QUANTUM SEARCH IN SETS WITH PRIOR KNOWLEDGE

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

UMUT ÇALIKYILMAZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
PHYSICS

SEPTEMBER 2021

Approval of the thesis:

QUANTUM SEARCH IN SETS WITH PRIOR KNOWLEDGE

submitted by **UMUT ÇALIKYILMAZ** in partial fulfillment of the requirements for the degree of **Master of Science in Physics Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Seçkin Kürkçüoğlu
Head of Department, **Physics** _____

Prof. Dr. Sadi Turgut
Supervisor, **Physics, METU** _____

Examining Committee Members:

Prof. Dr. Seçkin Kürkçüoğlu
Physics, METU _____

Prof. Dr. Sadi Turgut
Physics, METU _____

Prof. Dr. Bahtiyar Özgür Sarıoğlu
Physics, METU _____

Dr. Yusuf İpekoğlu
Physics, METU _____

Dr. Kıvanç Uyanık
Physics, Gazi University _____

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Umut Çalıkyılmaz

Signature :

ABSTRACT

QUANTUM SEARCH IN SETS WITH PRIOR KNOWLEDGE

Çalıkyılmaz, Umut
M.S., Department of Physics
Supervisor: Prof. Dr. Sadi Turgut

September 2021, 48 pages

Quantum search algorithm revolutionized the field by reducing the complexity significantly for the search problem. However, by not being able to decrease the complexity to logarithmic scales, this algorithm still needs a significant amount of time to solve the search problem for large sets. It is proven that the order of the time required to solve this problem cannot be reduced further but, making an improvement by some constant factor is still possible. This aim has been pursued by some scientists in the past and some improvement has been made. In this thesis, various methods that can be used to solve the search problem for sets with known probability distributions are developed. As expected, the complexity for this type of problems is reduced by constant factors.

Keywords: quantum search, search problem, search algorithm, quantum computing, Grover search, sets with prior knowledge, strict probability distribution

ÖZ

ÖNBİLGİ SAHİBİ OLUNAN KÜMELERDE KUANTUM ARAMA

Çalıkyılmaz, Umut
Yüksek Lisans, Fizik Bölümü
Tez Yöneticisi: Prof. Dr. Sadi Turgut

Eylül 2021 , 48 sayfa

Kuantum arama algoritması, arama probleminin karmaşıklığını önemli ölçüde düşürerek bu alanda devrim yarattı. Ancak bu algoritma problemi logaritmik zamanda çözmeyi başaramadığı için, bu algoritmayı kullanarak büyük kümeleri aramak hala önemli ölçüde zaman gerektiriyor. Bu problemi çözmek için gereken sürenin mertebesinin daha fazla azaltılamayacağı kanıtlandı ama bir sabit sayı ölçüsünde geliştirme yapmak hala mümkün. Bu hedef bazı bilim insanları tarafından geçmişte kovalandı ve bazı geliştirmeler yapılabildi. Bu tezde olasılık dağılımı bilinen kümelerde arama süresini azaltacak çeşitli yöntemler geliştirildi. Beklendiği gibi, bu tarz problemlerin karmaşıklığı sabit sayı ölçüsünde azaltmanın mümkün olduğu gösterildi.

Anahtar Kelimeler: kuantum arama, arama problemi, arama algoritması, kuantum bilgisayar, Grover araması, bilgi sahibi olunan kümeler, katı olasılık dağılımı

To Alan Turing for imagining possibility of such magical devices

ACKNOWLEDGMENTS

I would like to thank my sister Eylül Çalıkyılmaz for emotionally supporting me through all the challenges I face.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

CW	Clockwise
CCW	Counter Clockwise
$logN$	log_2N

CHAPTER 1

INTRODUCTION

As in any other advancement in science, the notion of quantum information has emerged from need and like any other breakthrough, it solved more than it was planned to do. The need for quantum information arises from the problem of simulating complex quantum systems. To simulate a system having n electrons with all parameters suppressed except for the spins, 2^n complex numbers must be kept and processed as classical information. Likewise, to compute the change in these parameters over time, 2^n differential equations must be solved. To overcome this obstacle Richard Feynman proposed the use of what he called Quantum Computers - Universal Quantum Simulators, using two level quantum systems such as electron spins as the fundamental information unit [1, 2].

This idea had a great impact and directed many scientists to investigate the ways and benefits of utilizing this new sort of information. The superiority of quantum computing over classical one soon proven to exist using some specially designed problems [3, 4, 5, 6]. Then even greater success is achieved by showing the advantage of quantum computers in solving some problems that are almost as old as humanity itself. The most famous problems benefitting from quantum algorithms are prime number factorization problem and search problem. The first was shown to be solvable in polynomial time using a quantum computer by Peter W. Shor [7]. The search problem for a set with N items and M solutions to a function f is proven to be solvable in $O(\sqrt{N/M})$ evaluations of f by Lov K. Grover [8, 9].

The ideal case would be finding an algorithm that solves the search problem in polynomial time, which means the time required to solve the problem being a polynomial of the number of bits required to represent the search space but solving the problem

in a time that is the square root of the time needed for its classical analogue is still very significant. This improvement might not seem as dramatical as the one made by Shor's algorithm. However, search problem is required to be solved much more frequently than factorization problem. Firstly, any problem can be solved by searching through all its possible solutions. If there is no structure between the solutions, which means that searching through all possible solutions is the only way to solve, this method might be the fastest possible one. Secondly, database search is used by almost all modern computer applications. Even the slightest improvement in complexity of a search algorithm working on a massive database, like the one that Google has, might cut the computation cost a great amount. These reasons make quantum search a very resourceful method.

Grover's method requires $(\pi/4)\sqrt{N/M}$ evaluations of the objective function when quantum search is applied on a set with N elements where M of them are solutions. Additionally it is proven that this search cannot be done with less than $O(\sqrt{N/M})$ evaluations [10]. However, an improvement with a constant factor can still be done on his original method. One of these improvements is done in [11]. With a method in this paper, expected number of iterations for quantum search is reduced to $0.69003\sqrt{N/M}$ which is about 88% of the number required for the original Grover search. This method employs consecutive quantum searches until a solution is found, where finding a solution in any of these steps is not certain. In each step $0.58278\sqrt{N/M}$ Grover iterations are applied to the initial state $|\psi_0\rangle$ and then the resulting state is measured (This gives a success probability of 0.84420.).

This thesis aims to find improvements like the one given above for some special cases of quantum search. For this purpose, search is repeated for many times using different initial quantum states and different number of evaluations at each step.

In the rest of this chapter, first the method of quantum search is summarized and the conventions to be used in the rest of the thesis are shown. Then the notion of sets with prior knowledge is described and the subset of them that is to be used are shown. Finally, the methods and applications used for numerical analyses used are given.

The second chapter is reserved for computing the theoretical limits for different methods of search in structured sets. In that chapter, the initial quantum state for the search

is assumed to be determined by a continuous function and the time required to prepare this state is neglected. With these assumptions, first the method of using the same initial state and same number of evaluations for each step is examined. Then the improvement made by using different states and different number of evaluations is observed. For both methods, the equations for the optimal cases are obtained using Lagrange multipliers. Then numerical methods are used to find approximations to the optimal parameters.

In the third chapter, practical use of the methods is of importance. For the methods described there, initial states are chosen to be prepared easily. For this purpose, they are arranged as the superposition of some fraction of the elements in the set to be searched. By this way easy-to-apply approximations of the methods of the previous chapter are formed. In that chapter first it is proven that preparing such quantum states does not add much complexity to the solution. Then a method decreasing the expected number of evaluations by only changing the worst-case scenario slightly is proposed. Then a method employing different initial states but using the same parameter to determine the number of evaluations at each step is studied. Lastly, the effect of relaxing all the assumptions by using different initial states and different number of iterations is found.

1.1 Quantum Search

Search problem is the problem of finding a solution of a function with two possible outcomes. The outcome of the function is assumed to be either 0 or 1 and the solutions are the inputs of this function giving 1 as the result. Any search problem with a different definition can be converted to be defined in this way. For instance, let us observe the problem of finding the elements with a square greater than 5 are to be searched in the set of integers from 0 to 10. A function giving 0 if the square of its input is less than or equal to 5 and giving 1 otherwise can be designed. The performance of a search algorithm is evaluated by how many times it needs to call the defining function to find a solution.

Using classical computation, finding a solution of such a function with M solutions

from a set with N elements takes $N - M + 1$ evaluations in the worst case. There are different types of algorithms like Branch and Bound that decreases the expected number of function evaluations in some cases. However, improving the worst-case scenario for this problem using classical computing seems impossible.

Quantum search is a method to solve the search problem defined above, using quantum computation. It requires $O(\sqrt{N/M})$ calls of the defining function to find a solution. To acquire this performance, this algorithm uses rotations in the quantum state space instead of trial by some order, which is used by every classical search algorithm.

To apply quantum search, the first requirement (assuming the quantum computer to run it has already been developed) is a quantum oracle that evaluates the defining function of the search problem. This oracle has two registers. The first is for storing the indices of the elements of the sets, and the second one is the oracle qubit that the oracle processes depending on the result of the function. The action of the oracle O is given as

$$O\left|x\right\rangle \left|q\right\rangle = \left|x\right\rangle \left|q \oplus f(x)\right\rangle \ . \tag{1.1}$$

where $\oplus$ denotes addition modulo 2, $|x\rangle$ is the first register and $|q\rangle$ is the oracle qubit. Initializing the oracle qubit as $|0\rangle$ and applying the oracle would make the function result for the given index to be written in there. In quantum search algorithm, the oracle qubit is initialized to be in the state $(|0\rangle - |1\rangle)/\sqrt{2}$. This provides a phase change for the solution indices. For this kind of oracle qubit, the action of the oracle becomes

$$O\left|x\right\rangle \frac{\left|0\right\rangle - \left|1\right\rangle}{\sqrt{2}} = (-1)^{f(x)}\left|x\right\rangle \frac{\left|0\right\rangle - \left|1\right\rangle}{\sqrt{2}} \ . \tag{1.2}$$

Since the oracle qubit stays unchanged in any result, it can be omitted for simplicity:

$$O\left|x\right\rangle = (-1)^{f(x)}\left|x\right\rangle \ . \tag{1.3}$$

The reason to change the phase of the solutions might appear to be ambiguous. It

only becomes clear when the whole process of the quantum search is explained. To construct the quantum circuit for quantum search, first the structure of the registers must be determined. For a set with $N = 2^n$ number of elements, the first register must have n qubits to be able to represent each element. In the case where the number of elements is not a perfect power of 2 (which is true for most of the time) the number of qubits can be chosen to be $\lceil \log N \rceil$ and the function can be modified to output 0 for each index $j > N$. The nature of the second register, namely the oracle qubit, is explained above and since it does not change, it will be omitted in the rest of the section.

