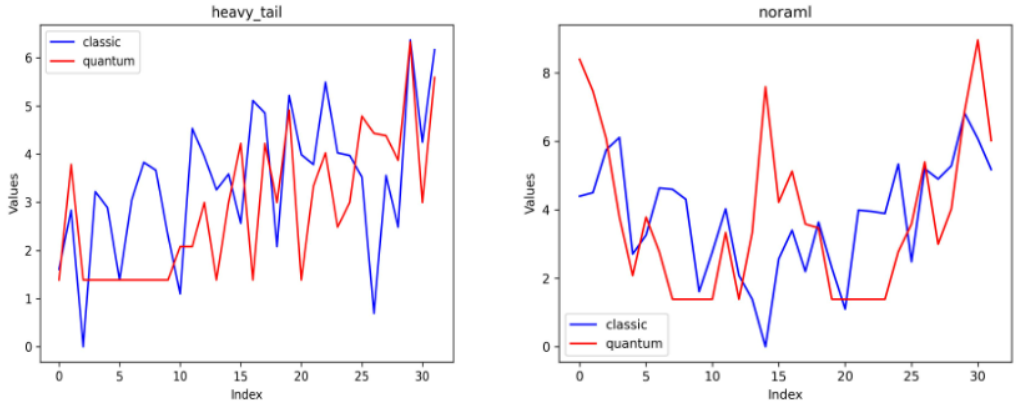

Figure 4: These are the result comparison of Adaptive Grover's algorithm and classic algorithm on data that follows heavy-tail distribution and normal distribution

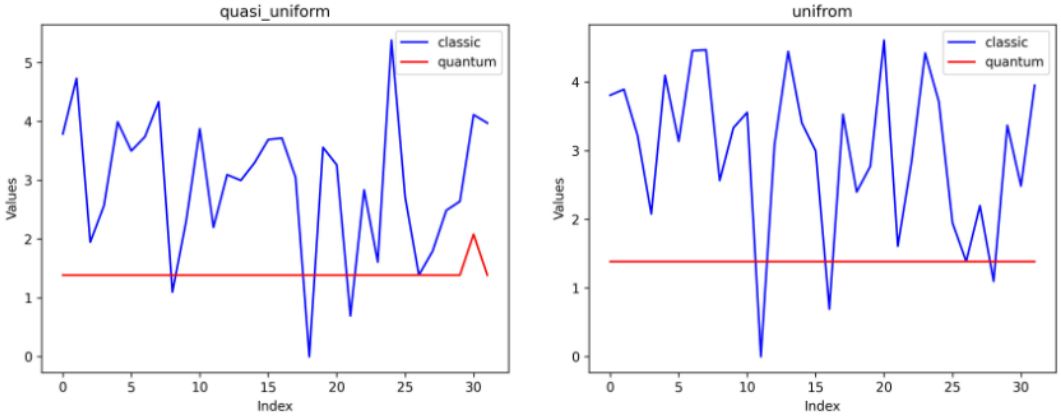

Figure 5: These are the result comparison of Adaptive Grover's algorithm and classic algorithm on data that follows quasi-uniform distribution and uniform distribution

Why adaptive Grover's algorithm perform poorly when applying data with uneven likelihood? Remember the conclusion we get in Section 2.2, for an arbitrary input $|\phi\rangle = \sum_{x=0}^{N-1} \alpha(x)|x\rangle$, after applying diffusion U_D, the result quantum state is a superposition $\sum_{y=0}^{N-1}(2CP(y) - \alpha(y))|y\rangle$, where C = $\sum_{x=0}^{N-1} P(x)\alpha(x)$. Actually, C quantifies the overlap between state $|\phi\rangle$ and $|w\rangle$.

