Quantum solution to the boolean satisfiability problem

Yash Prabhat[1,*]

[1] *Department of Physics, Indian Institute of Science Education and Research, Bhopal 462066, India*

(Dated: May 30, 2025)

abstract

I. INTRODUCTION

The Boolean Satisfiability Problem (SAT), determining whether a propositional formula has a satisfying variable assignment, represents the foundational NP-complete problem with far-reaching implications across automated theorem proving, hardware verification, and artificial intelligence. While 2-SAT admits polynomial-time classical solutions through implication graph analysis, the general k-SAT problem for $k \geq 3$ remains NP-hard, with best-known classical algorithms like DPLL and Schöning's algorithm requiring exponential time in worst-case scenarios [1]. The emergence of quantum computing introduces new paradigms for tackling SAT through quantum parallelism and interference effects, yet existing approaches face fundamental limitations in exploiting formula structure while maintaining favorable complexity scaling.

Classical SAT solvers employ two primary strategies: complete algorithms like conflict-driven clause learning (CDCL) that guarantee solution existence proofs but exhibit exponential worst-case complexity [2], and stochastic local search methods like WalkSAT that trade completeness for practical speed on industrial instances [3]. Quantum adaptations initially focused on Grover's algorithm, achieving $\mathcal{O}(\sqrt{N})$ complexity through amplitude amplification [4], but this quadratic speedup remains insufficient for structured SAT instances with inherent geometric constraints. Recent advances in quantum walk formulations [5] and the Quantum Approximate Optimization Algorithm (QAOA) [6] demonstrated improved clause satisfaction probabilities but lacked mechanisms for hierarchical variable assignment verification.

Our work bridges this gap through ...

The remainder of this paper proceeds as follows:

II. METHODS

A. Boolean Satisfiability Problem (SAT)

The Boolean Satisfiability Problem (SAT) is a foundational decision problem in computational complexity and mathematical logic. Given a Boolean formula composed of variables and logical operators, SAT asks whether there exists an assignment of truth values (TRUE/FALSE) to the variables that makes the entire formula evaluate to TRUE. This problem serves as a cornerstone for understanding NP-completeness and has profound implications across computer science and quantum computing. A Boolean formula ϕ in conjunctive normal form (CNF) is expressed as:

$$\phi = \bigwedge_{i=1}^{m} C_i \quad \text{where each clause } C_i = \bigvee_{j=1}^{k} l_{ij} \tag{1}$$

Here, l_{ij} represents a literal (a variable x_p or its negation $\bar{x}_p$, $p \in \{0, \ldots, n-1\}$). The problem is called k-SAT if every clause C_i contains exactly k literals. For $k \geq 3$, k-SAT is NP-complete, while 2-SAT is solvable in polynomial time. For simplicity, in this work, we shall use the notation of the variable x_j for the literal l_{ij} and their respective data qubit as q_j, $j \in \{1, \ldots, n\}$.

B. The idea

Our approach is based on the fact that a single quantum operator is capable of removing all the states that would not satisfy a particular clause from an equal superposition of all states. It is inspired by the dataset encoding method proposed in [7] for the structured search algorithm. Classically, to remove any arbitrary N entries that would not satisfy a clause would require N queries, hence the quantum advantage. The operator in question is based on the Hadamard operator $\hat{\mathrm{H}}$ defined as:

$$\hat{\mathrm{H}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{2}$$

The action of $\hat{\mathrm{H}}$ on:

$$\begin{aligned} \hat{\mathrm{H}}\left|0\right\rangle = \left|+\right\rangle, \hat{\mathrm{H}}\left|+\right\rangle = \left|0\right\rangle \\ \hat{\mathrm{H}}\left|1\right\rangle = \left|-\right\rangle, \hat{\mathrm{H}}\left|-\right\rangle = \left|1\right\rangle \end{aligned} \tag{3}$$