The algorithm starts with initializing the first qubit in a state which is a superposition of all possible indices with an equal positive coefficient. For this purpose, a fundamental quantum gate named as Hadamard gate is used. The effect of this gate on a single qubit can be shown as below, where Hadamard gate is denoted as H:

$$H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \qquad H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \ . \tag{1.4}$$

To prepare the required superposed state, the register is set as $|0\rangle^{\otimes n}$, or simply $|0\rangle$, by setting each qubit in this register to zero. Then Hadamard gate $H^{\otimes n}$, simply $\mathcal{H}$, is applied to it. The resulting state is denoted as $|\psi_0\rangle$:

$$|\psi_0\rangle = \mathcal{H}|0\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \ . \tag{1.5}$$

After this initialization step, a subroutine called Grover iteration is applied repeatedly to this state. This subroutine consists of two operations. The first one is applying the oracle, the effects of which is described in the former parts of this section. The second step is applying a reflection operator, which reflects the state it is applied with respect to $|\psi_0\rangle$.

The reflection operator, or R, makes the operation $2|\psi_0\rangle\langle\psi_0| - I$ on a state. This operator is obtained by using three elemental quantum circuit elements. First, Hadamard gate $\mathcal{H}$ is applied. Then a conditional phase shift operator, which changes the phase of a state by π only if the state is not $|0\rangle$, is deployed. Then another Hadamard gate

is applied. The combined effect of these three operators is shown below:

$$R = \mathcal{H}(2\left|0\right\rangle\left\langle 0\right| - I)\mathcal{H} = 2\left|\psi_0\right\rangle\left\langle\psi_0\right| - I \ . \tag{1.6}$$

Using the definitions given above, the Grover operator G can be defined as

$$G = RO = (2\left|\psi_0\right\rangle\left\langle\psi_0\right| - I)O \ . \tag{1.7}$$

The effect of applying consecutive Grover operators to $|\psi_0\rangle$ is understood most easily by geometrical visualization of the quantum states. To do this, $|\psi_0\rangle$ can be written as a superposition of two mutually orthogonal states $|ns\rangle$, the equal superposition of all states that are not solutions, and $|s\rangle$, the equal superpositions of all the solution states:

$$\left|ns\right\rangle = \frac{1}{\sqrt{N-M}} \sum_{i \notin S} \left|i\right\rangle \ , \tag{1.8}$$

$$\left|s\right\rangle = \frac{1}{\sqrt{M}} \sum_{i \in S} \left|i\right\rangle \ , \tag{1.9}$$

where S is the set including all the solutions. Using these vectors, $|\psi_0\rangle$ can be written as

$$\left|\psi_0\right\rangle = \sqrt{\frac{N-M}{N}}\left|ns\right\rangle + \sqrt{\frac{M}{N}}\left|s\right\rangle \ . \tag{1.10}$$

Oracle acts as a reflection with respect to the subspace spanned by all the indices that are not solutions. Thus, in the plane defined by $|ns\rangle$ and $|s\rangle$, it acts as a reflection about $|ns\rangle$. The reflection operator, as shown above, acts as a reflection with respect to $|\psi_0\rangle$. Consisting of two reflection operators, G is a rotation and when it is applied to a state in the plane spanned by $|ns\rangle$ and $|s\rangle$, the resulting state is also in that plane.

To show the direction and the amount of the rotation implemented by Grover iteration, let the angle ϕ to be defined by the equation

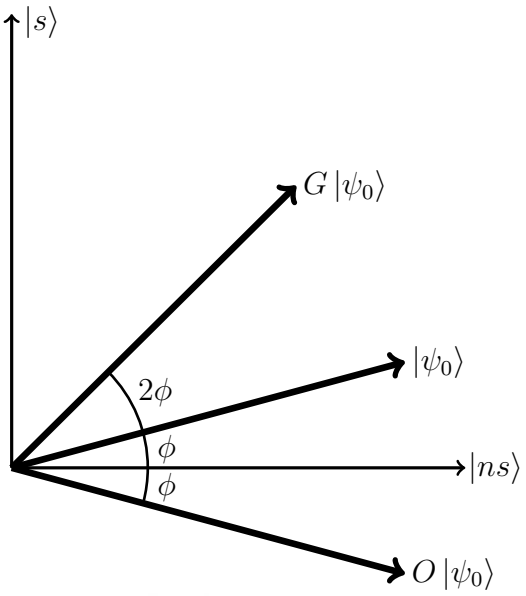

Figure 1.1: Geometric visualization of a single Grover iteration

$$\sin\phi = \sqrt{\frac{M}{N}}\ . \tag{1.11}$$

Then $|\psi_0\rangle$ can be written in terms of ϕ as:

$$|\psi_0\rangle = \cos\phi\,|ns\rangle + \sin\phi\,|s\rangle\ . \tag{1.12}$$

The state obtained by applying a single Grover iteration on the initial state then can be denoted as

$$G\,|\psi_0\rangle = \cos 3\phi\,|ns\rangle + \sin 3\phi\,|s\rangle\ . \tag{1.13}$$

This result can be deducted using geometrical visualization given above. It is known that both O and R are reflection operations. The effect of applying O and G to $|\psi_0\rangle$ can be understood using the nature of these operators as shown in the Figure 1.1. When O is applied, the initial state is reflected with respect to $|ns\rangle$ so rotated in CW direction by 2ϕ. Then when R is applied, the state is reflected by $|\psi_0\rangle$ so it rotated in CCW direction by 4ϕ. The final result is a CCW rotation by 2ϕ. It is obvious that applying G once more will result in an additional CCW rotation by 2ϕ. By this

analysis it can be concluded that each Grover iteration rotates the state by 2ϕ in CCW direction. The state after m consecutive Grover iterations on the initial step $|\psi_0\rangle$ is then

$$G^m |\psi_0\rangle = \cos[(2m+1)\phi] |ns\rangle + \sin[(2m+1)\phi] |s\rangle \ . \tag{1.14}$$

Let $P(m)$ be the probability of finding a solution if the state is measured after applying m consecutive Grover iterations. Using (1.11) and (1.14), $P(m)$ can be denoted as

$$P(m) = |\langle s| G^m |\psi_0\rangle|^2 = \sin^2[(2m+1)\phi] = \sin^2\left[(2m+1)\sin^{-1}\sqrt{\frac{M}{N}}\right] \ . \tag{1.15}$$

For $N \gg M$, this term can be approximated as

$$P(m) \approx \sin^2\left[(2m+1)\sqrt{\frac{M}{N}}\right] \approx \sin^2\left(2m\sqrt{\frac{M}{N}}\right) \ . \tag{1.16}$$

With this approximation, it can be verified that a solution of the problem is obtained almost certainly, which means $P(m) \approx 1$, by measuring the final state after applying $m = (\pi/4)\sqrt{N/M}$ Grover iterations, which also means $(\pi/4)\sqrt{N/M}$ number of evaluations of the defining function.

For the rest of this thesis, only the $M = 1$ case will be considered for simplicity and for that some of the methods in the third chapter only works for this case. With this premise, the approximate value of $P(m)$ becomes

$$P(m) \approx \sin^2\left(\frac{2m}{\sqrt{N}}\right) \ . \tag{1.17}$$

Also in this thesis, initial states other than $|\psi_0\rangle$ are used. Let $|\psi\rangle$ be one of these initial states. Using the real coefficients c_i (since the important thing is the angle between the states, which is a function of $|c_i|$, choosing real coefficients does not make us lose any generality), it is defined as:

$$|\psi\rangle = \sum_{i=0}^{N-1} c_i \, |i\rangle \ . \tag{1.18}$$

In this case, an initial angle same for each index cannot be considered. Instead, for each index i, an initial angle ϕ_i is defined as below:

$$\phi_i = \arcsin |\langle i|\psi\rangle| = \sin^{-1} |c_i| = \sin^{-1} c_i \ . \tag{1.19}$$

These angles can still be assumed to be very small. Then, they can be approximated to be

$$\phi_i \approx c_i \ . \tag{1.20}$$

Another term to be used throughout this thesis is θ_i. This term shows the angle between the final state and $|ns\rangle$, in the case that the solution to the problem is i. This term is defined as

$$\theta_i = (2m+1)\phi_i \approx 2mc_i \ . \tag{1.21}$$

The probability of finding a solution after the iterations also must be modified for using the initial state $|\psi\rangle$. Here, $P_i(m)$ shows the probability of finding the solution after m Grover iterations, provided that i is the solution. Using all the approximations applied above, this term can be written as

$$P_i(m) = \sin^2 \theta_i = \sin^2(2mc_i) \ . \tag{1.22}$$

In the following chapters, θ_i is used in the analyses for convenience. It is trivial to find the initial coefficients, c_i, using (1.21) after finding θ_i values.

Apart from choosing a different initial state, a method similar to the one in [11] can be implemented. This method is applying less than $(\pi/4)\sqrt{N}$, in a way that the expected number of iterations is reduced. Through the thesis, optimum number of iterations that minimizes the expected number of iterations are determined aside from θ_i values.

