A Grover Search-based Algorithm for the List Coloring Problem

Sayan Mukherjee*

August 19, 2021

Abstract

Graph coloring is a computationally difficult problem, and currently the best known classical algorithm for k-coloring of graphs on n vertices has runtimes $\Omega(2^n)$ for $k \geq 5$. The list coloring problem asks the following more general question: given a *list* of available colors for each vertex in a graph, does it admit a proper coloring? We propose a quantum algorithm based on Grover search [11] to quadratically speed up exhaustive search. Our algorithm loses in complexity to classical ones in specific restricted cases, but improves exhaustive search for cases where the lists and graphs considered are arbitrary in nature.

1 Introduction

Graph coloring problems provide for a rich family of NP-complete problems in theoretical computer science. While exhaustive search is believed to be the fastest classical approach for several NP-complete problems including satisfiability and hitting-set [7], there are much better classical algorithms using dynamic programming, inclusion-exclusion and other structural approaches for problems such as graph coloring [9, 3, 17], the traveling salesman problem [22, 12], set cover [14] etc. Several authors have obtained quantum speedup on these classical algorithms [2, 26, 29]; however, all of these algorithms have the limitation that they cannot be easily generalized to the list coloring problem.

Given a finite graph $G = (V, E)$, a proper coloring of G is a function $\chi : V \to \mathbb{N}$ such that for every edge $uv \in E$, $\chi(u) \neq \chi(v)$. The list coloring problem tries to determine a proper coloring χ of a graph $G = (V, E)$, given a list L_v of available colors for each vertex v. In other words, it is forced that $\chi(v) \in L_v$. When $L_v = \{1, 2, \ldots, k\}$ for every vertex v this reduces to the well-studied k-coloring problem. We propose a simple Grover search-based approach to obtain a quadratic speedup on exhaustive search for the list coloring problem.

Grover's algorithm [11] is known to speed up unstructured search quadratically using the technique of *amplitude amplification*. In its simplest form, to find some marked elements from a list of $N = 2^n$ entries, the algorithm starts with a uniform quantum superposition of all 2^n basis states of an n-qubit register. It then amplifies the amplitudes of the searched state and reduces those of the other states, such that a measurement of the n qubits leads to one of the searched states with high probability.

Grover's algorithm has been used to obtain quantum speedups for various problems in combinatorial optimization and computer science (see, for e.g., [18, 31, 16, 15, 20]). Needless to say, graph coloring problems are also not an exception in the literature, and have been attacked using quantum annealing [33, 19], hybrid approaches [32, 5], as well as using Grover search [36, 30, 28].

In [36], a qutrit-based approach has been used to demonstrate the cost-efficiency of ternary quantum logic; however, their main algorithm is not realizable right now on NISQ devices. The algorithm of [30] has the

*blueqat Co. Ltd., Shibuya Scramble Square 39F, Shibuya 2-24-12, Tokyo 150-0002, Japan. Email:sayan@blueqat.com.

same issue as it requires quantum RAM which has not been realized at this moment. On the other hand, the authors of [28] and [27] demonstrate a quantum algorithm solving the k-coloring problem on NISQ devices, comparing the efficiency of their algorithm against the reduction of 3-SAT to 3-coloring approach of Hu et. al. [13].

All of these algorithms use an oracle design which uses binary comparators, and provide solutions where almost all binary strings have positive probabilities of being selected, including those that do not represent valid colorings. Our approach circumvents this problem via a modified initialization and diffusion operator that restricts the evolution of the quantum algorithm to the only $\prod_{v \in L_v} |L_v|$ plausible states. Note that this is the total number of valid colorings when the underlying graph is empty. We achieve this via the restricted version of Grover search [11, 10].

Proposition 1 (Restricted Grover search)**.** *Let $S \subseteq \{1, 2, \ldots, 2^n - 1\}$, and suppose $S' \subsetneq S$ is a set of marked states. Let O be an oracle that marks these states and requires a ancillas. Then, there is a quantum circuit on $n + a + 1$ qubits which makes $O(\sqrt{2^n/|S'|})$ queries, which when measured, gives one of the marked states with high probability. Further, states outside S are never measured.*

Additionally, we use an oracle design different from those in [36, 28], and give a classical algorithm in Section 3 that can reduce the complexity of this oracle in several special cases (such as for the 3-coloring or 4-coloring problems). As a corollary of Proposition 1, our main theorem provides an algorithm for the list coloring problem.

Theorem 2 (Quantum list coloring algorithm)**.** *Given a graph $G = (V, E)$ on n vertices and m edges and lists of available colors $\{L_v : v \in V\}$, there exists a $(\sum_{v \in V} \lceil \log_2 \max L_v \rceil + m + 1)$-qubit quantum algorithm with query complexity $O(\prod_{v \in V} (\max L_v)^{1/2})$ that returns a valid list coloring of G with high probability.*

This paper is organized as follows. In Section 2, we describe Grover's algorithm and a gate-level implementation. Section 3 is devoted to tackling the list coloring problem, and proves Theorem 2. In Section 4, we run experiments on classical simulators as well as real quantum machines, and compare the outcomes. We discuss applications and provide concluding remarks in Section 5.