We only wish to work with $\{0, 1, +\}$ states so we add the Pauli operator $\hat{\mathrm{Z}}$ defined as

$$\hat{\mathrm{Z}} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{4}$$

The Eq. (3) can be modified as

$$\begin{aligned} \hat{\mathrm{H}}\left|0\right\rangle = \left|+\right\rangle, \hat{\mathrm{H}}\left|+\right\rangle = \left|0\right\rangle \\ \hat{\mathrm{Z}} \cdot \hat{\mathrm{H}}\left|1\right\rangle = \left|+\right\rangle, \hat{\mathrm{H}} \cdot \hat{\mathrm{Z}}\left|+\right\rangle = \left|1\right\rangle \end{aligned} \tag{5}$$

* Contact author; yashp20@iiserb.ac.in

Consider a single literal clause C_i as $C_i = l_{ij}$, where $l_{ij} = x_j$ or $l_{ij} = \bar{x}_j$ for $j \in \{1, \ldots, n\}$, where n is the total number of literals. We start with an equal superposition of all 2^n states $|\psi\rangle$ as

$$|\psi\rangle = \hat{\mathrm{H}}^{\otimes n} |0\rangle^{\otimes n} = |+\rangle^{\otimes n} = |+\rangle_1 |+\rangle_2 \ldots |+\rangle_j \ldots |+\rangle_n , \tag{6}$$

where we have n data qubits q_j that process all n literals in parallel. Here, we wish to remove the states that dissatisfy clause C_i, we achieve this via the operation

$$\hat{\mathrm{I}}^{\otimes(j-1)} \otimes \hat{\mathrm{H}} \cdot \hat{\mathrm{Z}} \otimes \hat{\mathrm{I}}^{\otimes(n-j)} |\psi\rangle = |+\rangle_1 |+\rangle_2 \ldots |1\rangle_j \ldots |+\rangle_n , \quad \text{if } l_{ij} = x_j, \tag{7}$$

$$\hat{\mathrm{I}}^{\otimes(j-1)} \otimes \hat{\mathrm{H}} \otimes \hat{\mathrm{I}}^{\otimes(n-j)} |\psi\rangle = |+\rangle_1 |+\rangle_2 \ldots |0\rangle_j \ldots |+\rangle_n , \quad \text{if } l_{ij} = \bar{x}_j, \tag{8}$$

where $\hat{\mathrm{I}}$ is the identity operator

$$\hat{\mathrm{I}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} . \tag{9}$$

Here, we remove 2^{n-1} states that would not satisfy the clause C_i with a single operation. However, we can not use the operations from Eq. (5) for multiple clauses as they are unitary and thus cannot form an AND gate. To overcome this we devise an AND operation $\hat{\mathrm{A}}$ using an ancilla qubit in state $|0_a\rangle$ as a combination of CNOT (CX) gates as:

$$\hat{\mathrm{A}} = \mathrm{CX}_{j,a} \cdot \mathrm{CX}_{a,j} \tag{10}$$

where the unitary operations for $\mathrm{CX}_{j,a}$ and $\mathrm{CX}_{a,j}$ are

$$\mathrm{CX}_{j,a} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \text{ and } \mathrm{CX}_{a,j} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} . \tag{11}$$

$\hat{\mathrm{A}}$ acts on one data qubit q_j and ancilla a, set to state $|0_a\rangle$. The quantum circuit for the operator $\hat{\mathrm{A}}$ is given in Fig. 1.

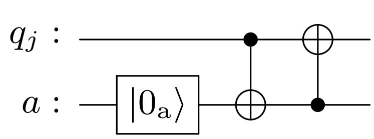

FIG. 1. Quantum circuit for operator $\hat{\mathrm{A}}$.