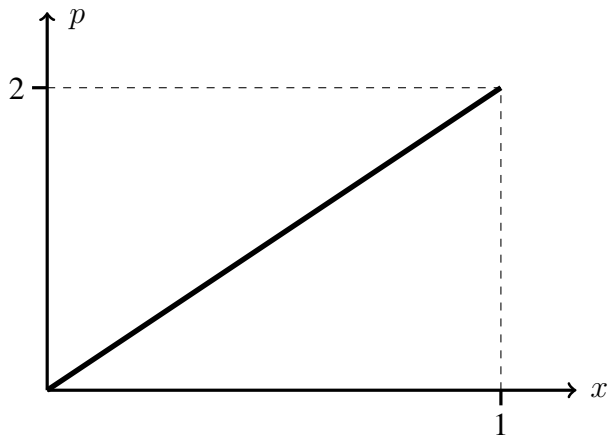

Figure 1.2: The graph of the distribution $p(x) = 2x$

1.2 Sets with Prior Knowledge

In a usual search problem, every element in the set to be searched are assumed to have equal probabilities. This assumption appears from the fact that there is no proof for any element to have a higher probability of being the solution. In this kind of problems, the probability distribution of the indices of the solutions is accepted to be uniform.

In some cases, there might be a prior knowledge about which elements are more likely to be the solution. One example of this is the search problem of sorted sets. In a sorted set, all the elements are sorted by the value to be searched. This problem can be solved in $\log N + 1$ function evaluations classically [12].

The sets considered in this study are not as extreme as the sorted sets, but they are assumed to have continuous, non-uniform probability distributions. Using the probability distribution given for a set, the initial state can be chosen in a way that the expected number of Grover iterations in quantum search can be reduced.

Choosing a suitable initial state would improve quantum search in any set with a non-uniform distribution; but for each different probability distribution, a distinct analysis must be conducted to find such a state. In the following chapters, numerical solutions are deployed for the sets having monotonically non-decreasing polynomial probability distributions. However, the analytic equations found are suitable to use

for any continuous distribution.

An advantage of continuous distributions is being independent of the number of elements in the sets. These distributions are non-negative for values of x in the interval [0,1] and 0 for all other values. The graph of one such distribution is given in figure 1.2. In a set with such probability distribution, $p(x)$, and N elements; probability of the index i being the solution, p_i is

$$p_i = \int_{i/N}^{(i+1)/N} p(x)dx \ . \tag{1.23}$$

Using (1.23), continuous probability distributions can be used for sets with finite number of elements. This makes the computations easier and the results of the thesis to be applied for any set with any number of elements.

1.3 Numerical Methods

In the following chapters, different search procedures are derived. The aim is to find the minimum expected number of iterations possible. To find this minimum value, the optimum values of all the included parameters are to be found. The implicit equations for the optimality of each procedure are found, however finding the explicit solutions using analytical methods seems impossible. Thus, various numerical computation methods are deployed and close estimates to the optimum values of the parameters are gathered.

In the second chapter, to find approximations to the exact values of the optimality parameters, inverse of a transcendental function $f(x)$ is to be found. To determine $f^{-1}(y)$ for a given y, x value satisfying the relation $f(x) - y = 0$ is found using Newton-Raphson method. In this method, first an initial value x_0 is assigned to x. Then next approximations x_α are found step by step using the equation given below:

$$x_\alpha = x_{\alpha-1} - \frac{f(x_{\alpha-1}) - y}{f'(x_{\alpha-1})} \ . \tag{1.24}$$

This method operates by drawing linear lines with a slope $f'(x_{\alpha-1})$ that are including

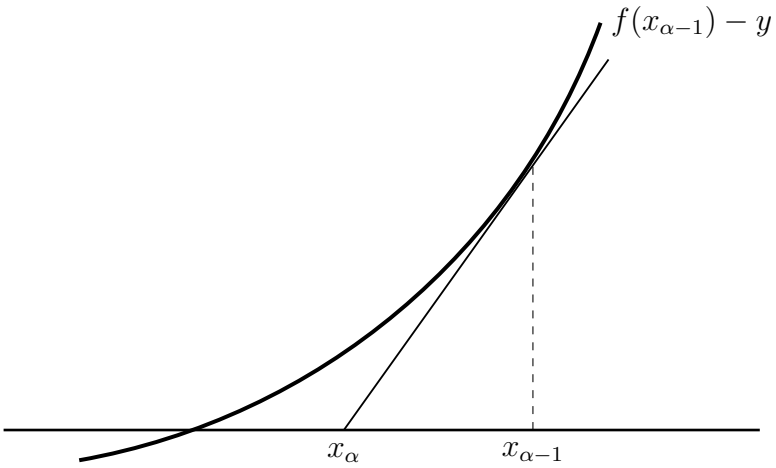

Figure 1.3: Geometric visualization of the Newton-Raphson method

the point $f(x_{\alpha-1}) - y$. Then it determines the next approximation as the x coordinate of the point of interception between this line and the x axis. This process continues until a desired accuracy is reached. Accuracy can be measured by comparing the values $f(x_\alpha)$ and y. The geometric visualization of an iteration of the Newton-Raphson method is given in Figure 1.3.

The probability functions under consideration are assumed to be continuous. In the calculations, discrete θ_i values are used but the number of elements in the set are chosen to be large so that the resulting optimum values of θ_i can be represented with continuous functions. The aim is to make the results independent of the size of the sets to be searched. How to calculate the probability of a discrete index i to be the solution, using the continuous probability function, is given in (1.23). The final angle of a discrete index θ_i is found similarly given the continuous function θ. In the analytical calculations, discrete probability and angle values are used with an unspecified set size N. In the numerical calculations, the discrete probabilities are computed using (1.23) for the given continuous distribution function. In these calculations the number of elements in the sets are assumed to be 10^6 as it is thought to give sufficient amount of precision.

The numerical calculations of chapter 2 are conducted by a custom-made computer program since no commercial mathematics program is up to the task. Newton-Raphson method is used to find inverses of functions in the various iterations for approximat-

ing to the required parameters. In these programs, a better approximation is obtained in each iteration until the difference between two successive approximation becomes smaller than a predetermined precision. The precision value, which determines the point to stop, is set to be 10^{-6} for all iterations.

Optimality equations for the methods of chapter 3 also cannot be solved analytically. However, the numerical solutions of these equations are much easier than the ones in the second chapter. To find the parameters of each method in the third chapter, finding the minimum of a function is adequate. To find these minimum points, FindMinimum function of Wolfram Mathematica is used.

CHAPTER 2

QUANTUM SEARCH WITH GENERIC INITIAL STATES

In this chapter, it is assumed that any state can be prepared consuming a negligible amount of computing power and time. Using this assumption, search methods utilizing repetitive quantum searches are developed. For these methods the starting state of the quantum search j can be any state $|\psi\rangle_j$ that can be defined as in (1.18), and the number of iterations m_j in each quantum search can be any integer between 0 and $(\pi/4)\sqrt{N}$. The expressions for optimal states and iteration numbers minimizing the expected number of function evaluations are found for each method. Then estimates for the optimum values of these parameters are obtained and the expected numbers of evaluations for different probability distributions are calculated.

The two methods of this chapter differ from each other only in number of parameters. The first method, namely "Symmetric Search with Generic Initial States" introduced here uses the same number of iterations, $m = m_1 = m_2 = ...$, and same initial state, $|\psi\rangle = |\psi\rangle_1 = |\psi\rangle_2 = ...$, for each quantum search where in the second method, "Asymmetric Search with Generic Initial States", different iteration numbers and initial states can be chosen for the steps.

2.1 Symmetric Search with Generic Initial States

2.1.1 Definition of The Method

This search method employs consecutive quantum searches with a fixed initial state and a fixed number of iterations. In any of the quantum searches, the probability of finding a solution is not equal to one. This means that the number of possible steps

to be used can be very large. However, using this kind of method instead of Grover's original algorithm, improves the expected total number of iterations for searches in sets with prior knowledge. It has already been proven in [11] that using $|\psi_0\rangle$ for each step of this kind of method decreases the expected number of Grover iterations in sets without prior knowledge. Unfortunately, the methods in this thesis cannot make any improvement on these sets.

Let m be the number of iterations to be applied and $|\psi\rangle$ be the initial state defined as in (1.18). Also let θ_i be the angle between the final state and $|ns\rangle$ when the solution to the problem is i. This term is defined in (1.21). Also define E_i to be the expected number of iterations applied before a solution is found, in the case that i is the solution to the problem. E_i can be written in terms of the parameters defined above as

$$E_i = m + (1 - \sin^2\theta_i)m + (1 - \sin^2\theta_i)^2 m + ... \tag{2.1}$$

Derivation of this expression is quite straightforward. The first quantum search is going to be applied definitely. If the solution is not found after the first search, which has a probability of $(1 - \sin^2\theta_i)$, the second search is applied. If a solution is not found after both the first and the second search, which happens with probability $(1 - \sin^2\theta_i)^2$, then the third search is applied and so on.

The expression for E_i can be simplified by being arranged as a geometric series

$$E_i = m\sum_{j=1}^{\infty}(1 - \sin^2\theta_i)^{j-1} = \frac{m}{\sin^2\theta_i}\ . \tag{2.2}$$

Knowing the expected number of iterations with a condition that i is the solution, the expected number of the whole procedure E can be written as

$$E = \sum_{i=0}^{N-1} p_i E_i = \sum_{i=0}^{N-1} p_i\ .\frac{m}{\sin^2\theta_i}\ , \tag{2.3}$$

where p_i is the probability of i being the solution. The angles θ_i given in these expressions are known to obey (1.21). Then the following equality is known to be (approximately) true:

$$\sum_{i=0}^{N-1} \theta_i^2 = 4m^2 \sum_{i=0}^{N-1} |c_i|^2 \tag{2.4}$$

Since c_i are the coefficients of a quantum state, normalization condition dictates that the sum at the right-hand side of the equation above is equal to one. Then this equation can be simplified as below:

$$\sum_{i=0}^{N-1} \theta_i^2 = 4m^2 \ . \tag{2.5}$$

The problem of finding the optimal "Symmetric Search with Generic Initial Search" method can be summarized using (2.3) and (2.5). The aim is to find such θ_i and m that, these parameters obey the constraint given in (2.5) and minimize E given as in (2.3).