2 Grover's Algorithm

In this section, we provide a concise exposition on Grover search. The main idea behind Grover search is to amplify the amplitudes of some number of marked states (states which are being searched for), and consequentially decrease that of unmarked states. Grover's Algorithm requires three different operators: Initialization, Oracle, and Diffusion. Below we present two formulations of the algorithm:

2.1 Unrestricted search space

When searching for a marked state among the full search space $S = \{0, 1\}^n$, the initialization step of the algorithm creates a uniform superposition of all the possible states of an n-qubit system. This is achieved via appending Hadamard gates on each qubit:

$$H^{\otimes n}|0\rangle_n = |+\rangle^n = \frac{1}{\sqrt{2^n}} \sum_{i=1}^{n} |i\rangle_n.$$

Here we abuse notation and write $|i\rangle_n$ to denote the state corresponding to an n-digit binary representation of i.

Next, Grover's Algorithm requires an oracle O that, given a uniform superposition of all 2^n possible states, can change the sign of the marked states. Let $S' \subseteq \{0, 1, \dots 2^n - 1\}$ be a set of marked states. The Oracle O then switches the signs of the states in S', i.e.

$$O|i\rangle_n = \begin{cases} |i\rangle_n, & i \notin S \\ -|i\rangle_n, & i \in S. \end{cases}$$

The circuit implementation of the oracle O usually is the most difficult (and computationally expensive) part of the algorithm, and one of the most basic implementations requires the usage of phase kickback [6].

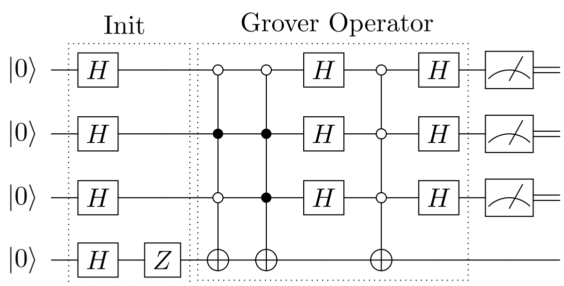
The final component of Grover's Algorithm is the diffusion operator D, which can be thought of as a reflection around the vector $|0\rangle^n$. As an operator, we have

$$D = 2|0\rangle_n\langle 0|_n - I.$$

D is usually implemented using phase kickback in the same fashion as the oracle O.

Grover's algorithm requires repeated usage of the operator $G = H^{\otimes n} D H^{\otimes n} O$ which has the net effect of reflecting around $|+\rangle^n$, amplifying the amplitudes of marked states and decreases those of other states. Measuring the state $G^r H^{\otimes n}|0\rangle_n$ ($r \geq 1$) gives one of the marked states with high probability, and this probability is maximum when $r = \lfloor \frac{\pi}{4}\sqrt{2^n/|S'|} \rfloor$. Since $|S'|$ is not known in general, the r is either randomly selected [8, 4], or is estimated using quantum counting algorithms [23, 1].

See Figure 2.1 for an example of a circuit implementing unrestricted Grover search with $n = 3$, $S' = \{|010\rangle_n, |011\rangle_n\}$, $S = \{0, 1, \dots, 7\}$.



In a standard circuit implementation of Grover search, the initialization is achieved by the Hadamard operator $H^{\otimes 3}$. Phase kickback from the fourth qubit initialized to the state $|-\rangle$ is used to negate the amplitudes of $|010\rangle$ and $|011\rangle$. Finally, diffusion is achieved via another phase kickback from the same qubit.

Figure 2.1: An example Grover search implementation

2.2 Restricted search space

Let us now consider a search space $S \subsetneq \{0, 1\}^n$. In this case, the algorithm is designed to only evolve over the states of S, and this is achieved via an initialization operator A such that

$$A|0\rangle_n = \frac{1}{\sqrt{|S|}} \sum_{i \in S} |i\rangle_n,$$

And the Grover operator is changed to a reflection around $A|0\rangle_n$ instead of $|+\rangle^n$:

$$G = ADA^\dagger O.$$

The usage of $ADA^\dagger$ instead of $H^{\otimes n} D H^{\otimes n}$ makes sure that the evolution of the quantum states remains in the subspace spanned by S instead of the entire space $\{0, 1\}^n$, and this leads to probability distribution of the measured outcomes being supported on the state S.

The only detail missing in this formulation is the construction of the initialization operator A. As we shall see in Section 3, for graph coloring problems (and most applications in general), A can be represented as a block matrix and can be implemented in time linear in the number of qubits.

We make a remark here that the graph coloring algorithm of [28] uses the unrestricted formulation of Grover's algorithm, but modifies the oracle to discard states that represent invalid colorings. On the other hand, they do not modify their diffusion operator, leading to states outside the search space having positive probabilities of being measured.

