Large Language Model-guided Quantum MaxSAT Algorithm Design

Aman Chawla
Indian Institute of Technology Delhi

In this paper, we use a computer-inspired design process to develop a new random walk based quantum algorithm for the satisfiability problem MaxSAT. We perform a quantum analysis and compute the measurement probability of the various outcomes of the algorithm. Finally, we run MATLAB simulations to gauge the performance and find that the algorithm performs substantially better than pure guessing.

I. Introduction

Solving problems using quantum computers really took-off after Peter Shor introduced the factoring algorithm about three decades ago [1]. Since quantum computers work on the principles of quantum physics, rather than on a digital and classical abstraction, one expects some difficulties associated with their implementation but also some efficiencies resulting from their use.

MaxSAT is a part of a group of problems that includes MaxCUT and related optimization problems. In the MaxSAT problem, one has variables which are binary in nature. One is assigned the task of finding the variable values that maximize the number of satisfied clauses of a Boolean function [2]. To understand what a clause is one only need recall what a truth table is from digital logic. Each row of the truth table essentially corresponds to a clause. In this sense MaxSAT is tangentially related to the Quine-McCluskey algorithm and Karnaugh Maps that help to systematize the procedure of finding a simple expression for a truth table and in the process identify minterms and maxterms which are akin to the clauses in a MaxSAT problem [3].

The MaxSAT problem has been the subject of intense study since its introduction, more so classically than in the quantum context. The use of large language models, machine learning in general and artificial intelligence, is becoming more prevalent in fields related to quantum physics. See [4] for a discussion of the contribution that these fields can make towards the scientific discourse. In this paper we are concerned with providing a quantum solution to the MaxSAT problem using random walks. Our paper is organized as follows. In Section II we present a review of the relevant literature. In Section III we provide the basic formulation of the algorithm suggested by the model under guidance. In Section IV we perform a novel and fundamental quantum analysis of the algorithm. In Section V we perform MATLAB simulations of the algorithm's outcomes and in Section VI we conclude with a discussion.

II. Literature Review

In this section we choose three current papers on quantum algorithms and their use and discuss them briefly. Finally, we point out their differences with our work.

In the paper by Winderl et. al. [5], the authors discuss an approach to MaxCUT which is a variant of MaxSAT. They build on prior work and propose three major improvements beyond prior work. Nevertheless, they acknowledge that the problem class they are focussed on, optimizing variational quantum algorithms, remains an NP-hard challenge. Note that MaxCUT can be solved using variational quantum eigensolvers. In [6], the authors consider the Grover adaptive search and provide a number of strategies to reduce the gate count in the algorithm's implementation. They also study the convergence of these algorithms and propose strategies which enhance the convergence by reducing the state space size and also reducing the gate count. In [7], the authors apply quantum computing algorithms to the solution of complex financial services problems. Their principal study is the Variational Quantum Eigensolver and they implement it on IBM's quantum computers to study its performance on portfolio optimization problems.

While the comparison can't be direct, some of the major differences of Winderl et. al. with our work include the use of an encoding function R_f, the use of a diagonal gate based on R_f, the use of a deterministic AltOPT procedure to search the space instead of a random walk, and finally, the use of interpolation to 'treat' a discrete solution space as a continuous one. None of these features are present in our approach. Impressively nevertheless, they compare their algorithm with a range of other algorithms such as basin hopping, NFT and a genetic algorithm. Their results show that their strategy performs quite well in comparison.

The work of Yuki Sano et. al. has relevance for our future work but is distinct since we were interested in the design of a solution to a specific problem - MaxSAT - and not with optimizing the solution. Nevertheless, their optimized Grover search can be useful as a replacement to our reflection step. Finally, the paper of Ginï¿œs Carrascal et. al. does not design and mathematically analyze a new algorithm, as we do in the present paper.

III. Basic Formulation

The goal of our study is to design a quantum algorithm under the model of quantum computation used in [1]. This model has three main steps or components. In the first step, an entangled state over registers is prepared. In the second step unitary evolution is performed on the quantum state and in the final step, the evolved state is measured.

For the problem of finding a MaxSAT solution, we instruct the LLM to design a suitable unitary evolution. It provides us with various options and we narrow down the choice to a random-walk based evolution based on the work of Julia Kempe [8] with which we are previously familiar. The LLM makes the remarkable suggestion of combining the random walk step operator and an oracle, with a Grover-type reflection operator to speed up the search across the state space. The final goal is of finding the target state that represents the maximal value or a value closest to the maximal value of a weighted sum of m known clauses.

In the initial run (Figure **??**) we use random weights, but in the remainder of the paper all the weights are held at unity. To attain our goal we leverage the structure of quantum walks on a Hilbert space associated with the n-qubit system.

A. Definition of the Classical MaxSAT Problem

We first define what a function and a clause mean in the context of MaxSAT.

1) Function: A function refers to a Boolean formula that is composed of several clauses. The function is typically expressed in Conjunctive Normal Form (CNF) [2], which means it is a conjunction (AND) of clauses. Mathematically, it can be represented as:

$$F(x_1, x_2, ..., x_n) = C_1 \bigwedge C_2 \bigwedge ... \bigwedge C_m \tag{1}$$

where F is the Boolean function, x_i are the variables, and C_j are the clauses.

2) Clause: A clause is a disjunction (OR) of one or more literals, where a literal is either a variable x_i or its negation $\neg x_i$. A clause is satisfied if at least one of its literals is true. For example, a clause like:

$$C_i = (x_1 \bigvee \neg x_2 \bigvee x_3) \tag{2}$$

is satisfied if at least one of x_1, $\neg x_2$, or x_3 is true.

3) MaxSAT Definition: In classical MaxSAT, we are trying to maximize the number of satisfied clauses of a Boolean function.

B. Quantum Problem Setup

The key components of the problem setup are as follows [9].