2.1.2 Optimality Conditions

Now that the problem is defined as an objective function to minimize and a constraint for the parameters to satisfy, the method of Lagrange multipliers can be employed to do the task. The function derived for this purpose, using (2.3) and (2.5), is given below:

$$\mathcal{L} = \sum_{i=0}^{N-1} p_i \frac{m}{\sin^2 \theta_i} + \lambda \left[\sum_{i=0}^{N-1} \theta_i^2 - 4m^2 \right] \ . \tag{2.6}$$

The parameters to compute optimal values to find the minimum expected number of iterations are m, λ and θ_i. It is known that at the optimal point, the variation of function $\mathcal{L}$ must be equal to zero:

$$\delta\mathcal{L} = \frac{\partial \mathcal{L}}{\partial m}\delta m + \frac{\partial \mathcal{L}}{\partial \lambda}\delta\lambda + \sum_{i=0}^{N-1} \frac{\partial \mathcal{L}}{\partial \theta_i}\delta\theta_i \ . \tag{2.7}$$

For variation of $\mathcal{L}$ to be equal to zero, all the partial derivatives on the right-hand side must be separately equal to zero. Let m^*, λ^* and θ_i^* be the optimal values of m, λ and

θ_i respectively. Then the optimality conditions for the method can be summed with the three equations below, which are derived from the partial derivatives of $\mathcal{L}$ with respect to m, λ and θ_i:

$$\sum_{i=0}^{N-1} \frac{p_i}{\sin^2 \theta_i^*} - 8\lambda^* m^* = 0 \ , \tag{2.8}$$

$$\sum_{i=0}^{N-1} \theta_i^{*2} - 4m^{*2} = 0 \ , \tag{2.9}$$

$$m^* \frac{p_i \cos \theta_i^*}{\sin^3 \theta_i^*} - \lambda^* \theta_i^* = 0 \ . \tag{2.10}$$

2.1.3 Numerical Analysis

The equations for optimality are found in the previous subsection but unfortunately, solving this system of equations analytically does not seem possible. Instead, a numerical analysis can be conducted to find the optimum values for a given probability distribution.

The numerical analysis is started by finding an expression for θ_i^* in terms of the other values. To do this; first, (2.10) is arranged as

$$\theta_i^* \frac{\sin^3 \theta_i^*}{\cos \theta_i^*} = p_i \frac{m^*}{\lambda^*} \ . \tag{2.11}$$

Let the function $f(x)$ to be defined as below:

$$f(x) = x \frac{\sin^3 x}{\cos x} \ . \tag{2.12}$$

Then, using this definition and (2.11), θ_i^* can be written as

$$\theta_i^* = f^{-1}(\kappa p_i) \ , \tag{2.13}$$

where κ is equal to m^*/λ^*. If the value of κ is found, all the θ_i values can be found from (2.13). Then inserting θ_i into (2.9) would give the value of m. Once the value of m is known calculating λ is a trivial process. This means that finding the value of κ would give all the optimal values of the parameters.

To find κ an equation is obtained using (2.8) and (2.9):

$$\kappa = \frac{2\sum_{i=0}^{N-1}\theta_i^{*2}}{\sum_{i=0}^{N-1}\frac{p_i}{\sin^2\theta_i^*}} = \frac{2\sum_{i=0}^{N-1}\left[f^{-1}(\kappa p_i)\right]^2}{\sum_{i=0}^{N-1}\frac{p_i}{\sin^2[f^{-1}(\kappa p_i)]}} \,. \tag{2.14}$$

An explicit equation for κ cannot be obtained from (1.15). However, an iterative method to estimate κ can be designed using this equation. For this purpose, let κ_0 be the initial value set for κ at the beginning of this method. At the αth iteration of this method, κ_{new} is computed as

$$\kappa_{new} = \frac{2\sum_{i=0}^{N-1}\left[f^{-1}(\kappa_{\alpha-1} p_i)\right]^2}{\sum_{i=0}^{N-1}\frac{p_i}{\sin^2[f^{-1}(\kappa_{\alpha-1}p_i)]}} \,, \tag{2.15}$$

where $\kappa_{\alpha-1}$ is the κ value set after $\alpha-1$ steps. The new estimation of κ, which is κ_α, is calculated by taking the average of κ_{new} and $\kappa_{\alpha-1}$. This procedure continues until the difference between two successive estimates are smaller than the precision value.

The optimality parameters of this method are computed for 4 different probability distributions. For this purpose, a computer program, which finds a close approximation to κ using the iteration method above and then calculates the optimal values of the other parameters from this approximate κ value, is written. The results gathered from running this program are given in table 2.1.

It is seen that for a uniform distribution, where $p(x) = 1$, the same result is found with [11] as expected. The results at the other rows show that as the skewness of the probability distribution increases, the effectiveness of the search method also increases. That is, there is an inverse proportion between the skewness of the distribution and the expected number of iterations to be applied.

Table 2.1: Results of the numerical analysis for Symmetric Search with Generic Initial States

$p(x)$	m^*	E^*
1	$0.583\sqrt{N}$	$0.690\sqrt{N}$
$2x$	$0.544\sqrt{N}$	$0.656\sqrt{N}$
$3x^2$	$0.495\sqrt{N}$	$0.606\sqrt{N}$
$4x^3$	$0.455\sqrt{N}$	$0.562\sqrt{N}$

Inserting the κ value and the probability distribution into (2.13), θ_i^* values can also be obtained. Since the number of elements in the search space is chosen to be 10^6, showing these values in a table is impractical. However, having this many elements makes the assumption of replacing θ_i with continuous function $\theta(x)$, where x can take values between 0 and 1, possible. The graph of $\sin^2\theta$ that is constructed using this assumption for the θ_i^* values of the distribution $p(x) = 2x$ is given in figure 2.1. From this graph it is seen that in the optimal case, the initial state is created in such a way that the higher indices have higher coefficients.

2.2 Asymmetric Search with Generic Initial States

2.2.1 Definition of the Method

In this search method, the assumption that is imposing using the same number of iterations and the same initial states is relaxed. This results in an increased number of parameters. For this method, different iteration number and final angles exist for each step. This results in a reduced number of expected Grover iterations.

Let m_j be the number of iterations applied at the j^{th} quantum search and $|\psi_j\rangle$ be the

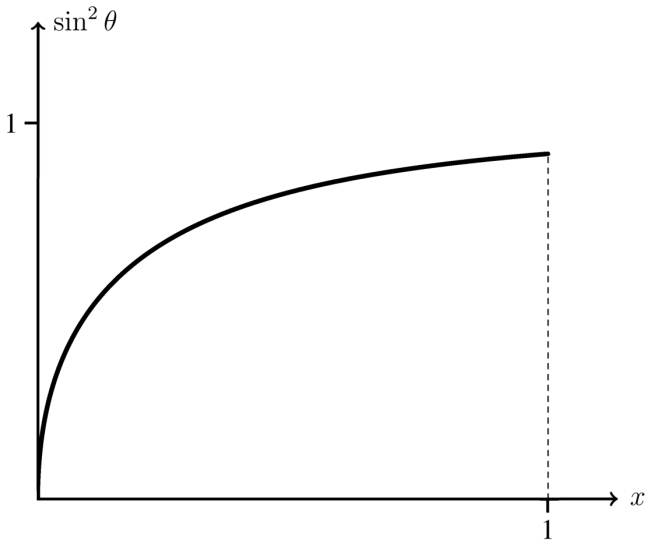

Figure 2.1: Graph of the estimated optimal value of $\sin^2\theta$ for Symmetric Search with Generic Initial States in a set with $p(x) = 2x$

initial state used there. $|\psi_j\rangle$ can be defined as

$$|\psi_j\rangle = \sum_{i=0}^{N-1} c_i^{(j)} |i\rangle \ . \tag{2.16}$$

Then in the case where i is the solution, the angle between the states $|i\rangle$ and $|ns\rangle$ at the end of the iterations of the j^{th} step is named as $\theta_i^{(j)}$. These angles are related to the coefficients of $|\psi_j\rangle$ by the equation below:

$$\theta_i^{(j)} = (2m_j + 1) \arcsin c_i^{(j)} \approx 2m_j c_i^{(j)} \ . \tag{2.17}$$

Now that the parameters of the method are defined, the expressions for expected number of iterations and the constraints can be derived. First, let E_i be the expected number of iterations only if i is the solution. Then E_i can be defined as

$$E_i = m_1 + (1 - \sin^2\theta_i^{(1)})m_2 + (1 - \sin^2\theta_i^{(1)})(1 - \sin^2\theta_i^{(2)})m_3 + ... \tag{2.18}$$

This equation can be organized using product notations. An important thing about the convention that is used here is given below:

$$\prod_{k=1}^{0} \cos^2 \theta_i^{(k)} = 1 \ . \tag{2.19}$$

Using this convention, the definition of E_i can be reorganized as

$$E_i = \sum_{j=1}^{\infty} m_j \prod_{k=1}^{j-1} \cos^2 \theta_i^{(k)} \ . \tag{2.20}$$

Using the definition given above, the expression for the expected number of iterations for the whole method is found as

$$E = \sum_{i=0}^{N-1} p_i E_i = \sum_{i=0}^{N-1} p_i \sum_{j=1}^{\infty} m_j \prod_{k=1}^{j-1} \cos^2 \theta_i^{(k)} \ . \tag{2.21}$$

The constraint equations for this method are similar to the one found for the method in the previous section. The only difference here is that instead of one constraint for the whole process, one constraint for each step is needed. For each step j, the equation given below is recognized:

$$\sum_{i=0}^{N-1} \left[\theta_i^{(j)}\right]^2 = 4m_j^2 \ . \tag{2.22}$$

Then the problem of optimizing the method can be defined as minimizing E given in (2.21) while obeying the constraint equations given in (2.22).