3 Quantum List Coloring Algorithm

Our goal in this section is to prove Theorem 2. For the remainder of this section, assume that $G = (V, E)$ is an arbitrary graph with $|V| = n$, $|E| = m$. Further, for every vertex v, let L_v denote the list of admissible colors for vertex v. As our algorithm is based on Grover search, we shall discuss the four basic steps of the algorithm: circuit setup, initialization, oracle and diffusion, each in their respective subsections.

3.1 Setup and qubit labels

Our algorithm design requires three different qubit registers:

- A *vertex register* to keep track of vertex colors. For each vertex v, we require the usage of $\lceil \log_2 \max L_v \rceil$ qubits to represent each color in L_v. Let us denote $j_v := \lceil \log_2 \max L_v \rceil$, and let the qubits corresponding to vertex $v \in V$ be labeled by $q_v^1, \ldots, q_v^{j_v}$.
- An *edge register* consisting of m qubits, one corresponding to each edge. Let q'_{uv} denote the qubit corresponding to an edge $uv \in E$.
- A single qubit *ancilla register*, used for phase kickback in Grover's algorithm. Let q^* denote this ancilla.

The total number of qubits required is $\sum\limits_{v \in V} j_v + m + 1$. Now we take a closer look at our list coloring algorithm.

3.2 Initialization

For each $v \in V$, we initialize qubits $q_v^1, \ldots, q_v^{j_v}$ to a uniform superposition $\chi_v = \frac{1}{\sqrt{|L_v|}} \sum_{i \in L_v} |i\rangle$. This can be achieved via a unitary operator U_v such that

$$U_v |0\rangle = \frac{1}{\sqrt{|L_v|}} \sum_{i \in L_v} |i\rangle.$$

We show one way of constructing the operator U_v. First, consider the standard basis $B = \{|i\rangle : i \in \{0, 1, \ldots, 2^{v_j} - 1\}\}$. We shall replace any one entry $|i\rangle$ with χ_v, where $i \in L_v$: let $B' = \{\chi_v\} \bigcup (B \setminus \{|i\rangle\})$. It can be seen that $\mathrm{Span}(B) = \mathrm{Span}(B')$. We can now consider B' as an ordered basis with its first entry as χ_v, and apply the Gram-Schmidt process to turn B' into an orthonormal basis B'' [25]. Note that as $\|\chi_v\| = 1$, it remains unchanged in B''. The transpose of the coefficients of the vectors in B'' constitutes a change of basis operator that maps $|0\rangle$ to χ_v, and this is how one can construct the matrix for U_v. It can then be implemented with quantum gates using the results of [21, 24, 34] for example.

For the sake of clarity of the above procedure, let us consider an example with $j_v = 2$ and $L_v = \{1, 2, 3\}$. Note that $\chi_v = \frac{1}{\sqrt{3}}(|1\rangle + |2\rangle + |3\rangle)$. Then, $B = \{|0\rangle, |1\rangle, |2\rangle, |3\rangle\}$, and we can take $B' = \{\chi_v, |0\rangle, |1\rangle, |2\rangle\}$. After the Gram-Schmidt process, we obtain

$$B'' = \left\{ \tfrac{1}{\sqrt{3}}|1\rangle + \tfrac{1}{\sqrt{3}}|2\rangle + \tfrac{1}{\sqrt{3}}|3\rangle,\ |0\rangle,\ \tfrac{\sqrt{2}}{\sqrt{3}}|1\rangle - \tfrac{1}{\sqrt{6}}|2\rangle - \tfrac{1}{\sqrt{6}}|3\rangle,\ \tfrac{1}{\sqrt{2}}|2\rangle + \tfrac{1}{\sqrt{2}}|3\rangle \right\}.$$

Hence, in this case, we get

$$U_v = \begin{bmatrix} 0 & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ 1 & 0 & 0 & 0 \\ 0 & \frac{\sqrt{2}}{\sqrt{3}} & -\frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{6}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}^\top .$$

Observe that by construction, $U_v|0\rangle = U_v \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^\top = \begin{bmatrix} 0 & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{bmatrix}^\top$, as desired.

Finally, let us denote $A = \bigotimes_{v\in V} U_v$. Then,

$$A \otimes \bigotimes_{e\in E} I \otimes ZH \tag{3.1}$$

is the full initialization operator applied to the circuit starting from $|0\rangle_{\sum_{v\in V} j_v} \otimes |0\rangle_m \otimes |0\rangle$. This creates the quantum state $\bigotimes_{v\in V} \chi_v \otimes |0\rangle_m \otimes |-\rangle$.

3.3 Oracle

Traditionally for the graph coloring problem, each vertex color is represented using the same number of qubits, and binary comparator circuits [36] are used to make sure that the two colors corresponding to two adjacent vertices are different.

While this approach is very efficient for the k-coloring problem where every vertex has the same set of admissible colors, the list coloring problem may sometimes require a large number of qubits. In fact, the total number of qubits required for implementing a comparator-based oracle would be $n \cdot \max_{v\in V} j_v + m + 1$, which can be much higher than our proposed oracle when the j_v's are not all equal.