The action of $\hat{\mathrm{A}}$ on respective qubits in the concerned states can be written as

$$\begin{aligned} \hat{\mathrm{A}} |0_a\rangle |0\rangle_j &= |0_a\rangle |0\rangle_j \\ \hat{\mathrm{A}} |0_a\rangle |1\rangle_j &= |1_a\rangle |0\rangle_j \\ \hat{\mathrm{A}} |0_a\rangle |+\rangle_j &= |+_a\rangle |0\rangle_j . \end{aligned} \tag{12}$$

Thus, $\hat{\mathrm{A}}$ can be used directly when $l_{ij} = x_j$. We modify the above equation using $\hat{\mathrm{X}}$ operator for the case when $l_{ij} = \bar{x}_j$ as:

$$\begin{aligned} \hat{\mathrm{I}} \otimes \hat{\mathrm{X}} \cdot \hat{\mathrm{A}} \cdot \hat{\mathrm{I}} \otimes \hat{\mathrm{X}} |0_a\rangle |0\rangle_j &= |1_a\rangle |1\rangle_j \\ \hat{\mathrm{I}} \otimes \hat{\mathrm{X}} \cdot \hat{\mathrm{A}} \cdot \hat{\mathrm{I}} \otimes \hat{\mathrm{X}} |0_a\rangle |1\rangle_j &= |0_a\rangle |1\rangle_j \\ \hat{\mathrm{I}} \otimes \hat{\mathrm{X}} \cdot \hat{\mathrm{A}} \cdot \hat{\mathrm{I}} \otimes \hat{\mathrm{X}} |0_a\rangle |+\rangle_j &= |+_a\rangle |1\rangle_j , \end{aligned} \tag{13}$$

where $\hat{\mathrm{X}}$ is the Pauli X operator defined as:

$$\hat{\mathrm{X}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} . \tag{14}$$

Note that in all the final states of the above two equations, the ancilla qubit is separable from the data qubit; thus, we can reset and reuse the same ancilla qubit for multiple clauses. We define the extrapolated CX gate in $\hat{\mathrm{A}}$ for $k+1$ qubits with first k qubits as control as CX^{k+1},

$$\mathrm{CX}^{k+1} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & & \vdots & \vdots \\ 0 & 0 & & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 \\ 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix}_{2^{k+1} \cdot 2^{k+1}} \tag{15}$$

We define an operator $\hat{\mathrm{C}}_i$ that would remove the states which do not satisfy the clause C_i as

$$\hat{\mathrm{C}}_i = \bigotimes_{j=1}^{k} \hat{\mathrm{L}}_j \otimes \hat{\mathrm{I}} \cdot \mathrm{CX}^{k+1} \cdot \bigotimes_{j=1}^{k} \hat{\mathrm{L}}_j \otimes \hat{\mathrm{I}} \cdot \hat{\mathrm{I}}^{\otimes \mathrm{t}-1} \otimes \mathrm{CX}_{a,t} \otimes \hat{\mathrm{I}}^{\otimes \mathrm{k}-\mathrm{t}}, \tag{16}$$

where $\hat{\mathrm{L}}_j$ is

$$\hat{\mathrm{L}}_j = \begin{cases} \hat{\mathrm{I}} & \text{if } l_{ij} = x_j \\ \hat{\mathrm{X}} & \text{if } l_{ij} = \bar{x}_j . \end{cases} \tag{17}$$

Here, t denotes the index of the target qubit q_t for the operation $\mathrm{CX}_{a,t}$. We choose in a manner which least affects the states from the other clauses, this is later discussed in detail in algorithm 1. The quantum circuit for the operation $\hat{\mathrm{C}}_i$ is given in Fig. 2. Here we have taken the first three literals with C_i as $\hat{\mathrm{C}}_i = (l_{i1}, l_{i2}, l_{i3})$.

As an example consider C_i, $C_i = (\bar{x}_1, x_2, \bar{x}_3)$. The assignment with values $(x_1 = 1, x_2 = 0, x_3 = 1)$ would not satisfy C_i. Thus, all such states shall be removed by