1) Quantum State Space: The system has n qubits, leading to a computational basis of dimension 2^n. Each basis state $|x\rangle$ corresponds to a specific assignment of 0 or 1 to the n variables. For example, if $n = 3$, the basis states will be eight in number, given explicitly as the set $\{|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle\}$.
2) Clauses: There are m clauses, each contributing to a (weighted) sum - the Boolean function - based on the specific assignment of variables in $|x\rangle$. The goal is to find the state $|x*\rangle$ that either maximizes or is closest to maximizing this weighted sum. The sum is designated by the functional form $f(\cdot)$.
3) Quantum Walk: We will design V based on a quantum walk that efficiently explores the computational basis, guiding the evolution towards the desired target state $|x*\rangle$. Next, we discuss our approach in detail.

C. Weighted Sum Objective Function

We define an objective function $f(x)$ that represents the weighted sum of the clauses for a given assignment x. The goal is to find the state $|x*\rangle$ where $f(x*)$ is maximal.

D. Hilbert Space and Oracle Function

Hilbert Space : The state space is a complex n-dimensional space C^n, with $2n$ real dimensions, where each basis state $\mid x\rangle$ corresponds to a potential solution.

Oracle Function O_f : We define an oracle unitary O_f that marks the basis states based on the value of $f(x)$ [10]. Specifically, if the value of $f(x)$ is larger than the value of $f(x)$ in all previous steps, the state x is marked as red and previous marked red states are flipped to blue. For another example: $O_f \mid x\rangle = -1^{f(x)} \mid x\rangle$. This oracle flips the phase of the state if $f(x)$ is above a certain threshold, thereby marking promising states.

E. Quantum Random Walk on the Hypercube

Consider a quantum walk on the n-dimensional hypercube, where each vertex corresponds to a state $|x\rangle$, and edges represent transitions between states that differ by one bit. Define a step operator S that applies a quantum walk on this hypercube, transitioning between adjacent vertices (states) [11].

The step operator could be designed as: $S = \sum_x \sum_{i=1}^{n} \mid x \bigoplus e_i\rangle \langle x \mid \otimes \mid i\rangle \langle i \mid$ where e_i is the basis vector corresponding to flipping the i-th bit.

F. Combining Walk with Oracle O_f

The key idea is to combine the quantum walk operator S with the oracle O_f. The evolution of the quantum state is then driven by the repeated application of these operations: $W = O_f S$. This unitary W effectively "guides" the quantum walk by marking promising states (based on $f(x)$) and then spreading amplitude towards those states.

G. Reflection and Iterative Amplification

a) Reflection about the Average : We introduce a reflection operation R that reflects the current state about the average state [12]. This is akin to Grover's diffusion operator: $R = 2|\varphi_0\rangle \langle\varphi_0| - 1$ where $\mid \varphi_0\rangle$ is the uniform superposition state [13].

b) Iterative Application : The final procedure involves iteratively applying the composite operator $V = RW$ several times. This amplifies the amplitude of the states that are marked by the oracle (i.e., states close to the maximum value of $f(x)$.

H. Measurement

After sufficient iterations (the optimal number can be determined based on the problem size n), the final state is measured. The probability of obtaining the state $\mid x*\rangle$ that maximizes or is closest to maximizing $f(x)$ should be significantly amplified as explained in Subsection III-F.

I. Overall Algorithm

Initialize: Start with the uniform superposition of all basis states. $\mid \varphi_0\rangle = \frac{1}{2^n} \sum_0^{2^{n-1}} \mid x\rangle$.

Quantum Walk and Oracle: Apply the composite operator $W = O_f\ S$ to guide the quantum walk towards states with higher objective values $f(x)$.

Amplitude Amplification: Use a reflection operator R to amplify the probability amplitude of promising states. The overall unitary $V = RW$ is applied iteratively.

Measurement: Measure the final state, which will most likely collapse to the state $\mid x*\rangle$ that maximizes $f(x)$ or is close to the maximum. This approach leverages the Kempe quantum random walk technique combined with amplitude amplification to efficiently find the target state $\mid x*\rangle$ in a high-dimensional search space.

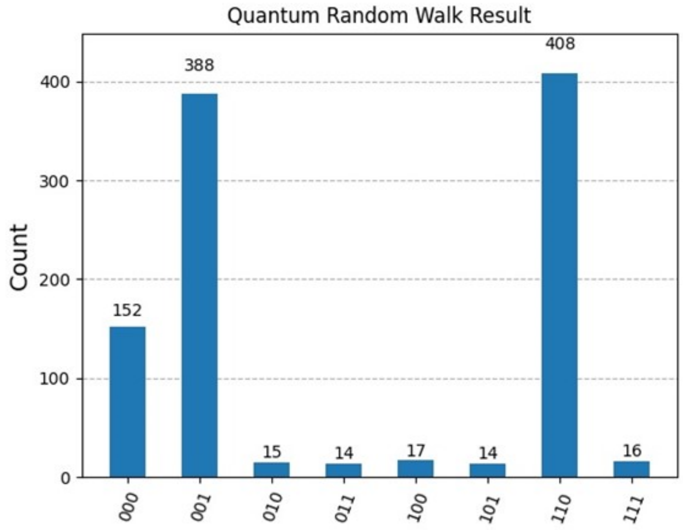

Fig. 1

J. Results

The weights corresponding to each clause were set randomly. The algorithm executed and provided the following output, shown in Figure 1. The counts obtained for each state or clause are clearly indicated. Based on the graph, the clauses "001" and "110" are highly probable. In other words, those specific variable assignments maximize the Boolean function.

IV. Quantum Analysis

In this section, we will undertake a full quantum mechanical analysis of the algorithm presented in Section III. We begin our analysis with a starting state which is a "cat" state. The starting state is specified as a standard superposition

$$|\phi_0\rangle = \frac{1}{2^n}\sum_{x=0}^{2^n-1}|x\rangle. \tag{3}$$

A. Step Operator

Next, we consider the random walk step operator from Kempe's work (Equation (12) in [8]). The operator may be written as

$$S = \sum_x \sum_{i=1}^{n} |x \oplus e_i\rangle\langle x| \otimes |i\rangle\langle i| \tag{4}$$

where e_i is a quantity that is linked with the ket $|i\rangle$. For example if it is a spin-half qubit space so $n = 2$, then $|i\rangle$ may stand for spin up when $i = 1$ and spin down when $i = 2$. And, relatedly, e_1 could be a positive 1 and e_2 a negative 1. The first step is to figure out the application of the step operator on the $|\phi_0\rangle$ state. We note that

$$S|\phi_0\rangle = \sum_x \sum_{i=1}^{n} |x \oplus e_i\rangle\langle x| \otimes |i\rangle\langle i| \left(\frac{1}{2^n}\sum_{x=0}^{2^n-1}|x\rangle\right) \tag{5}$$

by simply plugging in. We change the second label x to a y in order to avoid confusing the two enumerations. We have,

$$S|\phi_0\rangle = \sum_x \sum_{i=1}^{n} |x \oplus e_i\rangle\langle x| \otimes |i\rangle\langle i| \left(\frac{1}{2^n}\sum_{y=0}^{2^n-1}|y\rangle\right). \tag{6}$$

