A Hybrid Quantum Search Engine: A Fast Quantum Algorithm for Multiple Matches

Ahmed Younes* Jon Rowe † Julian Miller ‡
School of Computer Science Department of Electronics
University of Birmingham University of York

September 6, 2006

Abstract

In this paper we will present a quantum algorithm which works very efficiently in case of *multiple matches* within the search space and in the case of few matches, the algorithm performs classically. This allows us to propose a *hybrid quantum search engine* that integrates Grover's algorithm and the proposed algorithm here to have *general performance* better that any pure classical or quantum search algorithm.

1 Introduction

Quantum computers [6, 8, 12] are probabilistic devices, which promise to do some types of computation more powerfully than classical computers [3, 14]. Many quantum algorithms have been presented recently, for example, Shor [16] presented a quantum algorithm for factorising a composite integer into its prime factors in polynomial time. Grover [10] presented an algorithm for searching unstructured list of N items with quadratic speed-up over algorithms run on classical computers.

*Birmingham, Edgbaston, B15 2TT, United Kingdom , axy@cs.bham.ac.uk
†Birmingham, Edgbaston, B15 2TT, United Kingdom , jer@cs.bham.ac.uk
‡York, Heslington, YO10 5DD, United Kingdom, jfm@ohm.york.ac.uk

Grover's algorithm inspired many researchers, including this work, to try to analyze and/or generalize his algorithm [4, 1, 9, 11, 5]. Grover's algorithm is proved to be optimal for a single match within the search space, although the number of iterations required by the algorithm increases; i.e. the problem becomes harder, as the number of matches exceeds half the number of items in the search space [13] which is undesired behaviour for a search algorithm since the problem is expected to be easier. In this paper we will present a fast quantum algorithm, which can find a match among multiple matches within the search space after few iterations faster than any classical or quantum algorithm although for small number of matches the algorithm behaves classically.

This leads us to proposing a hybrid search engine that includes Grover's algorithm and the algorithm proposed here. We also discuss the conditions that allow both algorithms to be integrate into a single hybrid.

The plan of the paper is as follows: Section 2 gives a short introduction to quantum computers. Section 3 introduces the search problem and Grover's algorithm performance. Sections 4 to 6 introduce the proposed algorithm with analysis on its performance and behaviour. And we will end up with a conclusion in section 7.

2 Quantum Computers

2.1 Quantum Bits

In classical computers, a bit is considered as the basic unit for information processing; a bit can carry one value at a time (either 0 or 1). In quantum computers, the analogue of the bit is the quantum bit (*qubit* [15]), which has two possible states encoded as $|0\rangle$ and $|1\rangle$; where the notation $|\ \rangle$ is called *Dirac Notation* and is considered as the standard notation of states in quantum mechanics [7]. For quantum computing purposes, the states $|0\rangle$ and $|1\rangle$ can be considered as the classical bit values 0 and 1 respectively. An important difference between a classical bit and a qubit is that the qubit can exist in a linear superposition of both states ($|0\rangle$ and $|1\rangle$) at the same time and this gives the hope that quantum computers can do computation simultaneously (*Quantum Parallelism*). If we consider a quantum register with n qubits all in superposition, then any operation applied on this register will be applied on the 2^n states representing the superposition simultaneously.

2.2 Quantum Measurements

To read information from a quantum register (quantum system), we must apply a measurement on that register which will result in a projection of the states of the system to a subspace of the state space compatible with the values being measured. For example, consider a two-qubit system $\left|\phi\right\rangle$ defined as follows:

$$\left|\phi\right\rangle = \alpha\left|00\right\rangle + \beta\left|01\right\rangle + \gamma\left|10\right\rangle + \delta\left|11\right\rangle, \tag{1}$$

where α, β, γ, and δ are complex numbers called the amplitudes of the system and satisfy $\left|\alpha\right|^2 + \left|\beta\right|^2 + \left|\gamma\right|^2 + \left|\delta\right|^2 = 1$. The probability that the first qubit of $\left|\phi\right\rangle$ to be $\left|0\right\rangle$ is equal to $\left(\left|\alpha\right|^2 + \left|\beta\right|^2\right)$. If for some reasons we need to have the value $\left|0\right\rangle$ in the first qubit after any measurement, we must try some how to increase its probability before applying the measurement. Note that, the new state after applying measurement must be re-normalized so the total probability is still 1.

2.3 Quantum Gates

In general, quantum algorithms can be understood as follows: Apply a series of transformations (gates) then apply the measurement to get the desired result with high probability. According to the laws of quantum mechanics and to keep the reversibility condition required in quantum computation, the evolution of the state of the quantum system $\left|\psi\right\rangle$ of size n by time t is described by a matrix U of dimension $2^n \times 2^n$ [13]:

$$\left|\psi'\right\rangle = U\left|\psi\right\rangle, \tag{2}$$

where U satisfies the unitary condition: $U^{\dagger}U = I$, where $U^{\dagger}$ denotes the complex conjugate transpose of U and I is the identity matrix. For example, the X gate (NOT gate) is a single qubit gate (single input/output) similar in its effect to the classical NOT gate. It inverts the state $\left|0\right\rangle$ to the state $\left|1\right\rangle$ and visa versa. It's 2×2 unitary matrix takes this form,

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{3}$$

and its circuit takes the form shown in Fig.(1). Notice that, from now on we assume that a horizontal line used in a quantum circuit represents a qubit and the flow of the circuit logic is from left to right. For circuits with multiple qubits, qubits will be arranged according to the notation used in the figure.

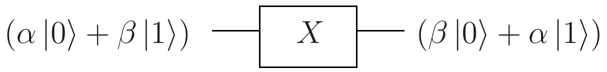

$(\alpha\left|0\right\rangle + \beta\left|1\right\rangle)$
X
$(\beta\left|0\right\rangle + \alpha\left|1\right\rangle)$

Figure 1: *NOT* gate quantum circuit.

Another important example is the Hadamard gate (H gate) which has no classical equivalent; it produces a completely random output with equal probabilities of the output to be $|0\rangle$ or $|1\rangle$ on any measurements. It's 2×2 unitary matrix takes this form,

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \tag{4}$$

and its circuit takes the form shown in Fig.(2).

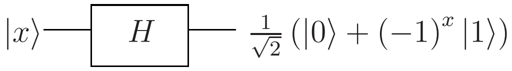
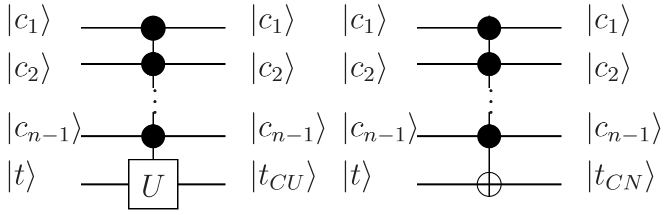

$\left|x\right\rangle$
H
$\frac{1}{\sqrt{2}}(\left|0\right\rangle + (-1)^x\left|1\right\rangle)$

Figure 2: Hadamard gate quantum circuit, where x is any Boolean variable.