In short, for every edge $uv \in E$, we shall encode all possible colorings in $L_u \times L_v$ via flipping the amplitudes of the states corresponding to valid colorings $\{|i_1\rangle_{j_u}|i_2\rangle_{j_v} : i_1 \in L_u, i_2 \in L_v, i_1 \neq i_2\}$. We propose a classical $O(|L_u|^2|L_v|^2)$-time algorithm to construct an efficient oracle $O_{u,v}$ for flipping these amplitudes. In short, $O_{u,v}$ should have the following net effect:

$$O_{u,v}\left(\frac{1}{\sqrt{|L_u||L_v|}} \sum_{i_1\in L_u, i_2\in L_v} |i_1\rangle|i_2\rangle|-\rangle\right) = \frac{1}{\sqrt{|L_u||L_v|}}\left(\sum_{\substack{i_1\in L_u, i_2\in L_v\\ i_1\neq i_2}} -|i_1\rangle|i_2\rangle + \sum_{i\in L_u\cap L_v} |i\rangle|i\rangle\right)|-\rangle \tag{3.2}$$

Given a string s of length ℓ and a subset $J = \{j_1, j_2, \ldots, j_r\} \subseteq \{1, 2, \ldots, \ell\}$ we use s_J to denote the substring $s_{j_1}s_{j_2}\ldots s_{j_r}$. $\mathbb{F}_2$ denotes the finite field of two elements. Our algorithm for implementing $O_{u,v}$ (Algorithm 1) makes use of a subroutine called oracleReduction that can significantly simplify the complexity and the number of controlled not operations required in many cases.

Algorithm 1: Oracle $O_{u,v}$ for marking valid colorings.

Input : Sets L_u, L_v (denote $j_u = \lceil\log_2 \max L_u\rceil$, $j_v = \lceil\log_2 \max L_v\rceil$).
Output: A $j_u + j_v + 1$-qubit quantum circuit $O_{u,v}$ satisfying (3.2).

Let $W' = \text{oracleReduction}(L_u, L_v)$;
Create a circuit C with quantum wires $q_1, \ldots, q_{j_u+j_v+1}$;
for *every pair* (J, s) *in* W' **do**
  Add a multicontrolled NOT gate to C with controls on wires $\{q_j : j \in J, s_j = 1\}$, anticontrols on wires $\{q_j : j \in J, s_j = 0\}$ and target $q_{j_u+j_v+1}$.
return C

Function oracleReduction(L_u, L_v)

Input : Sets L_u, L_v.
Output: A set of pairs W'.

Denote $j_u = \lceil \log_2 \max L_u \rceil$, $j_v = \lceil \log_2 \max L_v \rceil$;
Convert each entry of L_u and L_v into $\{0,1\}$-strings of lengths j_u and j_v, respectively;
Let $X = \{ab : a \in L_u, b \in L_v, a \neq b\}$, where "ab" denotes concatenation;
Let $Y = \{ab : a \in L_u.b \in L_v\}$, then $|Y| = |L_u||L_v|$;
Set $W = \varnothing$;
for $k = 1$ **to** $j_u + j_v$ **do**
  for *every k-element subset J of* $\{1, 2, \ldots, j_u + j_v\}$ **do**
    for *every* $\{0,1\}$*-string s of length k* **do**
      Add a $\{0,1\}$-variable x_J^s to W;
  Create an empty linear system of equations $\mathcal{L}$ over $\mathbb{F}_2$ with variables W;
  for *every* $\{0,1\}$*-string* $t \in Y$ **do**
    Calculate the expression $f(W,t) = \sum\{x_J^s \in W : t_J = s\}$;
    if $t \in X$ **then**
      Add $f(W,t) = 1$ to $\mathcal{L}$;
    else
      Add $f(W,t) = 0$ to $\mathcal{L}$;
  Solve the $|L_u||L_v| \times \sum_{j=1}^{k} \binom{j_u+j_v}{j} \cdot 2^j$ system $\mathcal{L}$ using Gaussian Elimination over $\mathbb{F}_2$;
  if *the system* $\mathcal{L}$ *is solvable* **then**
    Solve the linear program minimizing $\sum_{x_J^s \in W} |J| \cdot x_J^s$ subject to $\mathcal{L}$;
    Let $W' = \{(J,s) : x_J^s \in W, x_J^s = 1\}$;
    return W'

We shall now demonstrate the correctness of Algorithm 1.