Let's analyze the whole process step by step. The init state is $|0^{\otimes n}\rangle$. First, apply a unitary gate U_W which transforms $|0^{\otimes n}\rangle$ to $|W\rangle$. Next, apply oracle U_O, and get the state

$$|W'\rangle = \sum_{i=1}^{N} P(i)^{f(i)}|i\rangle \tag{9}$$

where f(k) is 1 if $|k\rangle$ is the target state and 0 otherwise. Considering two extreme cases, if the amplitude of the target state is close to 0, the overlap between $|W\rangle$ and $|W'\rangle$ remains nearly 1. In this scenario, C is close to 1. The state after applying diffusion U_D becomes

$$\sum_{y=0}^{N-1}(2CP(y) - P(y)^{f(y)})|y\rangle. \tag{10}$$

Because C is close to 1, the term 2CP(k) + P(K) is less than 3P(K). As a result, the increase in the target state's amplitude is minimal, making the amplification effect of the diffusion operator U_D negligible .

On the other hand, if the overlap between $|W\rangle$ and $|W'\rangle$ is close to -1/2, the amplitude of the target state after applying diffusion U_D becomes 2 * (-1/2) CP(k) + P(K) which approaches 0. Consequently, the target state's amplitude remains insignificant, making it challenging to identify the state effectively.

3.3 Scalability

The scalability of Grover's algorithm lies in its ability to provide a quadratic speedup over classical unstructured search methods, requiring $O(\sqrt{M})$ iterations for a search space of size M. However, the scalability of Adaptive-Grover's algorithm is influenced by the distribution of the data in the database. Consider a database x_1 , . . . , x_M, where M is an integer representing the total number of elements and there are N unique types of elements in the database, denoted as y_1,...,y_N. If the data follows a tail-like distribution, Adaptive-Grover's algorithm achieves a time complexity of $O(\sqrt{M})$. This matches the performance of the original Grover's algorithm applied directly on database x_1 , . . . , x_M with M unique types of elements, which also requires $\sqrt{M}$ steps on average to locate the target state. On the other hand, if data is evenly distributed, the search problem simplifies to operating on the unique element set y_1,...,y_N. In this case, the algorithm requires only $\sqrt{N}$ steps on average, aligning with the performance of the original Grover's algorithm applied to N unique elements. As a result, the scalability of Adaptive-Grover's algorithm is dependent on the distribution of the dataset, with a lower bound $\sqrt{M}$ and upper bound $\sqrt{N}$. This adaptability ensures that the algorithm performs optimally within the constraints of the data distribution, providing significant advantages over the standard Grover algorithm in cases where M >> N.

The difference in qubit requirements between the original Grover's algorithm and the Adaptive-Grover's algorithm also highlights their scalability. Origin Grover's algorithm needs $O(logM)$ qubits to represent the search space. As N increases exponentially, the qubit count grows logarithmically, maintaining efficient space utilization. In contrast, Adaptive-Grover's algorithm requires between $O(logM)$ and $O(logN)$ qubits. This range reflects the adaptability of the algorithm to the data distribution: for evenly distributed data, fewer qubits $O(logN)$ are sufficient, while for tail-like distributions, the requirement may approach $O(logM)$. Despite this variation, the analysis of the algorithm's time complexity and performance scalability remains consistent, as it dynamically adjusts to the dataset's structure, ensuring efficient utilization of both time and quantum resources.

4 Contributions

Each group member contributed equally to this project

References

Grover, L. K. Quantum mechanics helps in searching for a needle in a haystack. Phys. Rev. Lett. 79, 325 (1997).

Nielsen, M. A. & Chuang, I. L. Quantum computation and quantum information, Cambridge University Press, Cambridge (2000).

Gilliam, A., Woerner, S., & Gonciulea, C. Grover adap tive search for constrained polynomial binary optimiza tion. Quantum 5, 428 (2021).

Yifan Sun, Lian-Ao Wu, Quantum search algorithm on weighted databases. quant-ph, 2312.01590 (2024).

Simanraj Sadana, Grover's search algorithm for n qubits with optimal number of iterations, quant-ph, 2011.04051(2020).

https://github.com/lynkos/grovers-algorithm