As the next step we explicitly write out (expand out) the basis kets $|i\rangle$. We have,

$$S|\phi_0\rangle = \sum_x |x\oplus e_1\rangle\langle x| \otimes |1\rangle\langle 1| + \sum_x |x\oplus e_2\rangle\langle x| \otimes |2\rangle\langle 2| + \tag{7}$$

$$\ldots + \sum_x |x\oplus e_n\rangle\langle x| \otimes |n\rangle\langle n| \left(\frac{1}{2^n}\sum_{y=0}^{2^n-1}|y\rangle\right). \tag{8}$$

Here the x values index the positions on the line if it is a one dimensional random walk. The initial state is a quantum superposition of all the line indices. Thus x can follow the same pattern as y. The direct sum terms e_i represent actions that transition the ket $|y\rangle$ to the ket $|y \oplus e_i\rangle$. This relies on the orthonormality of the $|y\rangle$ kets. It is therefore more illustrative if we rewrite the expansion (Equation (8)) as follows, by switching the order of the tensor-ed operators. We have,

$$S|\phi_0\rangle = |1\rangle\langle 1| \otimes \sum_x |x\oplus e_1\rangle\langle x| + |2\rangle\langle 2| \otimes \sum_x |x\oplus e_2\rangle\langle x| + \tag{9}$$

$$\ldots + |n\rangle\langle n| \otimes \sum_x |x\oplus e_n\rangle\langle x| \left(\frac{1}{2^n}\sum_{y=0}^{2^n-1}|y\rangle\right). \tag{10}$$

Next we move the cat state into the operators on the left. We are precluded from doing this however, since the cat state is a state in the Hilbert space of the line. However, the operator being applied to it is an operator in the tensor product Hilbert space of the ancilla system and the line; this is clear from the first term on the RHS of Equation (10). So we must augment the cat state with an ancilla. The ancilla can be in the Hilbert space which has the number kets $|1\rangle$ through $|n\rangle$ as its basis. Thus we rewrite the cat state we are operating on as

$$|k\rangle \otimes |\phi_0\rangle \tag{11}$$

for some specific number state labeled by "k." We operate on this state by the step operator S. We obtain,

$$= |1\rangle\langle 1| \otimes \sum_x |x\oplus e_1\rangle\langle x| + |2\rangle\langle 2| \otimes \sum_x |x\oplus e_2\rangle\langle x| + \tag{12}$$

$$\ldots + |n\rangle\langle n| \otimes \sum_x |x\oplus e_n\rangle\langle x| \left[|k\rangle \otimes |\phi_0\rangle\right]. \tag{13}$$

We move each subsystem to its appropriate operator,

$$= |1\rangle\langle 1|k\rangle \otimes \sum_x |x\oplus e_1\rangle\langle x|\phi_0\rangle + |2\rangle\langle 2|k\rangle \otimes \sum_x |x\oplus e_2\rangle\langle x|\phi_0\rangle + \tag{14}$$

$$\ldots + |n\rangle\langle n|k\rangle \otimes \sum_x |x\oplus e_n\rangle\langle x|\phi_0\rangle. \tag{15}$$

This simplifies since $\langle j|k\rangle = \delta_{jk}$ (the Kronecker delta) and $\langle x|\phi_0\rangle = \frac{1}{2^n}\sum_{y=0}^{2^n-1}\langle x|y\rangle = \frac{1}{2^n}\sum_{y=0}^{2^n-1}\delta_{xy}$ where δ_{xy} is 1 unless x and y labels are distinct, in which case it is 0. We have $\langle x|\phi_0\rangle = \frac{1}{2^n}$. So the simplified expression is,

$$S\left(|k\rangle \otimes |\phi_0\rangle\right) = \left[|k\rangle \otimes \frac{1}{2^n}\sum_x |x \oplus e_k\rangle\right]. \tag{16}$$

In other words, the random walk moves the cat state to a new superposition state. Strictly speaking, to characterize the action of the operator S, we must show its operation on the basis states $\{|k\rangle \otimes |0\dots0\rangle, |k\rangle \otimes |0\dots1\rangle, |k\rangle \otimes |0\dots10\rangle, \dots\}$. The binary representations inside the line position kets represent the position number in binary. Here e_k flips the k-th binary digit via the direct sum. This means that the final states after the action of S will be $\{|k\rangle \otimes |0\dots0 \oplus e_k\rangle, |k\rangle \otimes |0\dots1 \oplus e_k\rangle, |k\rangle \otimes |0\dots10 \oplus e_k\rangle, \dots\}$. Just to reiterate, the n bits represent 2^n line positions (since each bit represents two positions via its two states 0 and 1) and e_k flips the k-th bit. Since there are only n bits, k runs from 1 through n.

In the next sections we will analyze the subsequent application of the reflection and oracle operators.