Theorem 3. *Algorithm 1 gives a circuit C satisfying (3.2). Further, if the cost of implementing a k-controlled NOT operation is k, then the cost of C is the smallest among all circuits that can be made using only controlled NOT operations onto the phase flip qubit.*

Proof. It is sufficient to verify the action of C on the states $|i_1\rangle|i_2\rangle|-\rangle$, where $i_1 \in L_u$ and $i_2 \in L_v$. Notice that a single controlled NOT gate corresponding to a pair (J,s) in C effectively flips the amplitudes of all basis states represented by $\{0,1\}$-strings x of length $j_u + j_v$ for which $x_i = s_i$, $i \in J$. In other words, *all* states of the following form are flipped (here $*$ denotes a wildcard, and $J = \{j_1, \ldots, j_k\}$, $s = s_1 \cdots s_k$):

$$|* \cdots * \underset{j_1\text{'th}}{s_1} * \cdots * \underset{j_2\text{'th}}{s_2} \underset{\cdots}{\cdots\cdots\cdots} \underset{j_k\text{'th}}{s_k} * \cdots *\rangle$$

Thus, after application of all the controlled NOT gates, only states which appeared in an odd number of (J,s)-pairs in W' will survive, and those appearing an even number of times will not have their amplitudes flipped.

Let us now fix a string $t = t_1 t_2 \cdots t_{j_u+j_v}$, where $t_1 \cdots t_{j_u} \in L_u$ and $t_{j_u+1} \cdots t_{j_u+j_v} \in L_v$, and analyze the function oracleReduction closely. Recall that $Y = \{ab : a \in L_u, b \in L_v\}$ and $X = \{ab : a \in L_u, b \in L_v, a \neq b\}$. We now make a crucial observation: the number of times the amplitude of $|t\rangle$ gets flipped by C is

$$|\{(J,s) \in W' : t_J = s\}| = |\{x_J^s \in W : x_J^s = 1, t_J = s\}| = f(W,t),$$

As x_J^s are all $\{0,1\}$-valued. Since the linear system $\mathcal{L}$ over $\mathbb{F}_2$ exactly contains the equations $f(W,t) = 1$ if $t \in X$ and $f(W,t) = 0$ if $t \in Y \setminus X$, any solution to the system $\mathcal{L}$ will give a correct circuit satisfying (3.2). ∎

Now we come to the second assertion of the theorem. Note that for each variable $x^s_J \in W$ with $x^s_J = 1$, we require a $|J|$-controlled NOT operation onto the phase flip ancilla. This means that the cost of C is exactly $\sum_{x^s_J \in W} |J| \cdot x^s_J$. As we have minimized this cost via a linear program in Step 19 of oracleReduction, this proves the second claim.

□

We now have all the ingredients required to implement our full oracle, which is presented as Algorithm 3 below.

Algorithm 2: Full oracle O for marking valid list colorings

Input : Graph $G = (V, E)$, lists $\{L_v : v \in V\}$, and a $\sum_{v \in V} j_v + m + 1$-qubit quantum circuit C.
Output: C appended with a graph coloring oracle O.

for *every edge* $uv \in E$ **do**
  Apply Oracle $O_{u,v}$ on qubits $q^1_u, \ldots, q^{j_u}_u$, $q^1_v, \ldots, q^{j_v}_v$, q'_{uv} of C;
Apply a controlled NOT operation with controls q'_* and target q^*;
for *every edge* $uv \in E$ **do**
  Apply Oracle $O_{u,v}$ on qubits $q^1_u, \ldots, q^{j_u}_u$, $q^1_v, \ldots, q^{j_v}_v$, q'_{uv} to de-entangle the vertex register of C;
return C

3.4 Diffusion

Our diffusion operator is very straightforward, and is a direct application of the restricted search space diffusion mentioned in section 2.2. Let A be the initialization operator we implemented in (3.1), then a diffusion is achieved by $ADA^\dagger$, where $D = 2|0\rangle\langle 0| - I$ can be implemented by a controlled NOT with anticontrols on each of the vertex qubits and target the phase flip ancilla.

Refer to Figure 3.1 for an illustration of the full list coloring algorithm.

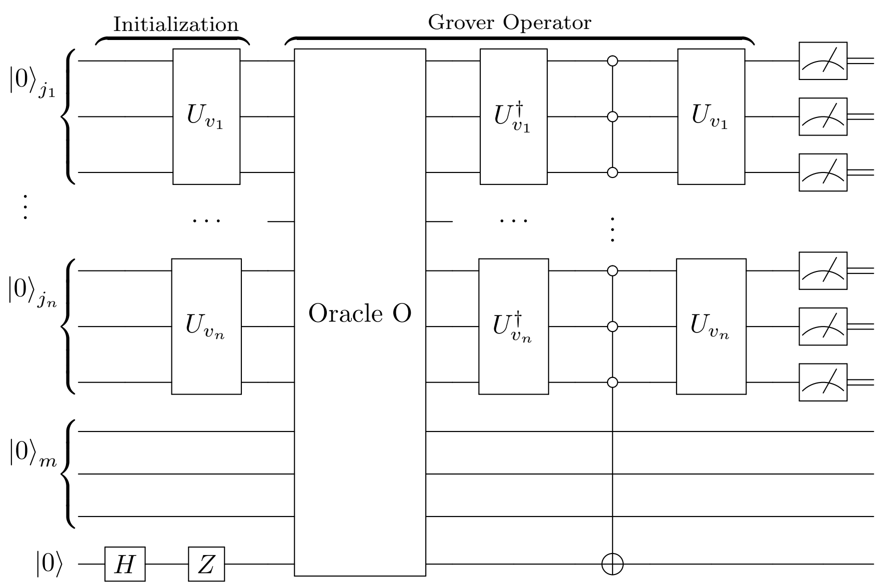


Figure 3.1: Outline of our list coloring circuit for a single Grover iteration.