Controlled operations are considered as the heart of quantum computing [2], the Controlled-U gate is the general case for any controlled gate with one or more control qubit(s) as shown in Fig.(3.a). It works as follows: If any of the control qubits $|c_i\rangle$'s ($1 \leq i \leq n-1$) is set to 0, then the quantum gate U will not be applied on target qubit $|t\rangle$; i.e. U is applied on $|t\rangle$ if and only if all $|c_i\rangle$'s are set to 1. The states of the qubits after applying the gate will be transformed according to the following rule:

$$\begin{array}{l} \left|c_i\right\rangle \to \left|c_i\right\rangle ; 1 \leq i \leq n-1 \\ \left|t\right\rangle \to \left|t_{CU}\right\rangle = U^{c_1 c_2 \ldots c_{n-1}}\left|t\right\rangle \end{array} \tag{5}$$

where $c_1c_2...c_{n-1}$ in the exponent of U means the AND-ing of the qubits $c_1,\ c_2, ..., c_{n-1}$.



a.Controlled-U b.Controlled-NOT

Figure 3: Controlled gates where the back circle $\bullet$ indicates the control qubits, and the symbol $\oplus$ in part (b.) indicates the target qubit.

If U in the general Controlled-U gate is replaced with the X gate mentioned above, the resulting gate is called a Controlled-NOT gate (shown in Fig.(3.b)). It works as follows: It inverts the target qubit if and only if all the control qubits are set to 1. Thus the qubits of the system $c_1, c_2, ..., c_{n-1}, t$ will be transformed according to the following rule:

$$\begin{array}{l} |c_i\rangle \rightarrow |c_i\rangle \,; 1 \leq i \leq n-1 \\ |t\rangle \rightarrow |t_{CN}\rangle = |t \oplus c_1c_2...c_{n-1}\rangle \end{array} \tag{6}$$

where $c_1c_2 \ldots c_{n-1}$ is the AND-ing of the qubits $c_1, c_2, \ldots, c_{n-1}$ and $\oplus$ is the classical XOR operation.

3 Search Problem

Consider a list L of N items; $L = \{0, 1, ..., N-1\}$, and consider a function f which maps the items in L to either 0 or 1 according to some properties these items shall satisfy; i.e. $f : L \rightarrow \{0, 1\}$. The problem is to find any $i \in L$ such that $f(i) = 1$ assuming that such i must exist in the list. It was shown classically that we need approximately $N/2$ tests to get a result with probability at least one-half. Let M denotes the number of matches within the search space such that $1 \leq M \leq N$ and for simplicity and without loss of generality we can assume that $N = 2^n$. Grover's algorithm was shown to solve this problem [4] in $O\left(\sqrt{N/M}\right)$. In [13], it was shown that the number of iterations will increase for $M > N/2$ which is undesired behaviour for a search algorithm. To overcome this problem it was proposed in [13] that the search space can be doubled so the number of matches is always less than half the search space and then iterate the algorithm $\pi/4\sqrt{2N/M}$ times so

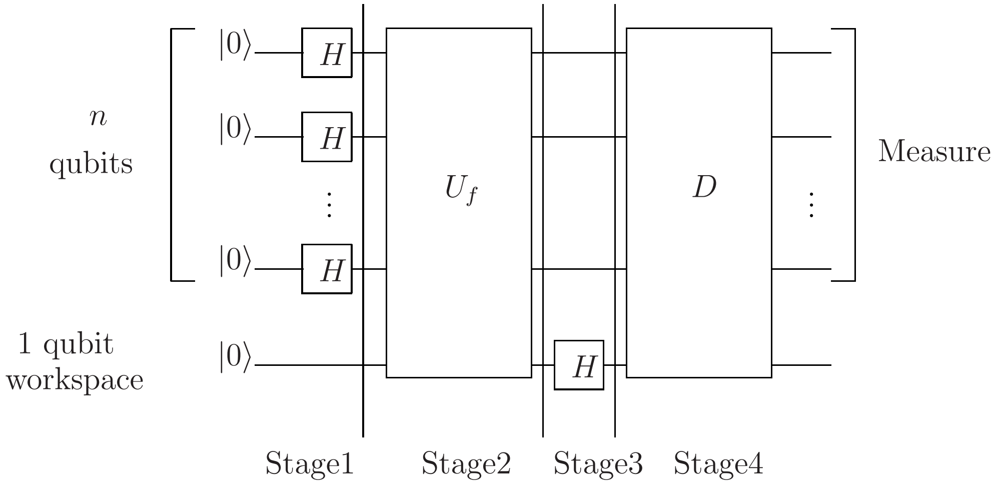


Figure 4: Quantum circuit for the proposed algorithm.

the algorithm still runs in $O\left(\sqrt{N/M}\right)$. But using this approach will double the cost of space/time requirement. In the following section we will present an algorithm that can find a solution for $M > N/2$ with probability at least 92.6% after applying the algorithm once.

4 The Algorithm

4.1 Iterating the algorithm once

For a list of size $N = 2^n$, the steps of the algorithm can be understood as follows as shown in Fig.(4):

1- *Register Preparation.* Prepare a quantum register of $n+1$ qubits all in state $|0\rangle$, where the extra qubit is used as a workspace for evaluating the oracle U_f:

$$|W_0\rangle = |0\rangle^{\otimes n} \otimes |0\rangle \, . \tag{7}$$

2- *Register Initialization.* Apply Hadamard gate on each of the first n qubits in parallel, so they contain the 2^n states, where i is the integer representation of items in the list:

$$\left|W_1\right\rangle = \left(H^{\otimes n} \otimes I\right)\left|W_0\right\rangle = \left(\frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}\left|i\right\rangle\right)\otimes\left|0\right\rangle;\ N = 2^n. \tag{8}$$

3- *Applying Oracle.* Apply the oracle U_f to map the items in the list to either 0 or 1 simultaneously and store the result in the extra workspace qubit:

$$\left|W_2\right\rangle = U_f\left|W_1\right\rangle = \frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}\left(\left|i\right\rangle \otimes \left|0 \oplus f(i)\right\rangle\right) = \frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}\left(\left|i\right\rangle \otimes \left|f(i)\right\rangle\right). \tag{9}$$

4- *Completing Superposition and Changing Sign.* Apply Hadamard gate on the workspace qubit. This will extend the superposition for the $n+1$ qubits with the amplitudes of the desired states with negative sign as follows:

$$\begin{aligned}\left|W_3\right\rangle &= \left(I^{\otimes n}\otimes H\right)\left|W_2\right\rangle = \tfrac{1}{\sqrt{N}}\sum_{i=0}^{N-1}\left(\left|i\right\rangle \otimes \left(\tfrac{\left|0\right\rangle + (-1)^{f(i)}\left|1\right\rangle}{\sqrt{2}}\right)\right)\\ &= \tfrac{1}{\sqrt{P}}\sum_{i=0}^{N-1}\left(\left|i\right\rangle \otimes \left(\left|0\right\rangle + (-1)^{f(i)}\left|1\right\rangle\right)\right);\ \ P = 2N = 2^{n+1}.\end{aligned} \tag{10}$$