B. Oracle Operator

We may begin our investigation with the oracle defined in Section III-D. There we said,

$$O_f|x\rangle = (-1)^{f(x)}|x\rangle. \tag{17}$$

So we fix a function $f(\cdot)$ which assigns an integer value to every n-bit binary number (state label) that is input to it. In practice, this function will be tailored to the MaxSAT objective (see Section III-C). Then, the oracle operator can be characterized by its action on every n-bit line position ket. In general,

$$O_f|0\dots0\rangle = (-1)^{f(0)}|0\dots0\rangle, \tag{18}$$

$$O_f|0\dots01\rangle = (-1)^{f(1)}|0\dots01\rangle \tag{19}$$

and so on. Consider the basis state $|k\rangle \otimes |0\dots0\rangle$. The step operator transforms it into $|k\rangle \otimes |0\dots0 \oplus e_k\rangle$. Since the oracle acts only on the line kets, we can augment it by the identity map I_n on the number kets $|k\rangle$. Thus the overall oracle becomes $I_n \otimes O_f$. The final state after the action of the step and then the oracle therefore is

$$\left(I_n \otimes O_f\right)\left(|k\rangle \otimes |0\dots0 \oplus e_k\rangle\right) = |k\rangle \otimes (-1)^{f(0\dots0\oplus e_k)}|0\dots0\oplus e_k\rangle. \tag{20}$$

C. Reflection Operator

Consider the reflection operator defined as

$$R = 2|\,\varphi_0\rangle\langle\varphi_0| - I_n. \tag{21}$$

Consider a generic state $|\psi\rangle$ which we can decompose into a part along $|\varphi_0\rangle$ and a part orthogonal to $|\varphi_0\rangle$. We write,

$$|\psi\rangle = \left(|\varphi_0\rangle\langle\varphi_0| + |\varphi_0^\perp\rangle\langle\varphi_0^\perp|\right)|\psi\rangle = \langle\varphi_0|\psi\rangle|\varphi_0\rangle + \langle\varphi_0^\perp|\psi\rangle|\varphi_0^\perp\rangle \tag{22}$$

which can be operated upon by R to obtain,

$$R|\psi\rangle = \left[2|\varphi_0\rangle\langle\varphi_0| - I\right]\left(|\varphi_0\rangle\langle\varphi_0| + |\varphi_0^\perp\rangle\langle\varphi_0^\perp|\right)|\psi\rangle \tag{23}$$

which simplifies to

$$R|\psi\rangle = 2|\varphi_0\rangle\langle\varphi_0|\psi\rangle - |\varphi_0\rangle\langle\varphi_0|\psi\rangle - |\varphi_0^\perp\rangle\langle\varphi_0^\perp|\psi\rangle \tag{24}$$

which in turn simplifies to

$$R|\psi\rangle = \langle\varphi_0|\psi\rangle|\varphi_0\rangle - \langle\varphi_0^\perp|\psi\rangle|\varphi_0^\perp\rangle \tag{25}$$

which should be compared with Equation (22). It is apparent that R has reflected the input state about the cat state $|\varphi_0\rangle$ which acts as the axis of reflection.

We take the final state after the action of the random walk step operator and oracle operator, given in Equation (20), and pass it through the reflection operator. Omitting the intermediate steps which are straightforward in the interest of brevity, we may write down the final state as,

$$|k\rangle \otimes \left[(-1)^{f(0\dots0\oplus e_k)}\left(2^{1-n} - 1\right)|0\dots0 \oplus e_k\rangle\right] \tag{26}$$

D. Measurement

1) Single Round: As the final operation, we must measure the final state after the first round. Consider a projection measurement. The state to be measured resides in a state space of size 2^n. So we want at least 2^n measurement outcomes which will tell us the basis-subspace to which the state to be measured belongs and with what probability. The probability will be in terms of the function $f(\cdot)$.

We write down the measurement operators as the projectors onto the basis states. Thus M_0 is the projector onto $|0\dots0\rangle$, M_1 is the projector onto $|0\dots01\rangle$, and $M_{2^{n-1}}$ is the projector onto $|1\dots1\rangle$. So we can construct the operators as follows:

$$M_0 = |0\dots0\rangle\langle0\dots0|, \tag{27}$$

$$M_1 = |0\dots1\rangle\langle0\dots1| \tag{28}$$

and so on

$$M_{2^{n-1}} = |1\dots1\rangle\langle1\dots1|. \tag{29}$$

Assuming the final state is given by Equation (26) with $k = 1$, the probability of getting the result m is

$$p(m) = (-1)^{f(0\dots0\oplus e_1)}\left(2^{1-n} - 1\right)\langle0\dots0\oplus e_1|M_m(-1)^{f(0\dots0\oplus e_1)} \tag{30}$$

$$\dots\left(2^{1-n} - 1\right)|0\dots0 \oplus e_1\rangle. \tag{31}$$

Next we collect the coefficient factors up front to get,

$$p(m) = \left[(-1)^{f(0\dots0\oplus e_1)}\left(2^{1-n} - 1\right)\right]^2\langle0\dots0\oplus e_1|m\rangle\langle m|0\dots0\oplus e_1\rangle \tag{32}$$

which can be simplified to

$$p(m) = \left[(-1)^{f(0\dots1)}\left(2^{1-n} - 1\right)\right]^2\langle0\dots1|m\rangle\langle m|0\dots1\rangle. \tag{33}$$

Clearly, $p(m)$ is zero unless m is 1. In general the input state will not be given by Equation (26) but some superposition

$$\sum_{j=0}^{2^{n-1}}|k\rangle \otimes \left[(-1)^{f(j\oplus e_k)}\left(2^{1-n} - 1\right)|j \oplus e_k\rangle\right]. \tag{34}$$

Then we find that

$$p(m) = \left[(-1)^{f(j'\oplus e_k)}\left(2^{1-n}-1\right)\right]^2 \langle j'\oplus e_k|m\rangle\langle m|j'\oplus e_k\rangle \tag{35}$$

where j' is the j such that

$$j'\oplus e_k = m \tag{36}$$

so that

$$p(m) = \left[(-1)^{f(m)}\left(2^{1-n}-1\right)\right]^2. \tag{37}$$

Recall that here n is the dimension of the ancilla system.