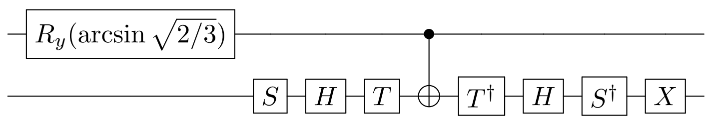
4 Results

We implement our list coloring algorithm in `python 3.8` using `blueqat-sdk`. In order to gauge the efficiency of our result, we run experiments for the 3 and 4-coloring problems used in [28] on the the Amazon Statevector simulator.

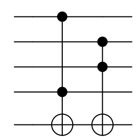
4.1 3-coloring K_3

3-coloring the triangle graph $G = (\{1,2,3\}, \{12, 23, 13\})$ is equivalent to the list-coloring problem on G with $L_1 = L_2 = L_3 = \{|01\rangle, |10\rangle, |11\rangle\}$. In this case, our initialization operator U_v and oracle component O_{uv} obtained from Algorithm 1 are:

U_v =

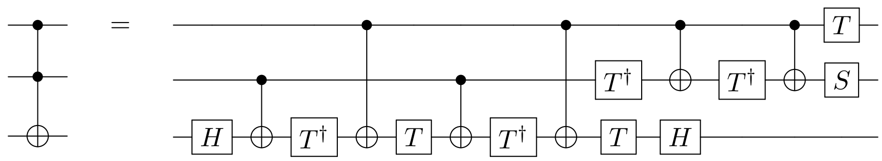


; O_{uv} =



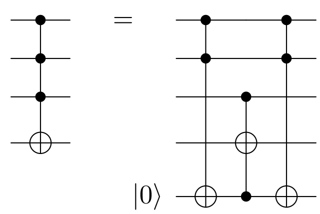
.

Each Toffoli gate can be decomposed into two-qubit gates using the standard decomposition:



.

To implement a `cccx` gate, we use a clean ancilla qubit to reduce circuit depth as demonstrated below:



Finally, we are able to run our circuit on the Amazon Statevector Simulator after decomposing into these elementary single and two-qubit operations. Our resulting histogram is shown in Figure 4.1.

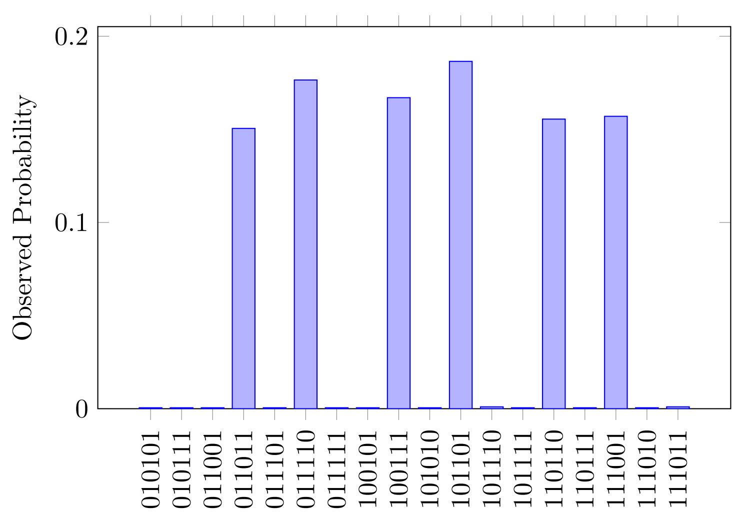


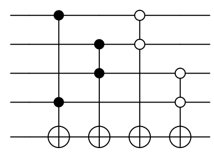
Figure 4.1: Observed frequencies for 2000 shots for 3-coloring K_3.

It is seen that the states $|011011\rangle, |011110\rangle, |100111\rangle, |101101\rangle, |110110\rangle, |111001\rangle$ are all massively amplified. A theoretical calculation of the statevector shows that each of these states have a probability of 0.165066 of being measured, whereas states representing colorings outside $\{|01\rangle, |10\rangle, |11\rangle\}^3$ have probability 0 of being measured. After a single Grover iteration, we therefore improve upon the results of [28].

4.2 4-coloring K_4

Let $G = \{\{1, 2, 3, 4\}, \{12, 13, 14, 23, 24, 34, 14\}$ be the complete graph on 4 vertices, and suppose $L_1 = L_2 = L_3 = L_4 = \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. In this case the simple Hadamard operator $H^{\otimes 2}$ initializes each vertex register to a uniform superposition of its valid colors, and we can then run Algorithm 1 to figure out the component O_{uv}. It turns out that one of the valid solutions minimizing the cost of gates used is the following circuit:

O_{uv} =



.