2.2.2 Optimality Conditions

A similar function to the one used to optimize the previous method can be formed using (2.21) and (2.22). Here, the constraint equations can be written as a sum in the optimization function. Then the expression for this function would be

$$\mathcal{L} = \sum_{i=0}^{N-1} p_i \sum_{j=1}^{\infty} m_j \prod_{k=1}^{j-1} \cos^2 \theta_i^{(k)} + \sum_{j=1}^{\infty} \lambda_j \left\{ \sum_{i=0}^{N-1} \left[\theta_i^{(j)}\right]^2 - 4m_j^2 \right\} \ . \tag{2.23}$$

Apparently, this function depends on infinitely many parameters. Using Lagrange multipliers method and finding the optimal values of the parameters is not valid for such a function. To find approximations to the optimality parameters and expectation value of this method, a different approach is taken in the next subsection.

2.2.3 Finite Number of Steps

Solving the optimality conditions gathered in the previous subsection is not possible. To be able to use numerical methods for finding approximations to the optimality values, a similar method with finite number of steps is created here. The effect of a step j on the expected number of iterations decreases as j increases, because the coefficient in front of m_j decreases with increasing j as it can be observed in (2.21). This means that stopping the search after a sufficient number of steps s would give a very similar result to applying infinitely many steps. For the method to be complete, it is assumed that the s^{th} step is a quantum search with a certain result (which means that a search that is done by applying $\pi/4\sqrt{N}$ Grover iterations to $|\psi_0\rangle$).

Finding an expression for the expected number of iterations involves very similar procedures to the ones in the previous subsection so they are omitted here. The final result is given below:

$$E = \sum_{i=0}^{N-1} p_i \sum_{j=1}^{s} m_j \prod_{k=1}^{j-1} \cos^2 \theta_i^{(k)} \ . \tag{2.24}$$

The constraint equation for jth step is the same as the one found in the previous subsection. The only difference is that there are only $s-1$ of them. The last step does not need a constraint because it is already known that $m_s = (\pi/4)\sqrt{N}$ and $\theta_i^{(s)} = \pi/2$. In light of this, the optimization function for this method is found to be

$$\mathcal{L} = \sum_{i=0}^{N-1} p_i \sum_{j=1}^{s} m_j \prod_{k=1}^{j-1} \cos^2 \theta_i^{(k)} + \sum_{j=1}^{s-1} \lambda_j \left\{ \sum_{i=0}^{N-1} \left[\theta_i^{(j)}\right]^2 - 4m_j^2 \right\} \ . \tag{2.25}$$

Using the partial derivatives of function $\mathcal{L}$, the equations for the optimal values m_j^*, λ_j^* and $\theta_i^{(j)*}$ can be found. These equations are given below:

$$\sum_{i=0}^{N-1} p_i \prod_{k=1}^{j-1} \cos^2 \theta_i^{(j)*} - 8\lambda_j^* m_j^* = 0 \ , \tag{2.26}$$

$$\sum_{i=0}^{N-1} \left[\theta_i^{(j)*}\right]^2 - 4m_j^{*2} = 0 \ , \tag{2.27}$$

$$p_i \cos\theta_i^{(l)*} \sin\theta_i^{(l)*} \sum_{j=l+1}^{s} m_j^* \prod_{\substack{k<j \\ k\neq l}} \cos^2 \theta_i^{(k)*} - \lambda_l^* \theta_i^{(l)*} = 0 \ . \tag{2.28}$$

Now that there are a finite number of parameters and optimality equations are known, a numerical analysis can be conducted.

2.2.4 Numerical Analysis

For a finite number of steps, the optimality conditions are found in the previous subsection. It is quite obvious that trying to find an analytical solution to this system of equations is the right way to proceed. Instead, the optimality parameters can be approximated iteratively to find an estimate to the minimum of the expectation value. This way, an opinion about the limits of the method can be formed.

An iterative method can be created to find an estimation for λ_j^*. Then all the other parameters can be derived using the value obtained by this method. To do this, an expression for λ_j^* that does not depend on the other parameters must be found. The process of finding such an expression is given below.

Let γ_j and $\beta_i^{(l)}$ to be defined as

$$\gamma_j = \sum_{i=0}^{N-1} p_i \prod_{k=1}^{j-1} \cos^2 \theta_i^{(k)*} \ , \tag{2.29}$$

$$\beta_i^{(l)} = \sum_{j=l+1}^{s} m_j^* \prod_{\substack{k<j \\ k\neq l}} \cos^2 \theta_i^{(k)*} \ . \tag{2.30}$$

Using one of these newly defined parameters and (2.28), an expression for $\theta_i^{(k)*}$ is obtained

$$\frac{\cos\theta_i^{(j)*}\sin\theta_i^{(j)*}}{\theta_i^{(j)*}} = \frac{\lambda_j^*}{p_i\beta_i^{(j)}} \ . \tag{2.31}$$

Now let the function $f(x)$ to be defined as

$$f(x) = \frac{\cos x \sin x}{x} \ . \tag{2.32}$$

Using the inverse of this function, $\theta_i^{(j)*}$ can be written in terms of λ^*, p_i and $\beta_i^{(j)}$

$$\theta_i^{(j)*} = f^{-1}\left(\frac{\lambda_j^*}{p_i\beta_i^{(j)}}\right) \ . \tag{2.33}$$

Then, an expression for λ_j^* can be found using (2.26) and (2.29)

$$\lambda_j^* = \frac{\gamma_j}{8m_j^*} \ . \tag{2.34}$$

Finally, by using (2.27), (2.33) and (2.34), the expression that is to be used for iteratively finding λ_j^* is found as

$$\lambda_j^* = \frac{\gamma_j}{\sqrt{\sum_{i=0}^{N-1}\left[f^{-1}\left(\frac{\lambda_j^*}{p_i\beta_i^{(j)}}\right)\right]^2}} \ . \tag{2.35}$$

Now that the required equation is at hand, creating a procedure for iteration is easy. The process begins with setting an initial value for λ_j, namely $\lambda_{j,0}$. At the α^{th} step of the iteration $\lambda_{j,new}$ value is computed using the $\lambda_{j,\alpha-1}$ value, which is the estimated value of λ_j after $\alpha - 1$ iteration steps. This $\lambda_{j,new}$ value is found using the equation below:

$$\lambda_{j,new} = \frac{\gamma_j}{\sqrt{\sum_{i=0}^{N-1} \left[f^{-1}\left(\frac{\lambda_{j,\alpha-1}}{p_i \beta_i^{(j)}} \right) \right]^2}} \,. \tag{2.36}$$

After finding $\lambda_{j,new}$, $\lambda_{j,\alpha}$ is set as the arithmetic mean of $\lambda_{j,new}$ and $\lambda_{j,\alpha-1}$. This procedure continues until the difference between two successive estimations is lower than a pre-defined precision value. Let this procedure be called as λ-iteration.

There is an important issue to be addressed that is encountered in the iteration. The graph of $f(x)$ for the relevant interval of x, which is $[0, \pi/2]$, is given in figure 2.2. From the graph it is seen that the range of $f(x)$ is the set $[0, 1]$. This means that $f^{-1}(y)$ is only defined for $0 \leq y \leq 1$. In the iteration, for certain values of i and j, $f^{-1}(y)$ is tried to be computed for some $y > 1$. This means that for some i and j, $\theta_i^{(j)}$ does not obey the optimality condition given in (2.28) and the optimum value of this parameter is in one of the boundaries. This kind of $\theta_i^{(j)}$ are set to 0 instead of $\pi/2$. Some $\theta_i^{(j)}$ values being equal to 0 makes sense since some of the indices are not searched in some steps.

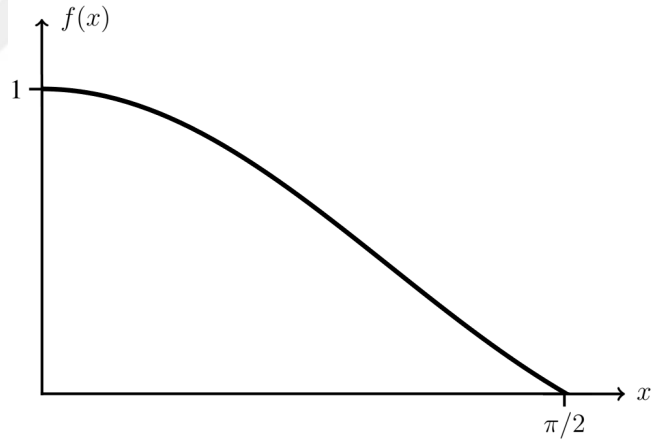

Figure 2.2: Graph of $f(x) = \frac{\cos x \sin x}{x}$ for the interval $0 \leq x \leq \pi/2$

Unlike the method of the previous section, estimating a single parameter by iteration and calculating the other parameters using that one does not complete the numerical analysis for this case. In this method there are s steps with $s - 1$ of them to optimize. Another problem is that $\beta_i^{(j)*}$ and γ_j depends on the parameters of the steps other than step j. This means that optimizing each step only once does not give a good

estimation of the minimum expectation number. Instead, for each step the optimum values of the parameters must be found repeatedly after finding the new estimations of the parameters of the other steps.