2) Two Rounds: We start with the state

$$|k\rangle\otimes\left[(-1)^{f(0\ldots0\oplus e_k)}\left(2^{1-n}-1\right)|0\ldots0\oplus e_k\rangle\right] \tag{38}$$

instead of the cat state and subject it to the step operator. We obtain,

$$|k\rangle\otimes\left[(-1)^{f(0\ldots0\oplus e_k\oplus e_k)}\left(2^{1-n}-1\right)|0\ldots0\oplus e_k\oplus e_k\rangle\right]. \tag{39}$$

Since flipping the same bit two times yields no change to the bit, we have,

$$|k\rangle\otimes\left[(-1)^{f(0\ldots0)}\left(2^{1-n}-1\right)|0\ldots0\rangle\right]. \tag{40}$$

Next we subject this last result to the oracle and obtain

$$|k\rangle\otimes\left[(-1)^{2\cdot f(0\ldots0)}\left(2^{1-n}-1\right)|0\ldots0\rangle\right]. \tag{41}$$

The last operator to be applied is the reflection and we obtain,

$$|k\rangle\otimes[2\cdot(-1)^{2\cdot f(0\ldots0)}\left(2^{1-n}-1\right)|\phi_0\rangle\langle\phi_0|0\ldots0\rangle-(-1)^{2\cdot f(0\ldots0)} \tag{42}$$

$$\left(2^{1-n}-1\right)|0\ldots0\rangle] \tag{43}$$

which simplifies to

$$|k\rangle\otimes[2^{(1-n)}\cdot(-1)^{2\cdot f(0\ldots0)}\left(2^{1-n}-1\right)|0\ldots0\rangle-(-1)^{2\cdot f(0\ldots0)} \tag{44}$$

$$\left(2^{1-n}-1\right)|0\ldots0\rangle] \tag{45}$$

and finally collecting terms,

$$|k\rangle\otimes\left[(2^{(1-n)}-1)\cdot(-1)^{2\cdot f(0\ldots0)}\left(2^{1-n}-1\right)|0\ldots0\rangle\right]. \tag{46}$$

Finally, we just need to measure this state. The post-measurement probability can be written down by inspection as,

$$p(m) = \left[(-1)^{f(m)}\left(2^{1-n}-1\right)\right]^4 \tag{47}$$

3) Multiple Rounds (Even Number): If we apply an even number r of rounds to the cat state, we can make some observations:

1. Since the number of rounds is even, an even number of e_k will be direct summed with the zero ket, leaving it unchanged.

2. Since the oracle will have been applied r times, there will be a coefficient of r instead of the 2 that appears in Equation (41).

3. Since the ket will be the zero ket, once the R operator is applied, which is based on the cat state, only the zero ket of the cat state will survive along with a coefficient of $\frac{1}{2^n}$.

Finally, therefore we will obtain a state of the form,

$$|k\rangle\otimes\left[(2^{(1-n)}-1)^r\cdot(-1)^{r\cdot f(0\ldots0)}|0\ldots0\rangle\right] \tag{48}$$

which would then be subject to measurement, to obtain

$$p(m) = \left[(-1)^{f(m)}\left(2^{1-n}-1\right)\right]^{2r} \tag{49}$$

4) Multiple Rounds (Odd Number): This would be a slightly more complicated case than the previous ones. We just need to observe that an odd number of flips of the same bit is equal to a single flip of the bit. Thus the final state before measurement can be written as,

$$|k\rangle\otimes\left[(2^{(1-n)}-1)^r\cdot(-1)^{r\cdot f(0\ldots01)}|0\ldots01\rangle\right]. \tag{50}$$

When the measurement is performed, we obtain,

$$p(m) = \left[(-1)^{f(m)}\left(2^{1-n}-1\right)\right]^{2r}. \tag{51}$$

This can then be treated as the general expression.

V. Simulations in MATLAB

Since we have a final expression for the probabilities of the various outcomes, Equation (51), we can perform simulations of the expression for various types of Boolean functions. The simulations utilize a Hilbert space of dimension 6. The probabilities are plotted normalized, for an algorithm run of 120 rounds.

A. Linear Function

The linear function assigns values to various measurement outcomes m using a linear equation (see Figure 2). Figure 3 depicts the results.

B. Sinusoidal Low Frequency Function

In this subsection we investigate a low frequency sinusoidal function depicted in Figure4. Figure 5 shows the results.

C. Sinusoidal High Frequency Function

In this subsection we investigate a high frequency sinusoidal function depicted in Figure 6. Figure 7 shows the results.

D. Decaying Sinusoidal Function

In this subsection we investigate an exponentially decaying sinusoidal function depicted in Figure 8. Figure 9 shows the results.

E. Decaying Cosinusoidal Function

In this subsection we investigate an exponentially decaying cosinusoidal function shown in Figure 10. Figure 11 shows the results.