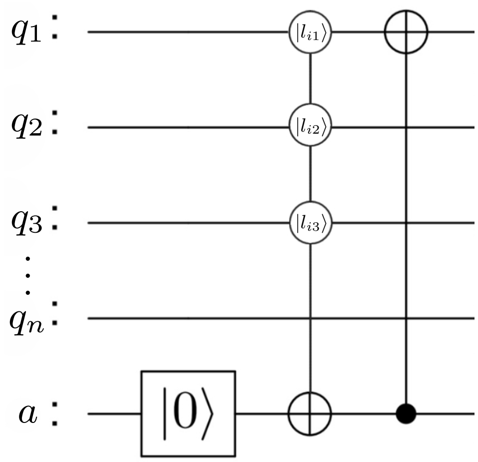

FIG. 2. The quantum circuit for the operation $\hat{\mathrm{C}}_i$. We have taken the first three literals with C_i as $\hat{\mathrm{C}}_i = (l_{i1}, l_{i2}, l_{i3})$. The notation of $|l_{ij}\rangle$ in the circle denotes the control state as per the values of l_{ij}.

the operator $\hat{\mathrm{C}}_i$ as

$$\begin{aligned}
&\hat{\mathrm{C}}_i \,|0_a\rangle\, |\psi\rangle_n \cdots |\psi\rangle_4\, |0\rangle_3\, |0\rangle_2\, |0\rangle_1 \\
&\quad = |0_a\rangle\, |\psi\rangle_n \cdots |\psi\rangle_4\, |0\rangle_3\, |0\rangle_2\, |0\rangle_1 \\
&\qquad\qquad \vdots \\
&\hat{\mathrm{C}}_i \,|0_a\rangle\, |\psi\rangle_n \cdots |\psi\rangle_3\, |1\rangle_2\, |0\rangle_1\, |1\rangle_1 \\
&\quad = |0_a\rangle\, |\psi\rangle_n \cdots |\psi\rangle_4\, |1\rangle_3\, |0\rangle_2\, |0\rangle_1 \\
&\qquad\qquad \vdots \\
&\hat{\mathrm{C}}_i \,|0_a\rangle\, |\psi\rangle_n \cdots |\psi\rangle_4\, |1\rangle_3\, |1\rangle_2\, |1\rangle_1 \\
&\quad = |0_a\rangle\, |\psi\rangle_n \cdots |\psi\rangle_4\, |1\rangle_3\, |1\rangle_2\, |1\rangle_1\,,
\end{aligned} \tag{18}$$

where $|\psi_j\rangle$ represents the state of the data qubit q_j, $j \in \{1, \ldots, n\}$. Here, the states $\bigotimes_{i=4}^{n} |\psi_i\rangle \otimes |1\rangle_3\, |0\rangle_2\, |1\rangle_1$ have changed to $\bigotimes_{i=4}^{n} |\psi_i\rangle \otimes |1\rangle_3\, |0\rangle_2\, |0\rangle_1$, which satisfies the clause C_i. Additionally, all the other states have not been affected. Thus, multiple clauses can be combined in such a manner to solve the boolean satisfiability problem.

Coming to the problem of the choice of the target qubit, if all the clauses have their separate target qubit without interfering with other clauses, we easily reach the perfect solution by applying them all in sequence. However, if we have an overlap in the literals of two clauses, for example, consider C_i and C_r on literals l_{ij} and l_{rj}, where $l_{ij} = \overline{l}_{rj}$. Here, if we choose q_j as the target qubit for clause C_r, we might create some states that would not satisfy clause C_i. These states need to be removed again with the operator $\hat{\mathrm{C}}_i$ using a different target qubit. For this purpose, we propose a conjecture that if there exists a state that satisfies all the clauses, it shall remain unaffected by the operations of $\hat{\mathrm{C}}_i$ for $1 \leq i \leq m$. This conjecture can be proved by extrapolating Eq. (18). Thus, we apply the same clause $\hat{\mathrm{C}}_i$ on multiple target qubits to completely weed out the accidentally created non-solution states again and again. This makes sure we are only left with the solution states.

We propose an additional conjecture that says if there exists a solution state for a boolean problem, we shall always be able to reach it using the proposed operations while weeding out all the undesired solutions. Thus we also suggest that an infinite loop of intersecting clauses would mean no satisfiable assignment to the said boolean problem exists. The combined complexity of the proposed three algorithms below is $\mathcal{O}(m^2 \cdot n^2)$. The worst-case circuit depth of the quantum circuit generated comes out to be $\mathcal{O}(m/2(m+1)) = \mathcal{O}(m^2)$. Here, we count each $\hat{\mathrm{C}}_i$ and respective CX operation as a single unit of circuit depth.