Let M be the number of matches, which makes the oracle U_f evaluate to 1 (solutions); such that $1 \leq M \leq N$; assume that $\sum_i{}'$ indicates a sum over all i which are desired matches ($2M$ states), and $\sum_i{}''$ indicates a sum over all i which are undesired items in the list. So, $\left|W_3\right\rangle$ can be re-written as follows:

$$\begin{aligned}\left|W_3\right\rangle = &\tfrac{1}{\sqrt{P}}\sum_{i=0}^{N-1}{}'\left(\left|i\right\rangle \otimes \left(\left|0\right\rangle - \left|1\right\rangle\right)\right)\\ &+\tfrac{1}{\sqrt{P}}\sum_{i=0}^{N-1}{}''\left(\left|i\right\rangle \otimes \left(\left|0\right\rangle + \left|1\right\rangle\right)\right)\\ &= \tfrac{1}{\sqrt{P}}\sum_{i=0}^{N-1}{}'\left(\left|i\right\rangle \otimes \left|0\right\rangle\right) - \tfrac{1}{\sqrt{P}}\sum_{i=0}^{N-1}{}'\left(\left|i\right\rangle \otimes \left|1\right\rangle\right)\\ &+\tfrac{1}{\sqrt{P}}\sum_{i=0}^{N-1}{}''\left(\left|i\right\rangle \otimes \left|0\right\rangle\right) + \tfrac{1}{\sqrt{P}}\sum_{i=0}^{N-1}{}''\left(\left|i\right\rangle \otimes \left|1\right\rangle\right).\end{aligned} \tag{11}$$

From Eqn.(11); we can see that there are M states with amplitude $({}^{-1}/_{\sqrt{P}})$ where $f(i) = 1$, and $(P - M)$ states with amplitude $({}^{1}/_{\sqrt{P}})$. Notice that, applying Hadamard gate on the extra qubit splits the $|i\rangle$ states (solution states), to M states $\left({\sum_i}' \left(|i\rangle \otimes |0\rangle\right)\right)$ with positive amplitude $({}^{1}/_{\sqrt{P}})$ and M states $\left({\sum_i}' \left(|i\rangle \otimes |1\rangle\right)\right)$ with negative amplitude $({}^{-1}/_{\sqrt{P}})$.

5- *Inversion About the Mean.* Apply the *Diffusion Operator* D similar to that used in Grover's algorithm [10] on the $n+1$ qubits. The diagonal representation of the diffusion operator D can take this form:

$$D = H^{\otimes n+1} \left(2\,|0\rangle\,\langle 0| - I\right) H^{\otimes n+1} = 2\,|\psi\rangle\,\langle\psi| - I. \tag{12}$$

where, $|\psi\rangle = \frac{1}{\sqrt{P}} \sum_{k=0}^{P-1} |k\rangle$ is an equally weighted superposition of states. The effect of applying D [13] on a general state $\sum_{k=0}^{P-1} \alpha_k\,|k\rangle$ produces $\sum_{k=0}^{P-1} \left[-\alpha_k + 2\,\langle\alpha\rangle\right] |k\rangle$, where, $\langle\alpha\rangle = \frac{1}{P} \sum_{k=0}^{P-1} \alpha_k$ is the mean of the amplitudes of all states in the superposition; i.e. the amplitudes α_k will be transformed according to the following relation:

$$\alpha_k \rightarrow \left[-\alpha_k + 2\,\langle\alpha\rangle\right]. \tag{13}$$

In our case, there are M states with amplitude $({}^{-1}/_{\sqrt{P}})$ and $P - M$ states with amplitude $({}^{1}/_{\sqrt{P}})$, so the mean $\langle\alpha\rangle$ is as follows:

$$\langle\alpha\rangle = \frac{1}{P}\left(M\left(\frac{-1}{\sqrt{P}}\right) + (P - M)\left(\frac{1}{\sqrt{P}}\right)\right). \tag{14}$$

So, applying D on the system $|W_3\rangle$ shown in Eqn.(11) can be understood as follows:

a- The M negative sign amplitudes (solutions): will be transformed from $({}^{-1}/_{\sqrt{P}})$ to a , where a is calculated as follows: Substitute $\alpha_k = \frac{-1}{\sqrt{P}}$ and $\langle\alpha\rangle$ shown (Eqn.(14)) in Eqn.(13) we get:

$$\begin{aligned} a &= -\left(\tfrac{-1}{\sqrt{P}}\right) + \tfrac{2}{P}\left(M\left(\tfrac{-1}{\sqrt{P}}\right) + (P - M)\left(\tfrac{1}{\sqrt{P}}\right)\right) \\ &= \tfrac{1}{\sqrt{P}}\left(3 - \tfrac{4M}{P}\right). \end{aligned} \tag{15}$$

b- The $(P-M)$ positive sign amplitudes will be transformed from $({}^{1}/\sqrt{P})$ to b , where b is calculated as follows: Substitute $\alpha_k = \frac{1}{\sqrt{P}}$ and $\langle\alpha\rangle$ shown (Eqn.(14)) in Eqn. (13) we get:

$$\begin{aligned} b &= -\left(\tfrac{1}{\sqrt{P}}\right) + \tfrac{2}{P}\left(M\left(\tfrac{-1}{\sqrt{P}}\right) + (P-M)\left(\tfrac{1}{\sqrt{P}}\right)\right) \\ &= \tfrac{1}{\sqrt{P}}\left(1 - \tfrac{4M}{P}\right). \end{aligned} \tag{16}$$

We can see that $a > b$ after applying D. The new system $|W_4\rangle$ can be written as follows:

$$\begin{aligned} D\,|W_3\rangle = |W_4\rangle = b\sum_{i=0}^{N-1}{}'\,(|i\rangle \otimes |0\rangle) + a\sum_{i=0}^{N-1}{}'\,(|i\rangle \otimes |1\rangle) \\ +b\sum_{i=0}^{N-1}{}''\,(|i\rangle \otimes |0\rangle) + b\sum_{i=0}^{N-1}{}''\,(|i\rangle \otimes |1\rangle). \end{aligned} \tag{17}$$

such that,

$$Ma^2 + (P-M)b^2 = 1. \tag{18}$$

Notice that, if no matches exist within the superposition (i.e. $M = 0$), then all the amplitudes will have positive sign and then applying the diffusion operator D will not change the amplitudes of the states as follows: Substituting $\alpha_k = \frac{1}{\sqrt{P}}$ and $\langle\alpha\rangle = \frac{1}{P}\left(P\left(\frac{1}{\sqrt{P}}\right)\right)$ in Eqn.(13) we get:

$$\frac{1}{\sqrt{P}} + \frac{2}{P}\left(P\left(\frac{1}{\sqrt{P}}\right)\right) = \frac{1}{\sqrt{P}} = \alpha_k, \tag{19}$$