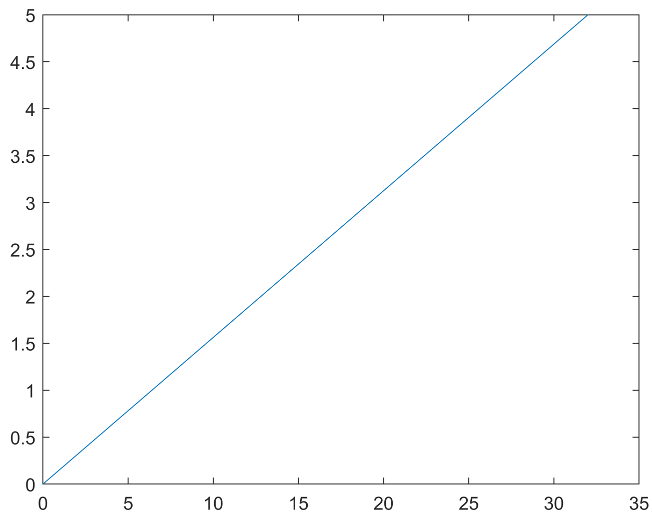

Fig. 2

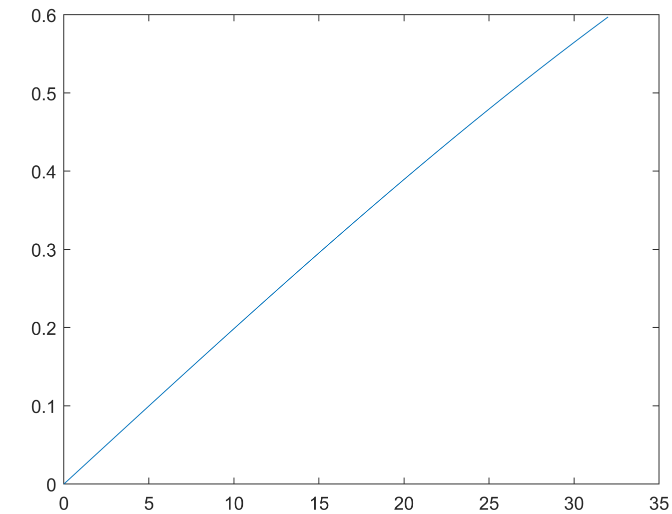

Fig. 4

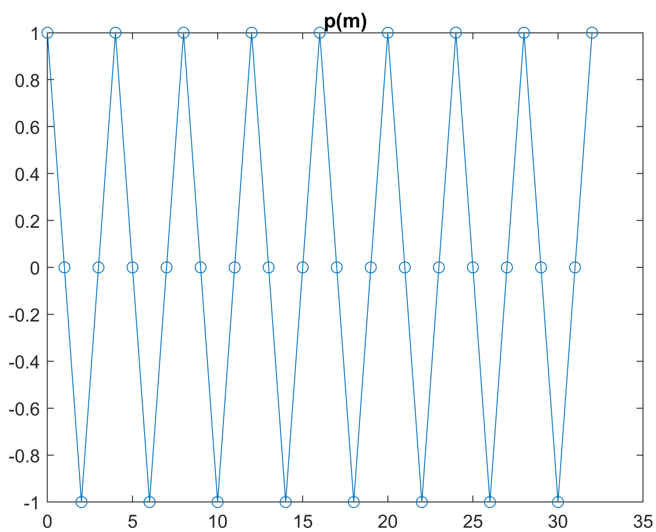

Fig. 3

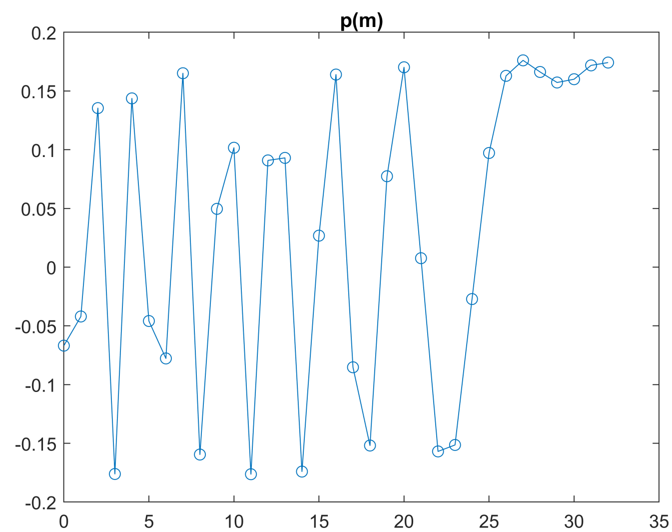

Fig. 5

F. Full Behavior

Finally to succinctly capture the full behavior of the algorithm, in this subsection we choose the sinusoidal low frequency (Subsection V-B) function, and vary n, computing the value of $p(m*)$ where $m*$ is the argument that maximizes $p(m)$. Figure 12 shows the results.

The figure shows that as the number of qubits is increased, the performance of both random guessing and our algorithm decays. However our algorithm always outperforms random guessing by a substantial margin; the margin however decreases with increasing number of qubits considered.

VI. Discussion, Limitations and Conclusion

It is worth remarking that the algorithm presented in this paper follows three steps: state preparation, quantum evolution and measurement and lacks the fourth step, that of classical evolution which is compulsary in the model of universal quantum computation in the brain presented in [14]. The classical evolution in the latter algorithm precedes a repetition of the entire four step process. In that paper it was remarked that the four step procedure therein could yield novel classical-quantum machine learning algorithms. In the present paper however, the repetition takes place within the second step and before the third step. Nevertheless, one may wonder what would happen if one replaced the measurement with a *weak measurement* in the third step [15] and then did some adjustment to the quantum evolution before reapplying it. This adjustment could take the form of modifying the oracle if the measurement indicates that it hasn't correctly picked out or marked the correct states. Such a modification could improve the scaling behavior of the algorithm with increase in qubit numbers.

In simulated annealing there are two schedules: initially the algorithm is run at a high temperature which allows the exploration of a wider solution space and later the temperature is cooled down to converge to the optimal solution. Motivated by