Algorithm 1 $k-$SAT Solution

```
m ← number of clauses.
n ← total number of literals.
Create an equal superposition state |ψ⟩ of all n qubits by
  H^{⊗n} operation.
      |ψ⟩ = H^{⊗n} |0⟩^{⊗n}
for each integer i following 1 ≤ i ≤ m do
    ancilla a ← |0⟩ {Reset the ancilla qubit}
    𝒬 ← []
    for all literals l_ij in C_i do
        Q ← []
        𝒬 ← 𝒬 + [FIND(C_i, l_ij, ℬ, Q, n)][0] {finds intersec-
          tions with other clauses which intersect with C_i.}
    end for
    Q ← min_len(𝒬) {minimum length Q in 𝒬; Q contains
      both q_t and the clauses.}
    for Clauses C_r and target qubits q_t in Q do
        Apply Ĉ_r on k, a qubits.
        Apply CX_a,t gate to remove the undesired states.
    end for
end for
Measure all n qubits. {measure the states satisfying the
  boolean formula.}
```

Algorithm 2 FIND

Require: $C_i, l_{ij}, \mathcal{B}, Q, n, \mathcal{L}_i = [], \mathcal{C} = [C]$
Ensure: Updated Q, $\mathcal{C}$

```
if ℒ_i = [] then
    ℒ_i ← [l_ij]
end if
if |ℒ_i| > n then
    return Q, 𝒞
end if
for C_r ∈ ℬ do
    for l_rs in C_r do
        if l_rs = l̄_ij then
            q_t ← Intersection(C_r, ℒ_i, 𝒞)
            if q_t = null then
                return No Solution
            end if
            if [q_t, C_r] ∉ Q then
                𝒞 ← 𝒞 ∪ {C_r}
                Q ← Q + [q_t, C_r]
                if l_rt ∉ ℒ_i then
                    ℒ_i ← ℒ_i ∪ {l_rt}
                    Q, 𝒞 ← Find(C_r, l_rt, ℬ, Q, n, ℒ_i, 𝒞)
                end if
            end if
        end if
    end if
end for
return Q, 𝒞
```

Algorithm 3 Intersection

Require: C_r, $\mathcal{L}_i$, $\mathcal{C}$

```
for l_rj ∈ C_r do
    if l_rj ∈ ℒ_i ∧ C_r ∉ 𝒞 then
        return q_j
    end if
end for
if C_r ∈ 𝒞 then
    for l_rj ∈ C_r do
        if l_rj has not been applied to qubit q_j then
            return q_j
        end if
    end for
end if
for l_rj ∈ C_r do
    if q_j ∉ |ℒ_i| then
        return q_j
    end if
end for
return null {No assignment satisfies}
```


[1] L. K. Grover, A fast quantum mechanical algorithm for database search, in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (1996) pp. 212–219, arXiv:quant-ph/9605043.

[2] A. M. Childs, A. J. Landahl, and P. A. Parrilo, Quantum algorithms for the ordered search problem via semidefinite programming, Physical Review A **75**, 032335 (2007).

[3] S. Arunachalam, *Satisfiability and derivative-free optimization*, Master's thesis, University of Waterloo (2014).

[4] S. Boulebnane, A. Montanaro, and A. Lucas, Solving k-sat problems with generalized quantum measurement, arXiv preprint arXiv:2406.13611 (2024).

[5] R. Zhang *et al.*, Procedure of solving 3-sat problem by combining quantum search algorithm and dpll algorithm, in *Quantum Computing: Algorithms and Applications* (Clausius Scientific Press, 2020) pp. 45–62.
[6] S. Boulebnane *et al.*, Quadratic speedup for unstructured search with bounded noise, PRX Quantum **5**, 030348 (2024).
[7] Y. Prabhat, S. Thakur, and A. Raina, Structured quantum search algorithm: A quantum leap, arXiv preprint 2504.03426 (2025).