6- *Measurement.* Measure the first n qubits, we get the desired solution with probability given below:

i- Probability P_s to find a match out of the M possible matches; taking into account that a solution $|i\rangle$ occurs *twice* as: $(|i\rangle \otimes |0\rangle)$ with amplitude b and $(|i\rangle \otimes |1\rangle)$ with amplitude a as shown in Eqn.(17), can be calculated as follows:

$$
\begin{aligned}
P_s &= M(a^2+b^2) \\
&= \tfrac{M}{2N}\left(10-16\left(\tfrac{M}{N}\right)+8\left(\tfrac{M}{N}\right)^2\right) \\
&= 5\left(\tfrac{M}{N}\right)-8\left(\tfrac{M}{N}\right)^2+4\left(\tfrac{M}{N}\right)^3 .
\end{aligned}
\tag{20}
$$

ii- Probability P_{ns} to find undesired result out of the states can be calculated as follows:

$$
P_{ns} = (P-2M)b^2. \tag{21}
$$

Notice that, using Eqn.(18)

$$
\begin{aligned}
P_s + P_{ns} &= M(a^2+b^2)+(P-2M)b^2 \\
&= Ma^2+(P-M)b^2 = 1.
\end{aligned}
\tag{22}
$$

4.1.1 Performance after Iterating the Algorithm Once

n, where $N=2^n$	Max. prob.	Min. prob.	Avg. prob.
2	1.0	0.8125	0.875
3	1.0	0.507812	0.937500
4	1.0	0.282227	0.968750
5	1.0	0.148560	0.984375
6	1.0	0.076187	0.992187

Table 1: Algorithm performance with different size search space.

Considering Eqn.(15), Eqn.(16), Eqn.(20) and Eqn.(21), we can see that the probability to find a solution varies according to the number of matches M in the superposition.

From Table.1, we can see that the maximum probability is always 1.0, and the minimum probability (worst case) decreases as the size of the list increases, which is expected for small M because the number of states will increase and the probability shall distribute over more states while the average probability increases as the size of the list increases. It implies that the average performance of the algorithm to find a solution increases as the size of the list increases.

To verify these results, taking into account that the oracle U_f is taken as a black box, we can define the average probability of success of the algorithm; $average(P_s)$, as follows:

$$\begin{aligned} average(P_s) &= \tfrac{1}{2^N} \sum\limits_{M=1}^{N} {}^N C_M P_s \\ &= \tfrac{1}{2^N} \sum\limits_{M=1}^{N} \tfrac{N!}{M!(N-M)!}.M \left(a^2+b^2\right) \\ &= \tfrac{1}{2^{N+1}N^3} \sum\limits_{M=1}^{N} \tfrac{N!}{(M-1)!(N-M)!} \left(10N^2 - 16MN + 8M^2\right) \\ &= 1 - \tfrac{1}{2N}. \end{aligned} \tag{23}$$

where ${}^N C_M = \frac{N!}{M!(N-M)!}$ is the number of possible cases for M matches. We can see that as the size of the list increases ($N \to \infty$), $average(P_s)$ shown in Eqn.(23) tends to 1.

Classically, we can try to find a random guess of the item, which represents the solution (one trial guess), we may succeed to find a solution with probability $P_s^{(classical)} = M/N$. The average probability can be calculated as follows:

$$\begin{aligned} average(P_s^{(classical)}) &= \tfrac{1}{2^N} \sum\limits_{M=1}^{N} {}^N C_M P_s^{(classical)} \\ &= \tfrac{1}{2^N} \sum\limits_{M=1}^{N} \tfrac{N.M}{M!(N-M)!N} \\ &= \tfrac{1}{2}. \end{aligned} \tag{24}$$

It means that we have an average probability one-half to find or not to find a solution by a single random guess even with the increase in the number of matches.

Similarly, Grover's algorithm has an average probabilty *one-half* after arbitrary number of iterations as we will see. It was shown in [4] that the probability of success of Grover's algorithm after q iterations is given by:

$$P_s^{G(q)} = \sin^2((2q+1)\theta), \text{ where, } 0 < \theta < \frac{\pi}{2} \text{ and } \sin^2(\theta) = \frac{M}{N}. \tag{25}$$

The average probability of success of Grover's algorithm after arbitrary number of iterations can be calculated as follows (Appendix A):

$$average(P_s^{G^{(q)}}) = \frac{1}{2^N} \sum_{M=1}^{N} {}^N C_M \sin^2((2t+1)\theta) = \frac{1}{2}. \tag{26}$$

Comparing the performance of the proposed algorithm, first iteration of Grover's algorithm and the classical guess technique, Fig.(5) shows the probability of success of the three algorithms just mentioned as a function of the ratio (M/N).

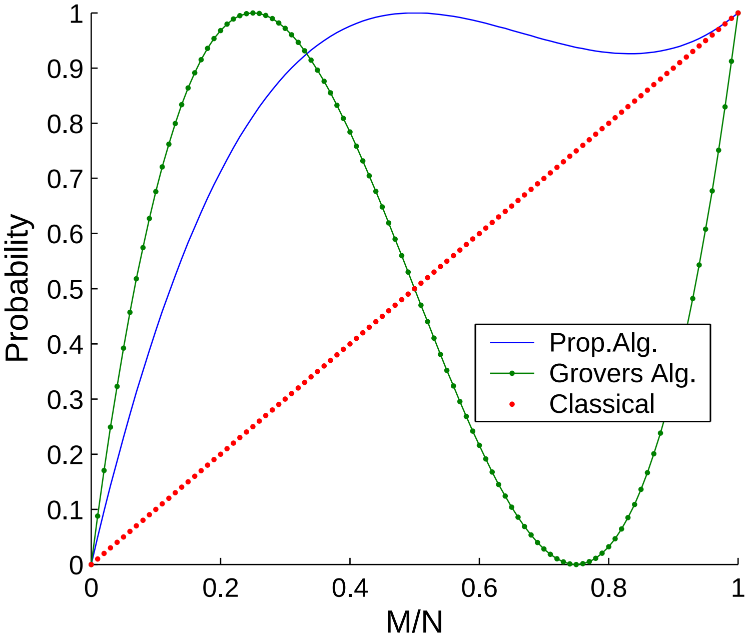


Figure 5: A plot of the probability of success of the proposed algorithm P_s, first iteration of Grover's algorithm $P_s^{G^{(1)}}$ and the classical guess $P_s^{(classical)}$ as a function of the ratio (M/N).