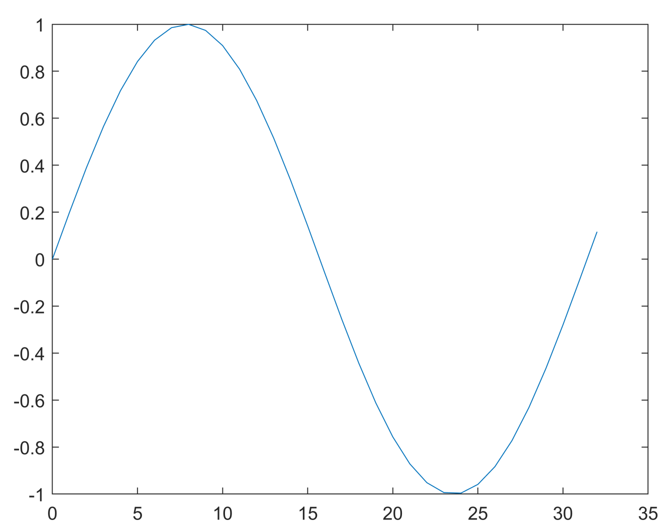

Fig. 6

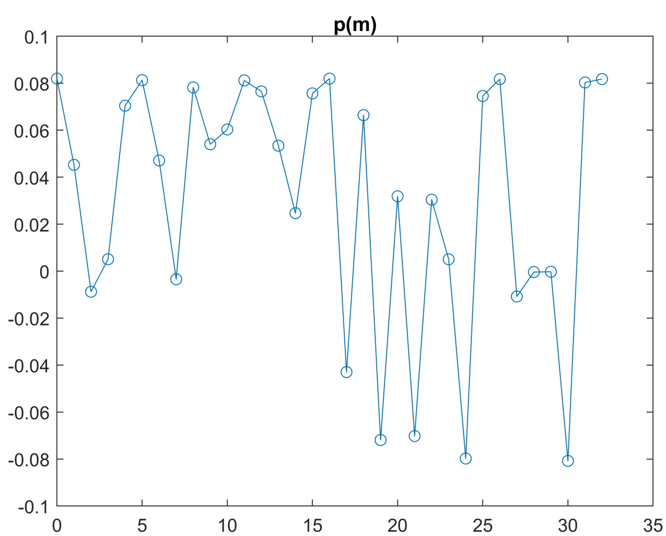

Fig. 7

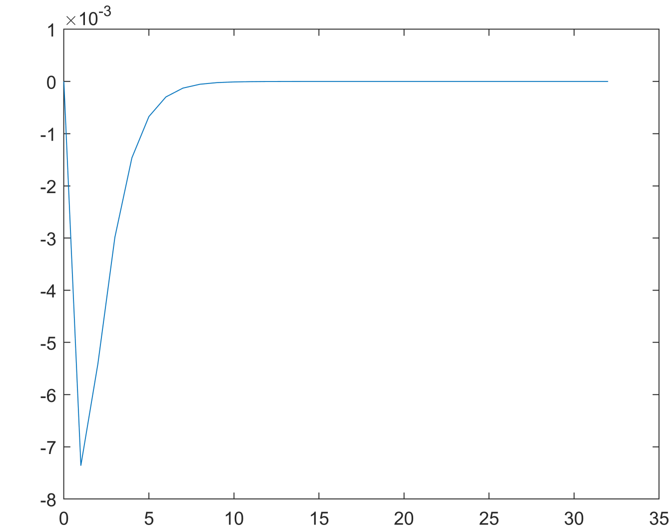

Fig. 8

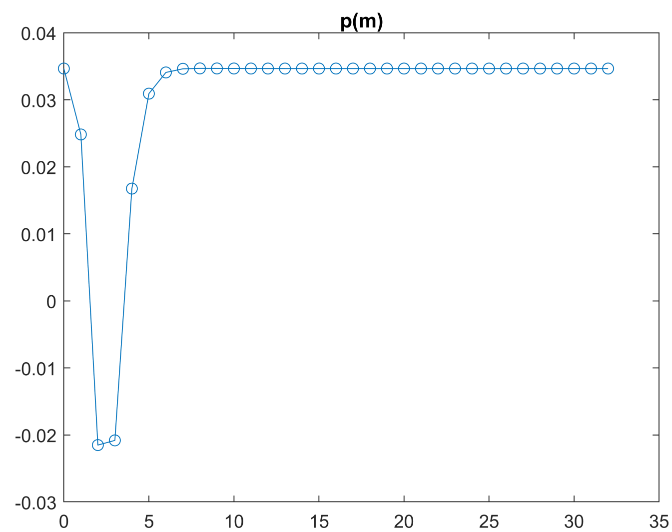

Fig. 9

this, our present algorithm can also be modified to run on two schedules by designing a reflection that initially reflects about the initial state to explore a wide range of possibilities, but later reflects only locally to narrow down to a solution. This may yield improvements to our algorithm and will be explored in future work.

To summarize, our key contribution in this paper is the interleaving of a quantum random walk with a Grover-type reflection operator and multiple runs in order to improve the probability of finding the solution to the MaxSAT problem. Our results indicate that the algorithm performs substantially better than random guessing. Future work as suggested in this section we hope will result in substantial improvements to algorithmic performance. This paper is thus a step forward in the direction of providing dependable quantum gains for classical problems.

Acknowledgment

The author would like to thank Prof. Kolin Paul, CSE, IIT Delhi, for suggesting this problem. The author also thanks OpenAI for making available the LLM ChatGPT.

Data availability statement

This paper has no data to share.

Competing interests statement

The author declares no competing interests.

References

[1] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.

[2] Stephan Hoyer. Quantum random walk search on satisfiability problems. Swarthmore College Dissertation. 2008.

[3] Javad Safaei and Hamid Beigy. Quine-mccluskey classification. In *2007 IEEE/ACS International Conference on Computer Systems and Applications*, pages 404–411. IEEE, 2007.

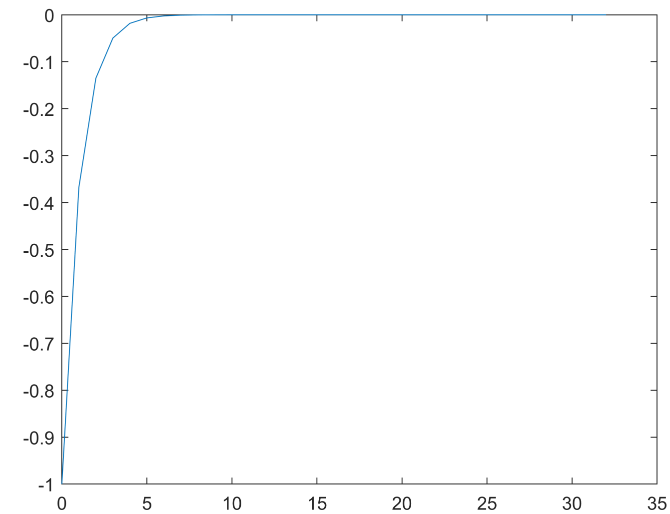

Fig. 10

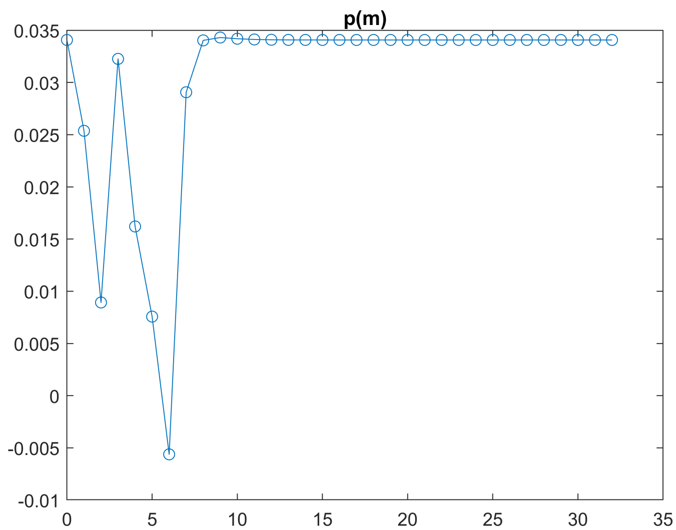

Fig. 11

[4] Mario Krenn, Manuel Erhard, and Anton Zeilinger. Computer-inspired quantum experiments. *Nature Reviews Physics*, 2(11):649–661, 2020.