In the numerical analysis, a procedure appropriate to the nature of the problem is used. First, the optimal parameters for Symmetric Search with Generic Initial States are found and all m_j^* and $\theta_i^{(j)^*}$ values, except for $j = s$, are set to these values. Then γ_1 and $\beta_i^{(1)}$ are computed using the current values and the initial value of the λ-iteration, $\lambda_{1,0}$, is determined by inserting $\lambda_{1,0}$ in place of λ_1^* in (2.34). The final value obtained in the λ-iteration is set as λ_1^* and the parameters m_1^* and $\theta_i^{(j)*}$ are recalculated using this value (Here some of the $\theta_i^{(j)*}$ might be found that are not obeying the optimality conditions as described above so, these are accepted to be at the boundary $\theta_i^{(j)*} = 0$). Then this procedure is repeated for the steps $j = 2, 3, ..., s-1$. After the parameters for all the steps are updated, the expected number of Grover iterations is calculated using (2.24).

As explained, computing the optimal values only for once does not suffice. Thus, the procedure of updating the estimations of the optimality parameters and recalculating E^* is conducted iteratively. After each iteration it is checked whether the newly found expectation value is smaller than the previous ones and if it is, the amount of improvement is calculated. After this amount is smaller than a pre-defined precision value, the procedure is concluded.

The analysis is done using $s = 10$ number of steps, the last one being the original Grover search. Estimated minimum of the expected number of iterations for each probability distribution in consideration are given in table 2.2.

The results show that this method does not make any improvement for sets with uniform distribution as expected. However, for all the other distributions, it gives significantly lower expected number of iterations than its symmetric counterpart. These results can be improved (except for the one for probability distribution $p(x) = 1$) by using more steps but these improvements might not be as significant as changing the first three decimal places since the effect of the latter steps are less than the former ones.

Table 2.2: Estimated minimum expectation values for Asymmetric Search with Generic Initial States for different probability distributions

$p(x)$	E^*
1	$0.690\sqrt{N}$
$2x$	$0.627\sqrt{N}$
$3x^2$	$0.558\sqrt{N}$
$4x^3$	$0.507\sqrt{N}$

Since there are 10 m_j^* values for each probability distribution, these are omitted in table 2.2. To give a better insight of the method, estimated optimal values of m_j for the probability distribution $p(x) = 2x$ are given in table 2.3.

During the analysis, $\theta_i^{(j)*}$ values are also estimated. Like the analysis of the previous method the sets to be searched are assumed to have 10^6 elements for precision so, using a continuous graph to show these values is more convenient. The optimal $\sin^2\theta$ values for the first three steps of the method for $p(x) = 2x$ are given in the figure 2.3. The optimal initial states of this method are much more complicated than the ones found in the previous section. Preparing such complicated states may not always be possible but this analysis is useful in a way that it shows the limits for the quantum search in sets with prior knowledge.

Table 2.3: Estimated optimum values of m_j Asymmetric Search with Generic Initial States in a set with probability distribution $p(x) = 2x$

j	m_j^*	j	m_j^*
1	$0.495\sqrt{N}$	6	$0.479\sqrt{N}$
2	$0.455\sqrt{N}$	7	$0.477\sqrt{N}$
3	$0.486\sqrt{N}$	8	$0.483\sqrt{N}$
4	$0.484\sqrt{N}$	9	$0.493\sqrt{N}$
5	$0.472\sqrt{N}$	10	$0.785\sqrt{N}$

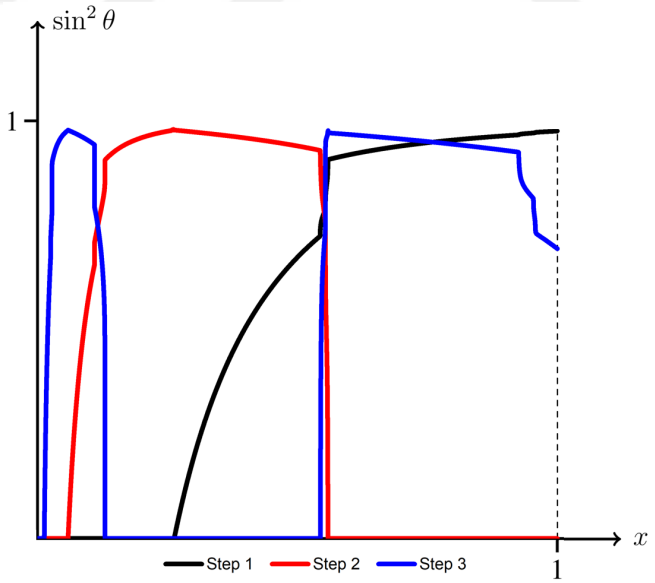

Figure 2.3: Graph of the estimated optimal value of $\sin^2\theta$ for Asymmetric Search with Generic Initial States in a set with $p(x) = 2x$

CHAPTER 3

QUANTUM SEARCH WITH EASY-TO-PREPARE INITIAL STATES

In this chapter the goal is to create methods that can be realistically used in a quantum computer to conduct quantum search faster. Thus, the cost of preparing an initial step is not neglected and states that are known to be prepared with a low computing power are considered. More specifically, states that are superpositions of only some of the indices with equal coefficients are used. It is proven in the next subsection that the number of quantum gates required to prepare this kind of states is only higher by a constant than number of gates that is used to prepare $|\psi_0\rangle$.

The methods used in this chapter can be described as follows. Let r be a real number satisfying the condition $0 < r \leq 1$. First a quantum search with m_1 Grover iterations is applied to a subset, with rN elements of the search space, which has N elements. Here the important part is that the subset is chosen to include the rN elements with the highest probabilities of being the solution (Since all the probability distributions used in this thesis are all monotonically non-decreasing, the highest rN indices are included.). If the solution is not found after this step, another quantum search with m_2 iterations is applied to the subset including the remaining $(1-r)N$ indices. The variable r that is used here is named as the fraction value and the methods working as the way described here are named as Partition Search methods.

The next section in this chapter is about the complexity of preparing the initial steps required for the methods to be introduced. Each of the other three sections is about a different Partition Search method. The first method introduced in this chapter is named as Exact Partial Search. This name is chosen because this method finds the solution definitely after two consecutive quantum searches. The number of iterations in the worst case is finite but, it improves the expected number of iterations in even a

set with uniform probability distribution. The other two methods concentrate on increasing the expected number of iterations in sets with prior knowledge only. These two are different from each other in the number of parameters. First of them, which is named as Symmetric Partition Search, uses the same parameter to determine the number of iterations for each step. The other method, namely Asymmetric Partition Search, aims to decrease the expected number of iterations by allowing to apply independent number of iterations to different partitions.

3.1 Preparing Initial States

The general procedure is explained above but, there is a remaining question. Can one prepare the states to be used for these methods easily? To answer this question, let $|\psi_1\rangle$ and $|\psi_2\rangle$ be the initial states to be used for searching the first and the second partitions respectively. These states can be defined as

$$|\psi_1\rangle = \frac{1}{rN} \sum_{i=(1-r)N}^{N-1} |i\rangle \ , \qquad |\psi_2\rangle = \frac{1}{(1-r)N} \sum_{i=0}^{(1-r)N-1} |i\rangle \ . \tag{3.1}$$

Focus of the following analysis is preparation of $|\psi_1\rangle$. Assume that the number of elements in the search space is a perfect power of 2 and can be expressed as $N = 2^n$. Also assume that the fraction value can be written as $r = 2^{-m}$ where m is a positive integer. Then the state which is a superposition of the highest rN indices can be prepared using the circuit shown in figure 3.1. Here the upper m gates are NOT gates used to convert $|0\rangle$ into $|1\rangle$ and the lower $n - m$ gates are Hadamard gates. This means that by only changing the first m Hadamard gates in the circuit to prepare $|\psi_0\rangle$ with m NOT gates, the required initial step can be prepared. The state prepared using this circuit is given as

$$|\psi_1\rangle = |1\rangle^{\otimes m} \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}} \right)^{\otimes m-n} \ . \tag{3.2}$$

It was shown in the first chapter that assuming N to be a perfect power of 2 does not make us lose any generality. However, this condition is not true for r. In each

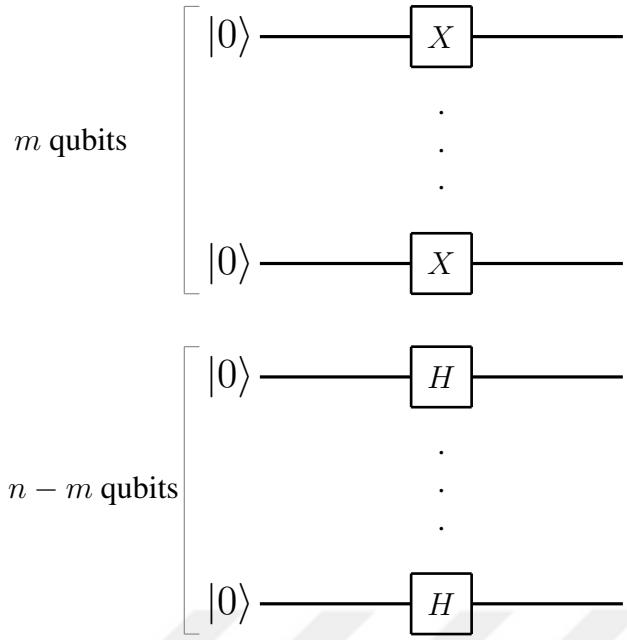

Figure 3.1: Circuit used to create a superposition of highest $2^m N$ indices

method, an optimal value for r which minimizes the expected number of iterations is to be found. Thus, rounding r to the closest perfect power is not acceptable. Instead, the number of operations needed to create a state depending on the precision of r can be found. For this purpose let the fraction value be $r = (3/4)2^{-m}$ where m is a positive integer. Then the initial step can be written as

$$|\psi_1\rangle = |1\rangle^{\otimes m} \left(\frac{|01\rangle + |10\rangle + |11\rangle}{\sqrt{3}}\right) \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)^{\otimes n-m-2} . \tag{3.3}$$

This state is a superposition of $3 \times (n-m-2) = (3/4)2^{n-m}$ indices, since the middle term is a superposition of 3 and the last terms is a superposition of 2^{n-m-2}. The most significant m qubits are all set to be $|1\rangle$ and the next most significant 2 qubits are set as a superposition of 3 highest indices of 4. Consequently, the given state is the superposition of the highest $(3/4)2^{n-m}$ indices.