We can see from Fig.(5) that the probability of success of the proposed quantum algorithm is always above that of the classical guess technique. Grover's algorithm solves the case where $M = N/4$ with certainty and the proposed algorithm solves the case where $M = N/2$ with certainty. The probability of success of Grover's algorithm will start to go below one-half for $M > N/2$ while the probability of success of the proposed algorithm will stay more reliable with propabilty at least 92.6%. For $M < N/8$, the

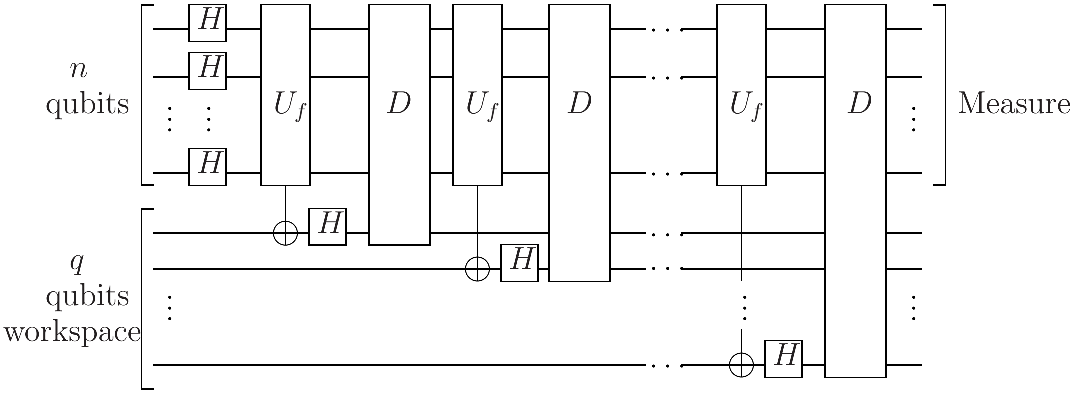


Figure 6: Quantum circuit for the iterative version of the proposed algorithm.

probability of success of the proposed algorithm will start to go below one-half where performance of Grover's algorithm will be much better as we will verify in the next section.

4.2 Iterating the algorithm

If we consider iterating the above algorithm: For a list of size $N(= 2^n)$, prepare n qubits and append extra q qubits for applying q iterations of the algorithm. The iterating version of the algorithm works as follows (as shown in Fig.(6)):

1- Initialize the whole $n + q$ qubits system to the state $|0\rangle$.

2- Apply Hadamard gate on each of the first n qubits in parallel.

3- Iterate the following, for iteration k:

a. Apply the oracle U_f taking the first n qubits as control qubits and the k^{th} qubit workspace as the target qubit exclusively.

b. Apply Hadamard gate on the k^{th} qubit workspace.

c. Apply diffusion operator on the whole $n + k$ qubit system inclusively.

4- Apply measurement on the first n qubits.

To understand how the iterative version of the algorithm affects the system, we will trace the state of the system during the first few iterations.

Consider the system after the first iteration shown in Eqn.(17), second iteration will modify the system as follows (to clear ambiguity, a and b used in the above section will be denoted as $a_0^{(1)}$ and $b_0^{(1)}$ respectively, where the *superscript index* denotes the iteration and the *subscript index* is used to distinguish amplitudes):

1- Append second qubit workspace to the system:

$$\begin{aligned}\left|W_1^{(2)}\right\rangle = b_0^{(1)} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |0\rangle\right) \otimes |0\rangle + a_0^{(1)} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |1\rangle\right) \otimes |0\rangle \\ + b_0^{(1)} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |0\rangle\right) \otimes |0\rangle + b_0^{(1)} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |1\rangle\right) \otimes |0\rangle.\end{aligned} \tag{27}$$

2- Apply U_f as shown in step 3-a:

$$\begin{aligned}\left|W_2^{(2)}\right\rangle = b_0^{(1)} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |0\rangle\right) \otimes |1\rangle + a_0^{(1)} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |1\rangle\right) \otimes |1\rangle \\ + b_0^{(1)} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |0\rangle\right) \otimes |0\rangle + b_0^{(1)} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |1\rangle\right) \otimes |0\rangle.\end{aligned} \tag{28}$$

3- Apply Hadamard gate on second qubit workspace ($I^{\otimes n+1} \otimes H$):

$$\begin{aligned}\left|W_3^{(2)}\right\rangle = \tfrac{b_0^{(1)}}{\sqrt{2}} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |0\rangle\right) \otimes |0\rangle - \tfrac{b_0^{(1)}}{\sqrt{2}} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |0\rangle\right) \otimes |1\rangle \\ + \tfrac{a_0^{(1)}}{\sqrt{2}} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |1\rangle\right) \otimes |0\rangle - \tfrac{a_0^{(1)}}{\sqrt{2}} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |1\rangle\right) \otimes |1\rangle \\ + \tfrac{b_0^{(1)}}{\sqrt{2}} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |0\rangle\right) \otimes |0\rangle + \tfrac{b_0^{(1)}}{\sqrt{2}} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |0\rangle\right) \otimes |1\rangle \\ + \tfrac{b_0^{(1)}}{\sqrt{2}} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |1\rangle\right) \otimes |0\rangle + \tfrac{b_0^{(1)}}{\sqrt{2}} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |1\rangle\right) \otimes |1\rangle.\end{aligned} \tag{29}$$

4- Apply diffusion operator as shown in step 3-c:

$$\begin{aligned}\left|W_4^{(2)}\right\rangle = {} & b_0^{(2)} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |0\rangle\right) \otimes |0\rangle + b_1^{(2)} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |0\rangle\right) \otimes |1\rangle \\ & + a_0^{(2)} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |1\rangle\right) \otimes |0\rangle + a_1^{(2)} \sum_{i=0}^{N-1}{}' \left(|i\rangle \otimes |1\rangle\right) \otimes |1\rangle \\ & + b_0^{(2)} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |0\rangle\right) \otimes |0\rangle + b_0^{(2)} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |0\rangle\right) \otimes |1\rangle \\ & + b_0^{(2)} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |1\rangle\right) \otimes |0\rangle + b_0^{(2)} \sum_{i=0}^{N-1}{}'' \left(|i\rangle \otimes |1\rangle\right) \otimes |1\rangle. \end{aligned} \tag{30}$$

where the mean of the amplitudes to be used in the diffusion operator is calculated as follows:

$$\begin{aligned}\langle \alpha_2 \rangle &= \tfrac{1}{2^{n+2}} \left(\left(2^{n+2} - 4M\right) \tfrac{b_0^{(1)}}{\sqrt{2}} \right) \\ &= \tfrac{b_0^{(1)}}{\sqrt{2}} \left(1 - \tfrac{M}{N}\right). \end{aligned} \tag{31}$$

And the new amplitudes $a_0^{(2)}$, $a_1^{(2)}$,$b_0^{(2)}$ and $b_1^{(2)}$ are calculated as follows:

$$\begin{aligned} a_0^{(2)} &= 2\langle \alpha_2 \rangle - \tfrac{a_0^{(1)}}{\sqrt{2}}; \quad a_1^{(2)} = 2\langle \alpha_2 \rangle + \tfrac{a_0^{(1)}}{\sqrt{2}}. \\ b_0^{(2)} &= 2\langle \alpha_2 \rangle - \tfrac{b_0^{(1)}}{\sqrt{2}}; \quad b_1^{(2)} = 2\langle \alpha_2 \rangle + \tfrac{b_0^{(1)}}{\sqrt{2}}. \end{aligned} \tag{32}$$

And the probability of success:

$$P_s^{(2)} = M\left(\left(a_0^{(2)}\right)^2 + \left(a_1^{(2)}\right)^2 + \left(b_0^{(2)}\right)^2 + \left(b_1^{(2)}\right)^2 \right). \tag{33}$$