Figure 4.2 shows the results of running our circuit on the statevector simulator.

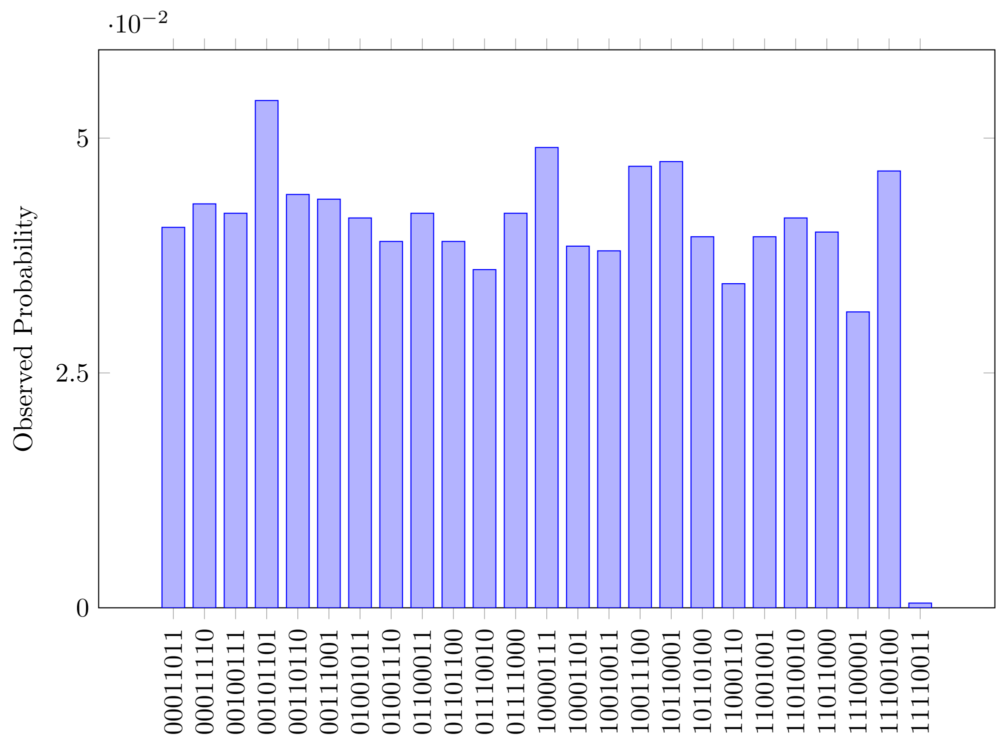


Figure 4.2: Observed frequencies for 2000 shots for 4-coloring K_4.

In this case, a statevector computation shows that each of the $4! = 24$ states corresponding to valid colorings of K_4 has a theoretical probability of 0.041657 of being measured, whereas the same for every unmarked state is approximately 10^{-6}. The result of our experiment also follows this distribution.

We also ran our circuits on the IonQ physical device, however, reasonable results were not obtained due to the limitations of current quantum computers.

5 Applications and Concluding Remarks

The list coloring problem is ubiquitous in real life, as it not only generalizes an already well-appearing problem of graph coloring (scheduling, satisfiability etc.), but is also applicable to several other scenarios, such as:

- Wireless network Allocation [35]: Consider a wireless network. Assume that due to hardware restrictions, each radio in the network has a limited set of frequencies through which it can communicate. Also assume radios within a certain distance of each other cannot operate on the same frequency without their transmissions interfering with each other. This problem can be modeled in terms of list-coloring by representing the radios by vertices, and joining them by edges if their corresponding radios are within a certain distance of each other. The lists for each vertex can be assigned according to the available frequencies for its corresponding radio.
- Register Allocation: In compiler optimization, register allocation is the process of assigning a large number of target program variables (n) onto a small number of CPU registers (k), which reduces to a k-coloring problem on an n-vertex graph.
- Sudoku: We can represent every cell in a sudoku problem with a vertex, and join two vertices with an edge if they are in same row or same column or same block. Given x already filled cells, we can formulate the sudoku problem as a list-coloring problem on $81 - x$ vertices and at most 9 colors.

We proposed a Grover search-based quantum algorithm that achieves quadratic speedup in query complexity compared to a classical brute force search, and also proposed a classical algorithm that can simplify the oracle design for several special instances of the list coloring problem. We demonstrate the efficiency of our method in comparison with previous work by running our algorithm on the Amazon statevector simulator for the 3 and 4-coloring problems.

Unfortunately, the list coloring problem is difficult to solve both classically and using quantum algorithms, as for generic lists with no known structure, brute force seems to be the only way to attack the problem. As our algorithm is basically a brute force quantum search with some optimizations in the oracle, we perform better in general cases where the structure of the lists are unknown. However, the existence of clever hybrid algorithms exploiting specific structures for known lists cannot be ignored, and is a very promising future direction.