To prepare this state, each of the first m qubits are applied NOT gates and each of the last $n-m-2$ gates are applied Hadamard gates so, no additional computing power for these. However, there are additional operations to prepare the remaining 2 qubits. The specifics of these operations are not important but the upper limit for the

number of these can be found. In [13] it is proven that any state with n qubits can be prepared with $O(2^n)$ complexity. This means that for this example preparing the middle 2 qubits requires application of at most 4 CNOT gates (In reality one CNOT gate is enough to prepare a 2-qubit state but the formula shows the asymptotic limit.).

Another important thing is to determine how many qubits of the initial state require additional computation. In the example above it was shown to be 2. The binary representation of $r = (3/4)2^{-m}$ is $0.00...011$ where there are m zeroes before the last two digits. For $r = c2^{-m}$, where c is a rational number between 0 and 1 which is not a multiple of any perfect power of 2, the binary representation would require m zeroes followed by some adequate number of digits to represent c. Then the number of digits required to represent c gives the number of qubits to be additionally operated. This number can be determined from a given value of r. Since the aim of this analysis is to set an upper limit for the number of additional operations, the limiting case is observed where $m = 0$ and the number of binary digits needed to represent c and r are equal.

All the r values found in this thesis are shown with a precision value of 10^{-3}. The number of binary digits needed to write them can be found to be $-\lceil \log 10^{-3} \rceil = 10$. Then using the results of [13], the upper limit for the additional CNOT gates to be applied for preparing $|psi_1\rangle$ for a given r value is equal to $2^{-\lceil \log 10^{-3} \rceil} = 1024$. To search a big database, this many of additional operations is indeed negligible. When the size of the database is large enough, r values with even higher precision can be used.

3.2 Exact Partition Search

3.2.1 Definition of the Method

This search method is about dividing the search space into two partitions and applying original Grover search to them consecutively. Number of iterations in a Grover search, which is equal to $(\pi/4)\sqrt{N}$ for a set with N elements, guarantees to find a solution in the case that there is one. This means that if a solution is not found in

the first partition, it is certainly in the second one. Thus, this method uses 2 search procedures in the worst case and 1 in the best case.

Let r be the fraction value and N be the number of elements in the search space, so that the first partition has rN elements and the second one has $(1-r)N$. Also let p_1 be the probability of the solution being in the first partition. Then the probability of the solution to be in the second partition is $1-p_1$. Since definite results are aimed, number of iterations applied at the first and the second steps are $(\pi/4)\sqrt{rN}$ and $(\pi/4)\sqrt{(1-r)N}$ respectively. In the light of this information, the expected number of iterations is found to be

$$E = \frac{\pi}{4}\sqrt{rN} + (1-p_1)\frac{\pi}{4}\sqrt{(1-r)N}\ . \tag{3.4}$$

3.2.2 Optimality Calculations

To use this method with maximum efficiency, the optimum value of r, which minimizes E defined in (3.4), must be found. The term p_1, which appears in the equation for expectation value, depends on the probability distribution and r. This means that the optimal r value must be computed separately for a given probability distribution.

The probability distributions to be analyzed here are the same with the ones used in the previous chapter. Using the equations for these distributions, p_1 can be written as a function of r. By this way, the expected number of iterations can be written as a function of only r. In the first step of this method (and for all the other Partition Search methods), rN indices with the highest probability of being the solution, which means the rN highest indices, are searched. Then it can be shown that $p_1(r)$ can be found by the equation below:

$$p_1(r) = \int_{1-r}^{1} p(x)dx\ . \tag{3.5}$$

As E can be shown to be a function of solely r, the minimum of this function can be found by the differential equation

$$\frac{dE}{dr} = 0 \ . \tag{3.6}$$

This equation can be solved even by analytical methods but, finding the exact value of r that satisfies this equation requires numerical computation. By such calculations, minimum of E and the optimal value of r ensuring this minimality, E^* and r^* respectively, are given for each probability distribution in table 3.1.

Table 3.1: Results of the numerical analysis for Exact Partition Search

$p(x)$	r^*	E^*
1	0.873	$0.769\sqrt{N}$
$2x$	0.593	$0.688\sqrt{N}$
$3x^2$	0.465	$0.624\sqrt{N}$
$4x^3$	0.389	$0.575\sqrt{N}$

It can be observed that for sets without prior knowledge, where $p(x) = 1$, this method finds the solution using an expected number of iterations which is about 98% of the number of iterations used for a Grover search. In the worst case the required number of iterations is $(\pi/4)(\sqrt{r} + \sqrt{1-r})\sqrt{N}$. This might be considered as an improvement considering the method decreases the expected number with only increasing the iteration number by 29% in the worst case for $p(x) = 1$. There is a known method, which is introduced in [11], having better expected number of iterations for such sets. However, for that method even finding the solution in the second step, where the possible number of steps to be applied is infinite, means a 74% more iterations than Grover search. Thus, the performance of Exact Partition Search is between Grover Search and the method in [11] in both persistence and average performance.

The method also makes improvements for sets with prior knowledge as it is apparent from the table. These improvements are also without much compromise from the

worst case-scenario. There are methods introduced in this thesis with better expected number of iterations for such sets, but Exact Partition Search can be applied when a more consistent search time is targeted.

3.2.3 Limits of the Method

In the previous subsection, it is shown that dividing the search space into partitions provides advantages. This raises the idea that the method can be improved further by partitioning the search space in the latter steps. It is known that if a solution is not found in the first step, then it is definitely in the second partition. Thus the search in the second partition can be thought of a completely new search of a set with $(1-r)N$ elements. Then partitioning this set would also make an improved search. If a solution is not found again, the next step can be thought as a new search in a set with $(1-r)^2N$ elements and so on. This repeated partitioning can be done as many times as desired depending on the size of the set to be searched. In the case where $N \to \infty$, this operation can be assumed to be repeated infinite times. In that case, the expected number of iterations is found to be

$$E = \frac{\pi}{4}\sqrt{rN} + (1-p_1)\frac{\pi}{4}\sqrt{r(1-r)N} + (1-p_1)^2\frac{\pi}{4}\sqrt{r(1-r)^2N} + ... \qquad (3.7)$$

This expression can be simplified using the definition of geometric series.

$$E = \frac{\pi}{4}\sqrt{rN}\sum_{j=0}^{\infty}\left[(1-p_i)\sqrt{1-r}\right]^j = \frac{\pi\sqrt{rN}}{4\left[1-(1-p_i)\sqrt{1-r}\right]} \qquad (3.8)$$

The minimum of this expression can be found using a similar method with the one in the previous section since, E is solely a function of r. The optimal values of r and E obtained by such method is given in table 3.2.

From the table, it is obvious that partitioning the search space infinitely results in an enhanced performance. The expected values that are shown here are not possible for sets with finite number of elements but, they show the mathematical limits. Before

Table 3.2: Results of the numerical analysis for the limits of Exact Partition Search

$p(x)$	r^*	E^*
1	0.866	$0.769\sqrt{N}$
$2x$	0.533	$0.674\sqrt{N}$
$3x^2$	0.377	$0.596\sqrt{N}$
$4x^3$	0.291	$0.538\sqrt{N}$

beginning to use this method, the number of repeated partitioning can be determined depending on the number of elements in the search space.

The advantage of partitioning the set further increases as the skewness of the probability distribution increases. In sets with uniform distribution, where $p(x) = 1$, the amount of decrease is in the order of $10^{-4}\sqrt{N}$. This amounts to only about 0.1% improvement. For $p(x) = 4x^3$ the percentage of the change in the expectation value is up to about 6%.

3.3 Symmetric Partition Search

3.3.1 Definition of the Method

Symmetric Partition Search is a generalization of the Exact Partition Search where the number of iterations to be applied at any step does not guarantee finding a solution. However, in this method, the coefficient determining the number of iterations are equal. The number of partitions here is two again. Since the searches in this method are not exact; if a solution is not found in both the first and the second partitions, the first partition is searched again and then the second partition again. This repetition goes on and on until the solution is obtained. This makes the number of iterations

to be infinite in the worst case, like the methods in chapter 2 or in [11], but it also decreases the expected number of iterations for sets with prior knowledge.

Let r be the fraction value, p_1 be the probability of the solution being in the first partition for given r and, μ be the parameter such that the number of iterations applied to a partition is μ times the square root of the number of elements in that partition (This means applying $\mu\sqrt{rN}$ iterations to the first partition and $\mu\sqrt{(1-r)N}$ to the second.). Additionally let q be the probability of finding the solution after the iterations are applied, provided that it is located in the partition that is currently being searched. Then the expected number of iterations becomes

$$E = \mu\sqrt{rN} + [(1-p_1) + p_1(1-q)]\mu\sqrt{(1-r)N} + (1-q)\mu\sqrt{rN} + ... \quad (3.9)$$

Here, the first term is about the first search, which is certainly done, that is applied to the first partition. The second term is about searching the second partition for the first time and the part in square brackets show the probability of this search to be applied. This probability is the sum of two probabilities. The first one is the probability of the solution not being in the first partition $(1-p_1)$ and the second one is the probability of the solution being in the first partition but the search not being able to find it $p_1(1-q)$. Searching the first partition for the second time is done if a solution is not found in the first two searches, which is done with probability $p_1(1-q)+(1-p_1)(1-q) = (1-q)$. The following terms follow the same logic. This infinite series can be organized as

$$E = \mu\sqrt{N}\left\{\sqrt{r} + [(1-p_1) + p_1(1-q)]\sqrt{1-r}\right\}\sum_{j=0}^{\infty}(1-q)^j \ . \quad (3.10)$$

Then using the geometric series, it takes its final form

$$E = \frac{\mu\sqrt{N}}{q}\left\{\sqrt{r} + [(1-p_1) + p_1(1-q)]\sqrt{1-r}\right\} \ . \quad (3.11)$$

3.3.2 Optimality Calculations

In the previous section, it was shown that p_1 can be written as a function of r. It can also be proven that q is a function of μ. In (1.17), the probability of finding the solution, provided that there is only one, after m iterations were given. Replacing m with $\mu\sqrt{N}$ gives

$$q = \sin^2(2\mu) \ . \tag{3.12}$$

Now that it is known that E is a function of only r and μ, the values of these parameters making E minimum can be found. This is done by a similar procedure used for the previous method. The difference is now E is not a function of a single variable so, partial derivatives are employed. At the minimum point of E both its partial derivatives with respect to r and m are equal to zero (to make its variation zero). The optimal values of these parameters can be computed numerically. The optimal values of r, μ and minimum value of E are given in table 3.3.