For the sake of simplicity, we can trace the effect of each iteration on the amplitudes of the system instead of writing the state of the system explicitly; for example, the amplitudes of the system after third iteration will be as follows:

1- The mean of the amplitudes to be used in the diffusion operator:

$$\begin{aligned}\langle\alpha_3\rangle &= \tfrac{1}{2^{n+3}}\left(\left(2^{n+3}-8M\right)\tfrac{b_0^{(2)}}{\sqrt{2}}\right)\\ &= \tfrac{b_0^{(2)}}{\sqrt{2}}\left(1-\tfrac{M}{N}\right).\end{aligned} \tag{34}$$

2- The new amplitudes:

$$\begin{array}{ll} a_0^{(3)} = 2\left\langle\alpha_3\right\rangle - \frac{a_0^{(2)}}{\sqrt{2}}; & a_1^{(3)} = 2\left\langle\alpha_3\right\rangle + \frac{a_0^{(2)}}{\sqrt{2}}.\\ a_2^{(3)} = 2\left\langle\alpha_3\right\rangle - \frac{a_1^{(2)}}{\sqrt{2}}; & a_3^{(3)} = 2\left\langle\alpha_3\right\rangle + \frac{a_1^{(2)}}{\sqrt{2}}.\\ b_0^{(3)} = 2\left\langle\alpha_3\right\rangle - \frac{b_0^{(2)}}{\sqrt{2}}; & b_1^{(3)} = 2\left\langle\alpha_3\right\rangle + \frac{b_0^{(2)}}{\sqrt{2}}.\\ b_2^{(3)} = 2\left\langle\alpha_3\right\rangle - \frac{b_1^{(2)}}{\sqrt{2}}; & b_3^{(3)} = 2\left\langle\alpha_3\right\rangle + \frac{b_1^{(2)}}{\sqrt{2}}.\end{array} \tag{35}$$

3- And the probability of success:

$$P_s^{(3)} = M\left(\left(a_i^{(3)}\right)^2 + \left(b_i^{(3)}\right)^2\right); i = 0,1,2,3. \tag{36}$$

In general, after q iterations the recurrence relations representing the iteration can be written as follows:
The initial conditions: $a_0^{(0)} = b_0^{(0)} = \frac{1}{\sqrt{N}}$.

1- The mean to be used in the diffusion operator:

$$\left\langle\alpha_q\right\rangle = \frac{b_0^{(q-1)}}{\sqrt{2}}\left(1-\frac{M}{N}\right); q \geq 1. \tag{37}$$

2- The new amplitudes of the system:

$$a_0^{(1)} = 2\left\langle\alpha_1\right\rangle + \frac{a_0^{(0)}}{\sqrt{2}}, \quad a_{0\to 2^{q-1}-1}^{(q)} = 2\left\langle\alpha_q\right\rangle \mp \frac{a_{0\to 2^{q-2}-1}^{(q-1)}}{\sqrt{2}}; \quad q \geq 2. \tag{38}$$

$$b_0^{(1)} = 2\left\langle\alpha_1\right\rangle - \frac{b_0^{(0)}}{\sqrt{2}}, \quad b_{0\to 2^{q-1}-1}^{(q)} = 2\left\langle\alpha_q\right\rangle \mp \frac{b_{0\to 2^{q-2}-1}^{(q-1)}}{\sqrt{2}}; \quad q \geq 2. \tag{39}$$

3- The probability of success for $q \geq 1$:

$$P_s^{(q)} = M\left(\left(a_i^{(q)}\right)^2 + \left(b_i^{(q)}\right)^2\right); i = 0, 1, 2, ..., 2^{q-1} - 1. \quad (40)$$

Using mathematical induction, we can prove that the probability of success after q iterations shown in Eqn.(40) can take this form (Appendix B):

$$P_s^{(q)} = \left(\frac{M}{N} - 1\right)\left(1 - \frac{2M}{N}\right)^{2q} + 1, \quad q \geq 1. \quad (41)$$

4.2.1 Performance of Iterating the Algorithm

i- The case where multiple instances of a match exist within the search space: Consider the following cases using Eqn.(41):

1- The case where $M = N/2$: the algorithm can find a solution with *certainty* after arbitrary number of iterations (one iteration is enough).

2- The case where $M > N/2$: the probability of success is; for instances, at least 92.6% after the first iteration, 95.9% after second iteration and 97.2% after third iteration.

3- For iterating the algorithm once ($q = 1$) and to get probability at least one-half, so, M must satisfy the condition $M \geq N/8$.

ii- The case where few instances of a match exist within the search space:

First, we need to represent the number of iterations q in terms of the ratio M/N. From Eqn.(41) and using Taylor's expansion we get:

$$q \geq \frac{P_s^{(q)} - \frac{M}{N}}{4\frac{M}{N}\left(1 - \frac{M}{N}\right)}. \quad (42)$$

For the cases where $q > 1$, the following conditions must be satisfied:

$$n \geq 4 \text{ and } 1 \leq M < N/8. \quad (43)$$

It means that first iteration will cover approximately 87.5% of the problem with probability at least one-half; two iterations will cover approximately 92% and three iterations will cover 94%. It is easy to prove that the rate of the increase of the coverage range will decrease as number of iterations increases as shown in Fig.(7). We can also see from Eqn.(42) and Eqn.(43) that the algorithm needs $O\left(N/M\right)$ iterations for $n \geq 4$ and $1 \leq M < N/8$, which is similar to classical algorithms behaviour. It leads to a conclusion that first few iterations of the algorithm will do the best performance and there will be no big gain from continuing to iterate the algorithm.

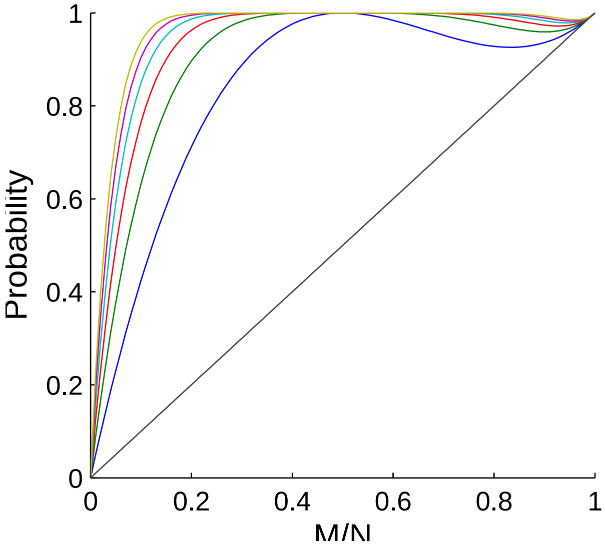


Figure 7: A plot of the probability of success of the iterative version of the proposed algorithm where q = 1,2,...,6.

5 A Hybrid Quantum Search Engine

We have devised a quantum search algorithm, which performs very well in case of multiple instances of the solution within the search space and a classical behaviour in case of few instances of the solution. On contrary, Grover's algorithm needs $O\left(\sqrt{N/M}\right)$ to solve the problem but it's performance decreases for $M > N/2$ [13].