[5] David Winderl, Nicola Franco, and Jeanette Miriam Lorenz. A comparative study on solving optimization problems with exponentially fewer qubits. *IEEE Transactions on Quantum Engineering*, 2024.

[6] Yuki Sano, Kosuke Mitarai, Naoki Yamamoto, and Naoki Ishikawa. Accelerating grover adaptive search: Qubit and gate count reduction strategies with higher-order formulations. *IEEE Transactions on Quantum Engineering*, 2024.

[7] Ginés Carrascal, Paula Hernamperez, Guillermo Botella, and Alberto del Barrio. Backtesting quantum computing algorithms for portfolio optimization. *IEEE Transactions on Quantum Engineering*, 2023.

[8] Julia Kempe. Quantum random walks: an introductory overview. *Contemporary Physics*, 44(4):307–327, 2003.

[9] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[10] Anthony Chefles. Distributed implementation of standard oracle operators. *Physical Review A Atomic, Molecular, and Optical Physics*, 78(6):062304, 2008.

[11] Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 575–584, 2007.

[12] Anirban Narayan Chowdhury, Yigit Subasi, and Rolando D Somma.

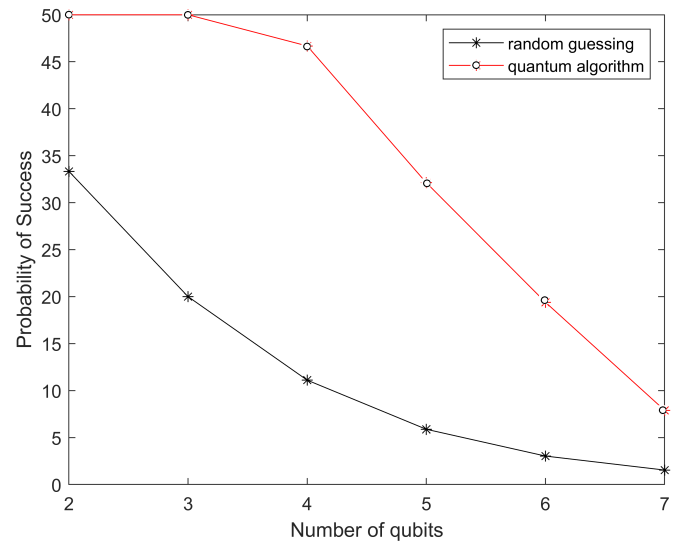

Fig. 12

Improved implementation of reflection operators. *arXiv preprint arXiv:1803.02466*, 2018.

[13] Avatar Tulsi. Faster quantum searching with almost any diffusion operator. *Physical Review A*, 91(5):052307, 2015.

[14] Aman Chawla and Salvatore Domenic Morgera. Mechanism of universal quantum computation in the brain. *Journal of Applied Mathematics and Physics*, 12(2):468–474, 2024.

[15] Yakir Aharonov, Eliahu Cohen, and Avshalom C Elitzur. Foundations and applications of weak quantum measurements. *Physical Review A*, 89(5):052105, 2014.

Figure Legends

Figure 1. First run of the algorithm. The horizontal axis is 'm' or the state in binary expansion and the vertical axis is the count (not '$p(m)$' or probability as in the remainder of the figures) obtained by the algorithm for that state to be the clause-sum maximizing one.

Figure 2. Depiction of a linear function representing the (weighted) sum of the clauses at each possible state with maximum at one extremity. The horizontal axis is the various values of the state variables, expressed in decimal value instead of a binary expansion. The vertical axis is the value of the sum of the clauses at that state.

Figure 3. The figure depicts the probabilities (and not counts like in Figure 1), p(m), for the linear function depicted in Figure 2. The horizontal axis represents the state space and the vertical axis represents the probability that the algorithm selects that particular state in the state space as the function maximizing one.

Figure 4. Depiction of a sinusoidal low frequency function with maximum attained at one extremity of the state space. The horizontal axis is the various values of the state variables, expressed in decimal value instead of a binary expansion. The vertical axis is the value of the sum of the clauses at that state.

Figure 5. The figure depicts the probabilities (and not counts like in Figure 1), p(m), for the sinusoidal low frequency function. The horizontal axis represents the state space and the vertical axis represents the probability that the algorithm selects that particular state in the state space as the function maximizing one.

Figure 6. Depiction of a sinusoidal high frequency function with maximum attained at the state '8' or '1000'. The horizontal axis is the various values of the state variables, expressed in decimal

value instead of a binary expansion. The vertical axis is the value of the sum of the clauses at that state.

Figure 7. The figure depicts the probabilities (and not counts like in Figure 1), $p(m)$, for the sinusoidal high frequency function. The horizontal axis represents the state space and the vertical axis represents the probability that the algorithm selects that particular state in the state space as the function maximizing one.

Figure 8. Depiction of a decaying sinusoidal function. The horizontal axis is the various values of the state variables, expressed in decimal value instead of a binary expansion. The vertical axis is the value of the clause sum at that state.

Figure 9. The figure depicts the probabilities (and not counts like in Figure 1), $p(m)$, for the decaying sinusoidal function. The horizontal axis represents the state space and the vertical axis represents the probability that the algorithm selects that particular state in the state space as the function maximizing one.

Figure 10. Depiction of a decaying cosinusoidal function. The horizontal axis is the various values of the state variables, expressed in decimal value instead of a binary expansion. The vertical axis is the value of the clause sum at that state.

Figure 11. The figure depicts the probabilities (and not counts like in Figure 1), $p(m)$, for the decaying cosinusoidal function. The horizontal axis represents the state space and the vertical axis represents the probability that the algorithm selects that particular state in the state space as the function maximizing one.

Figure 12. Variation of algorithm performance for a fixed function - the sinusoidal low frequency function - with number of qubits or state-space dimension, compared with random guessing. The horizontal axis is the number of qubits used and the vertical axis is the probability of success of the algorithm for the two scenarios - random guessing and quantum MaxSAT.