Table 3.3: Results of the numerical analysis for Symmetric Partial Search

$p(x)$	r^*	μ^*	E^*
1	1	0.583	$0.690\sqrt{N}$
$2x$	0.597	0.667	$0.639\sqrt{N}$
$3x^2$	0.464	0.680	$0.584\sqrt{N}$
$4x^3$	0.387	0.689	$0.542\sqrt{N}$

For sets without prior knowledge, where $p(x) = 1$, the computation foresees that the best method is searching without partitioning as in [11]. This result was an expected one because there is no known probabilistic superiority between the indices so

no advantage should be gained by giving some indices priority to be searched. For monotonically increasing probability distributions, this method gives improved expectation values for the number of iterations. The improvements for these probability distributions are better than that of Definite Partition Search.

3.4 Asymmetric Partition Search

3.4.1 Definition of the Method

This method is a further generalization of Symmetric Partition Search. Thus, the procedure is quite alike. Search space is divided into two partitions and quantum search is applied sequentially. The first partition is searched by using some number of iterations at the start. If a solution is not found in that search, the second partition is searched. Neither step requires to find the solution with certainty so; if the solution is not found after the first two searches, the first partition is searched again. The difference here is that the coefficients defining the number of iterations in each step can be different from each other. By adding one more degree of freedom, a further improvement is targeted.

As always let r be the fraction value that determines the number of elements in each partition and p_1 be the probability of the solution being in the first partition. Then let parameters μ_1 and μ_2 be in such a way that the quantum searches for partition 1 and 2 uses $\mu_1\sqrt{rN}$ and $\mu_2\sqrt{(1-r)N}$ iterations respectively. Also define q_1 and q_2 as the probabilities of finding a solution in first and the second partitions, in the case that it is located in the partition that is searched. Using these parameters, the expectation value of the total number of Grover iterations to be applied is found as

$$\begin{aligned} E = \mu_1\sqrt{rN} + [(1-p_1) + p_1(1-q_1)]\mu_2\sqrt{(1-r)N} \\ +[p_1(1-q_1) + (1-p_1)(1-q_2)]\mu_1\sqrt{rN} + ... \end{aligned} \tag{3.13}$$

The first search is applied certainly, so the first term in the sum only includes the number of iterations. The probability coefficient on the second term implies that the second partition is searched if the solution is not found after searching the first

partition. This probability consists of the probability of the solution not being in the first partition, $1-p_1$, and of the solution being in the first partition but not being able to be found by the first search, $p_1(1-q_1)$. After searching the two partitions for the first time; if the solution is not found, the first partition is searched again. This is shown by the third term in the sum. The probability of not finding the solution in the first two search is the total of the probabilities of the solution being in the first partition but the first search not being able to find it, $p_1(1-q_1)$, and of the solution being in the second partition but the second search not being able to find it, $(1-p_1)(1-q_2)$. The other terms in the sum are found with the same approach. This equation can be written in a more compact form as

$$E = \sqrt{N}\left\{\mu_1\sqrt{r} + [(1-p_1) + p_1(1-q_1)]\mu_2\sqrt{(1-r)}\right\} \sum_{j=0}^{\infty}[p_1(1-q_1) + (1-p_1)(1-q_2)]^j \ . \tag{3.14}$$

Using geometric series expansion, this equation takes its final form

$$E = \frac{\sqrt{N}\left\{\mu_1\sqrt{r} + [(1-p_1) + p_1(1-q_1)]\mu_2\sqrt{(1-r)}\right\}}{1 - p_1(1-q_1) - (1-p_1)(1-q_2)} \ . \tag{3.15}$$

3.4.2 Optimality Calculations

It can be shown that the expected number of iterations depends on only three parameters. The explicit dependency of p_1 on r is shown in (3.5). Following the logic in the previous section, expressions for q_1 and q_2 can be written as below:

$$\begin{aligned} q_1 &= \sin^2(2\mu_1) \ , \\ q_2 &= \sin^2(2\mu_2) \ . \end{aligned} \tag{3.16}$$

Now that it is known that E depends on the free variables r, μ_1 and μ_2; the values of these minimizing E can be found. As in the previous sections, variational method is used for finding the optimal values r^*, μ_1^* and μ_2^* and the exact values of them are

found using computation. The minimum value of E and the optimal values of the related parameters for each probability distribution are given in the table 3.4.

Table 3.4: Results of the numerical analysis for Asymmetric Partition Search

$p(x)$	r^*	μ_1^*	μ_2^*	E^*
1	1	0.583	-	$0.690\sqrt{N}$
$2x$	0.593	0.669	0.614	$0.638\sqrt{N}$
$3x^2$	0.454	0.681	0.563	$0.580\sqrt{N}$
$4x^3$	0.375	0.686	0.506	$0.535\sqrt{N}$

Just like Symmetric Partition Search, Asymmetric Partition Search also cannot create better results than that of [11] in the case of uniform probabilities. Since it is shown that partitioning the search space does not supply any improvement in the $p(x) = 1$ case, it is not a surprise to see using independent iteration numbers does not provide any advantage. However, for other probability distributions that are examined, adding one more degree of freedom obviously gives improved results. The performance difference between the Partition Search methods increases as the probability function gets steeper. This shows the superiority of this method over the one in the previous section for sets with prior knowledge.

CHAPTER 4

CONCLUSION

Quantum computers are proven to provide a great advantage for search problem in the original paper of L. K. Grover [7]. It is also shown that an improvement to Grover search by a constant factor can be made for any set in [11]. In this thesis, it is additionally proven that quantum search can be improved further when there is a single solution and the probability distribution of the indices for being the solution is known (and not uniform).

The methods of the second chapter are generally not applicable, since the initial required states are not easily prepared. However, analyzing these methods (especially the second method introduced there) informs about the limits of possible advancements for the sets with known probability distributions. The results of Asymmetric Search with Generic Initial States set the lowest possible expectation value for all studied distributions. Any sub-optimal method employing different initial state then $|\psi_0\rangle$ is known to perform with an expectation value greater than or equal to the one that is computed for that method.

Partition Search methods are examples of such sub-optimal procedures. They are specifically designed as easily applicable methods for monotonically non-decreasing probability distributions. Exact Partition Search can be applied to slightly decrease the expected number of iterations for sets with any kind of non-decreasing probability distribution (including uniform distribution) without compromising much from the worst-case scenario. Asymmetric Partition Search hits the target of converging the limit results using states that can be prepared easily. Note that these methods are also applicable to the sets with non-increasing probability distributions by only reversing which part of the set to be searched (by searching the rN elements with the lowest

indices for a set with N elements).

The numerical analyses here are conducted for non-decreasing probability distributions, but the optimality conditions of the methods in the second chapter are appliable to any distribution. To find the optimality values and the optimum initial states for any other distribution, the numerical analyses can be conducted using the same procedures of sections 2.1 and 2.2. The methods of the third chapter, however, are only designed for monotonically non-increasing or non-decreasing distributions as it was explained. For a different distribution than the ones analyzed here, which obeys one of these conditions, the same optimality conditions also hold, and the optimality parameters can be computed using them.

The limits of quantum computers are not known to mankind yet. There might be new quantum algorithms to be discovered that can solve some problems easily that are known to be very complex but this should not be the sole focus of the research about quantum algorithms. Small improvements on such fundamental problems like the search problem have a potential to make a big impact. Improving the known methods for this kind of problems are worthy of a pursuit. Better methods for different probability distributions can be developed to solve different kind of search problems or the problems that can be reduced to search problem, like numerical optimization.

REFERENCES

[1] R. P. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, no. 6, pp. 467–531, 1982.

[2] R. P. Feynman, "Quantum mechanical computers," *Foundations of Physics*, vol. 16, pp. 507–531, 1986.

[3] D. Deutsch, "Quantum theory, the church-turing principle and the universal quantum computer," *Proceedings of the Royal Society A*, vol. 400, pp. 97–117, 1985.

[4] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation," *Proceedings of the Royal Society A*, vol. 439, pp. 553–558, 1992.

[5] D. R. Simon, "On the power of quantum computation," in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, IEEE, nov. 1994.

[6] E. Bernstein and U. Vazirani, "Quantum complexity theory," in *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, ACM, nov. 1994.

[7] P. W. Shor, "Polynomial-time algorithms for quantum computation: Discrete logarithms and factoring," in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, IEEE, nov. 1994.

[8] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, ACM, july 1996.

[9] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Physical Review Letters*, vol. 79, no. 2, pp. 325–328, 1997.

[10] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, "Strenghts and weaknesses of quantum computing," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1510–1523, 1997.

[11] M. Boyer, G. Brassard, P. Høyer, and A. Tapp, “Tight bounds on quantum searching,” *Fortschritte der Physik*, vol. 46, no. 4-5, pp. 493–505, 1998.

[12] I. Flores, *Computer Storing*, pp. 40–46. Englewood Cliffs, N.J: Prentice-Hall, 1969.

[13] M. Plesch and Časlav Brukner, “Quantum state preperation with universal gate decompositions,” *Physical Review A*, vol. 83, 2011.