This leads up to propose a *hybrid quantum search engine,* which combines both algorithms and can be integrated as follows:

i- If the number of solutions M is *known in advance*:

1- If $1 \leq M < N/8$: Use Grover's algorithm with $O\left(\sqrt{N/M}\right)$.

2- If $N/8 \leq M < N$: Use the proposed algorithm with $O(1)$.

ii- If the number of solutions M is *unknown*:

Iterate the proposed algorithm few times; say three iterations, which results in a chance of approximately 94% to find a solution. If it fails, we apply Grover's algorithm so we still have the same complexity $O\left(\sqrt{N/M}\right)$.

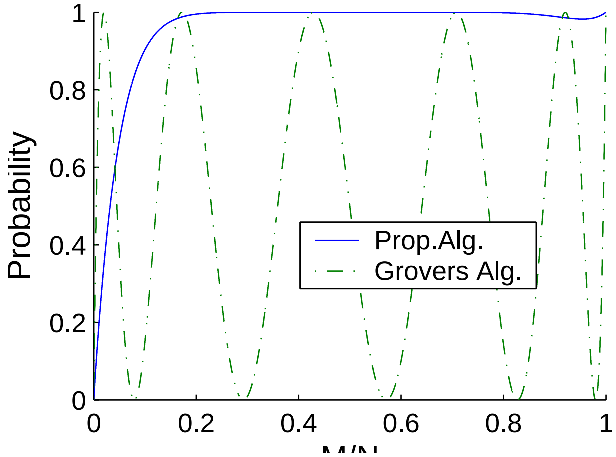


Figure 8: The probability of success after five iterations from Grover's algorithm's vs. the proposed algorithm.

We can see from Fig.(8) that Grover's algorithm is much faster in the case of few instances of the solution (ratio M/N is small) and the proposed algorithm is more stable and reliable in case of multiple instances of the solution.

6 Conclusion

In this paper, we proposed a quantum search algorithm, which performs very fast in the case of multiple instances of the solution within the search space (almost constant run-time) but the performance turns out to be classical for few instances of the solution.

On the other hand we have Grover's algorithm, which performs very well in case of few instances of the solution and the performance decrease as number of solutions increase within the search space.

This gave us the chance to propose a *hybrid quantum search engine* with *general performance* better that any pure classical or quantum search algorithm and still has $O\left(\sqrt{N}\right)$ for the hardest case and approximately $O(1)$ for $M \geq N/8$.

References

[1] Accardi, L., Sabbadini, R. (2000),*A Generalization of Grover's Algorithm.* Los Alamos Physics Preprint Archive, quant-ph/0012143.

[2] Barenco, A., Bennett, C., Cleve, R., Divincenzo, D. P., Margolus, N., Shor, P., Sleator, T., Smolin, J., and Weinfurter, H. (1995), *Elementary Gates for Quantum Computation.* Physical Review A, 52(5), pp. 3457-3467.

[3] Bernstein, E. and Vazirani, U. (1993), *Quantum Complexity Theory.* In Proceedings of the 25^{th} Annual ACM Symposium on Theory of Computing, pp. 11-20.

[4] Boyer, M., Brassard, G., Hoyer, P. and Tapp, A. (1996), *Tight Bounds on Quantum Searching.* In Proceedings of the 4^{th} Workshop on Physics and Computation, pp. 36-43.

[5] Brassard, G., Hyer, P., Mosca, M., and Tapp, A. (2002), *Quantum Amplitude Amplification and Estimation.* In Quantum Computation and Quantum Information: A Millennium Volume, AMS Contemporary Mathematics Series, Volume 305.

[6] Deutsch, D. (1985), *Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer*. In Proceedings of the Royal Society of London A, 400, pp. 97-117.

[7] Dirac, P. (1947), *The Principles of Quantum Mechanics*. Clarendon Press, Oxford, United Kingdom.

[8] Feynman, R.P. (1986), *Quantum Mechanical Computers*. Foundations of Physics, 16, pp. 507-531.

[9] Galindo, A., Martin-Delgado, M. A. (2000), *A Family of Grover's Quantum Searching Algorithms*. Los Alamos Physics Preprint Archive, quant-ph/0009086.

[10] Grover, L. K. (1996), *A Fast Quantum Mechanical Algorithm for Database Search*. In Proceedings of the 28^{th} Annual ACM Symposium on the Theory of Computing (STOC), pp. 212-219.

[11] Jozsa, R. (1999),*Searching in Grover's Algorithm*. Los Alamos Physics Preprint Archive, quant-ph/9901021.

[12] Lloyd, S. (1993), *A Potentially Realizable Quantum Computer*. Science, 261, pp. 1569-1571.

[13] Nielsen, M. and Chuang, I. (2000), *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, United Kingdom,Chap.6

[14] Simon, D. R. (1994), *On the Power of Quantum Computation*. In Proceedings of the 35^{th} Annual Symposium on Foundations of Computer Science, pp. 116-123.

[15] Schumacher, B. (1995), *Quantum Coding*. Physical Review A, 51, pp. 2738-2747.

[16] Shor, P.W. (1997), *Polynomial-time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*. SIAM Journal on Computing, 26(5): pp.1484-1509.

Appendix A

To proof the identity shown in Eqn.(26), we first need the next Lemma.

Lemma 6.1 *Let α and β angles; $0 < \alpha, \beta < \pi/2$, such that :*

$$\sin^2(\alpha) = \cos^2(\beta). \tag{44}$$

Then, if k is any odd positive integer, we have:

$$\sin^2(k\alpha) = \cos^2(k\beta). \tag{45}$$

Proof *Since, $0 < \alpha, \beta < \pi/2$, then we have,*

$$\sin(\alpha) = \cos(\beta). \tag{46}$$

Also, we can write the following: $\cos(\beta) = \sin(\pi/2 - \beta) = \sin(\alpha)$.
So, $\alpha = \pi/2 - \beta$.
Therefore;

$$\begin{aligned} \cos(k\beta) &= \cos\left(k\left(\tfrac{\pi}{2} - \alpha\right)\right) \\ &= \cos\left(\tfrac{k\pi}{2} - k\alpha\right) \\ &= \cos\left(\tfrac{k\pi}{2}\right)\cos(k\alpha) + \sin\left(\tfrac{k\pi}{2}\right)\sin(k\alpha) \\ &= \pm\sin(k\beta). \end{aligned} \tag{47}$$

Then, $\cos^2(k\beta) = \sin^2(k\beta)$.

Theorem 6.2 *For any odd positive integer k and an angle θ_M; $0 < \theta_M < \frac{\pi}{2}$, defined as follows:*

$$\sin^2(\theta_M) = \frac{M}{N}. \tag{48}$$

Then,

$$\sum_{M=1}^{N} {}^{N}C_M \sin^2(k\theta_M) = 2^{n-1}. \tag{49}$$

Proof *Consider*

$$sin^2(\theta_M) + \sin^2(\theta_{N-M}) = \frac{M}{N} + \frac{N-M}{N} = 1. \tag{50}$$

So,

$$\begin{aligned} \sin^2(\theta_M) &= 1 - \sin^2(\theta_{N-M}) \\ &= \cos^2(\theta_{N-M}) . \end{aligned} \tag{51}$$

By Lemma,

$$\sin^2(k\theta_M) = \cos^2(k\theta_{N-M}) . \tag{52}$$

Or,

$$\sin^2(k\theta_M) + \sin^2(k\theta_{N-M}) = 1. \tag{53}$$

Now consider,

$$\begin{aligned} 2\sum_{M=1}^{N} {}^{N}C_M \sin^2(k\theta_M) &= \sum_{M=1}^{N} {}^{N}C_M \sin^2(k\theta_M) + \sum_{M=1}^{N} {}^{N}C_{N-M} \sin^2(k\theta_M) \\ &= \sum_{M=1}^{N} {}^{N}C_M \sin^2(k\theta_M) + \sum_{M=0}^{N} {}^{N}C_M \sin^2(k\theta_{N-M}) \\ &= \sum_{M=1}^{N} {}^{N}C_M \left(\sin^2(k\theta_M) + \sin^2(k\theta_{N-M})\right) \\ &= \sum_{M=1}^{N} {}^{N}C_M \\ &= 2^n. \end{aligned} \tag{54}$$

Appendix B

To prove that the probability of success after q iterations is as shown in Eqn.(41), we need first to prove the following relation:

Let $b_0^{(0)} = \frac{1}{\sqrt{2^n}} = \frac{1}{\sqrt{N}}$, and given by the definition of the diffusion operator for $q \geq 1$ that,

$$\langle \alpha_q \rangle = \frac{b_0^{(q-1)}}{\sqrt{2}} \left(1 - \frac{M}{N} \right) \tag{55}$$

And,

$$b_0^{(q)} = 2 \langle \alpha_q \rangle - \frac{b_0^{(q-1)}}{\sqrt{2}} \tag{56}$$

Then,

$$b_0^{(q)} = \frac{b_0^{(0)}}{\left(\sqrt{2}\right)^q} \left(1 - 2\frac{M}{N} \right)^q \tag{57}$$

Proof (By Mathematical Induction)

Step 1: For $q = 1$, it follows directly from Eqn.(16) as follows:

$$\begin{aligned} b_0^{(1)} &= \tfrac{1}{\sqrt{2^{n+1}}} \left(1 - 2\tfrac{M}{N}\right) \\ &= \tfrac{b_0^{(0)}}{\sqrt{2}} \left(1 - 2\tfrac{M}{N}\right) \end{aligned} \tag{58}$$

Step 2: Assume the relation is true for $q = t$:

$$b_0^{(t)} = \frac{b_0^{(0)}}{\left(\sqrt{2}\right)^t} \left(1 - 2\frac{M}{N} \right)^t \tag{59}$$

Step 3: Prove for $q = t + 1$:

By definition,

$$\langle \alpha_{t+1} \rangle = \frac{b_0^{(t)}}{\sqrt{2}} \left(1 - \frac{M}{N} \right) \tag{60}$$

And,

$$\begin{array}{l} b_0^{(t+1)} = 2\left\langle \alpha_{t+1} \right\rangle - \frac{b_0^{(t)}}{\sqrt{2}} \\ \quad = \frac{2b_0^{(t)}}{\sqrt{2}}\left(1 - \frac{M}{N}\right) - \frac{b_0^{(t)}}{\sqrt{2}} \\ \quad = \frac{b_0^{(t)}}{\sqrt{2}}\left(1 - 2\frac{M}{N}\right) \end{array} \tag{61}$$

Substitute by the assumption, it directly gives the term for $q = t + 1$,

$$b_0^{(t+1)} = \frac{b_0^{(0)}}{\left(\sqrt{2}\right)^{t+1}}\left(1 - 2\frac{M}{N}\right)^{t+1} \tag{62}$$

Now, to prove that the probability of success of the proposed algorithm after q iterations can take this form:

$$P_s^{(q)} = \left(\frac{M}{N} - 1\right)\left(1 - \frac{2M}{N}\right)^{2q} + 1. \tag{63}$$

Given by definition that,

$$P_s^{(q)} = M\left(\left(a_i^{(q)}\right)^2 + \left(b_i^{(q)}\right)^2\right); i = 0, 1, 2, ..., 2^{q-1} - 1. \tag{64}$$

Proof (By Mathematical Induction)

Step 1: For $q = 1$, it is straight forward from Eqn.(24).

Step 2: Assume the relation is true for $q = t$,

$$\begin{array}{l} P_s^{(t)} = M\left(\left(a_i^{(t)}\right)^2 + \left(b_i^{(t)}\right)^2\right); i = 0, 1, ..., 2^{t-1} - 1. \\ \quad = \left(\frac{M}{N} - 1\right)\left(1 - \frac{2M}{N}\right)^{2t} + 1 \end{array}. \tag{65}$$

Step 3: Proof for $q = t + 1$,

By definition,

$$\begin{aligned} P_s^{(t+1)} &= M\left(\left(a_{0\to 2^t-1}^{(t+1)}\right)^2 + \left(b_{0\to 2^t-1}^{(t+1)}\right)^2\right) \\ &= M\left(2^{t+2}\left\langle \alpha_{t+1}\right\rangle^2 + \left(a_{0\to 2^{t-1}-1}^{(t+1)}\right)^2 + 2^{t+2}\left\langle \alpha_{t+1}\right\rangle^2 + \left(b_{0\to 2^{t-1}-1}^{(t+1)}\right)^2\right) \\ &= M2^{t+3}\left\langle \alpha_{t+1}\right\rangle^2 + P_s^{(t)} \end{aligned} \tag{66}$$

Using Eqn.(57), we have,

$$\begin{aligned} \left\langle \alpha_{t+1}\right\rangle^2 &= \left(\frac{b_0^{(t)}}{\sqrt{2}}\left(1-\frac{M}{N}\right)\right)^2 \\ &= \left(\frac{b_0^{(0)}}{\left(\sqrt{2}\right)^{t+1}}\right)^2\left(1-\frac{M}{N}\right)^2\left(1-\frac{2M}{N}\right)^{2t} \\ &= \frac{1}{N2^{t+1}}\left(1-\frac{M}{N}\right)^2\left(1-\frac{2M}{N}\right)^{2t} \end{aligned} \tag{67}$$

Substitute in Eqn.(66), we get,

$$\begin{aligned} P_s^{(t+1)} &= M2^{t+3}\frac{1}{N2^{t+1}}\left(1-\frac{M}{N}\right)^2\left(1-\frac{2M}{N}\right)^{2t} + \left(\frac{M}{N}-1\right)\left(1-\frac{2M}{N}\right)^{2t} + 1 \\ &= \left(\frac{M}{N}-1\right)\left(1-\frac{2M}{N}\right)^{2t}\left(\frac{4M}{N}\left(\frac{M}{N}-1\right)+1\right) + 1 \\ &= \left(\frac{M}{N}-1\right)\left(1-\frac{2M}{N}\right)^{2(t+1)} + 1 \end{aligned} \tag{68}$$