Finally, we note that one can obtain improvements on our algorithm by just changing a given list coloring problem to a reduced problem. For example, if $G = (V, E)$ has a vertex v with $|L_v| = 1$, we can color v first and remove its color from each L_u such that $uv \in E$. Secondly, we can remap admissible colors to ensure that the lists are as compactly represented as possible, leading to lower qubit requirement for our algorithm.

Acknowledgments. Part of this work was completed during a summer internship at Elyah Co. in 2020. The author is grateful to Sydney Andrews, Yūichirō Minato, Ryūtarō Nagai and Goutham Rajendran for several helpful discussions and comments.

References

[1] Scott Aaronson and Patrick Rall. Quantum approximate counting, simplified. In *Symposium on Simplicity in Algorithms*, pages 24–32. SIAM, 2020.

[2] Andris Ambainis, Kaspars Balodis, Jānis Iraids, Martins Kokainis, Krišjānis Prūsis, and Jevgēnijs Vihrovs. Quantum speedups for exponential-time dynamic programming algorithms. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1783–1793. SIAM, 2019.

[3] Hans L Bodlaender and Dieter Kratsch. An exact algorithm for graph coloring with polynomial memory. *UU-CS*, 2006, 2006.

[4] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik: Progress of Physics*, 46(4-5):493–505, 1998.

[5] Sergey Bravyi, Alexander Kliesch, Robert Koenig, and Eugene Tang. Hybrid quantum-classical algorithms for approximate graph coloring. *arXiv preprint arXiv:2011.13420*, 2020.

[6] Richard Cleve, Artur Ekert, Chiara Macchiavello, and Michele Mosca. Quantum algorithms revisited. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969):339–354, 1998.

[7] Marek Cygan, Holger Dell, Daniel Lokshtanov, Dániel Marx, Jesper Nederlof, Yoshio Okamoto, Ramamohan Paturi, Saket Saurabh, and Magnus Wahlström. On problems as hard as cnf-sat. *ACM Transactions on Algorithms (TALG)*, 12(3):1–24, 2016.

[8] Christoph Durr and Peter Hoyer. A quantum algorithm for finding the minimum. *arXiv preprint quant-ph/9607014*, 1996.

[9] David Eppstein. Small maximal independent sets and faster exact graph coloring. In *Workshop on Algorithms and Data Structures*, pages 462–470. Springer, 2001.

[10] Austin Gilliam, Stefan Woerner, and Constantin Gonciulea. Grover adaptive search for constrained polynomial binary optimization. *Quantum*, 5:428, 2021.

[11] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[12] Karla L Hoffman, Manfred Padberg, Giovanni Rinaldi, et al. Traveling salesman problem. *Encyclopedia of operations research and management science*, 1:1573–1578, 2013.

[13] Shaohan Hu, Peng Liu, ChunFu Richard Chen, Marco Pistoia, and Jay Gambetta. Reduction-based problem mapping for quantum computing. *Computer*, 52(6):47–57, 2019.

[14] Qiang-Sheng Hua, Yuexuan Wang, Dongxiao Yu, and Francis CM Lau. Dynamic programming based algorithms for set multicover and multiset multicover problems. *Theoretical Computer Science*, 411(26-28):2467–2474, 2010.

[15] Stacey Jeffery, Robin Kothari, François Le Gall, and Frédéric Magniez. Improving quantum query complexity of boolean matrix multiplication using graph collision. *Algorithmica*, 76(1):1–16, 2016.

[16] Kamil Khadiev and Artem Ilikaev. Quantum algorithms for the most frequently string search, intersection of two string sequences and sorting of strings problems. In *Theory and practice of natural computing*, volume 11934 of *Lecture Notes in Comput. Sci.*, pages 234–245. Springer, Cham, [2019] ©2019.

[17] Mikko Koivisto. An o*(2ˆ n) algorithm for graph coloring and other partitioning problems via inclusion–exclusion. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 583–590. IEEE, 2006.

[18] Dmitry Kravchenko, Kamil Khadiev, and Danil Serov. On the quantum and classical complexity of solving subtraction games. In *Computer science—theory and applications*, volume 11532 of *Lecture Notes in Comput. Sci.*, pages 228–236. Springer, Cham, 2019.

[19] Kazue Kudo. Constrained quantum annealing of graph coloring. *Physical Review A*, 98(2):022301, 2018.

[20] Troy Lee, Frédéric Magniez, and Miklos Santha. Improved quantum query algorithms for triangle detection and associativity testing. *Algorithmica*, 77(2):459–486, 2017.

[21] Chi-Kwong Li, Rebecca Roberts, and Xiaoyan Yin. Decomposition of unitary matrices and quantum gates. *International Journal of Quantum Information*, 11(01):1350015, 2013.

[22] John DC Little, Katta G Murty, Dura W Sweeney, and Caroline Karel. An algorithm for the traveling salesman problem. *Operations research*, 11(6):972–989, 1963.

[23] Michele Mosca et al. Quantum searching, counting and amplitude amplification by eigenvector analysis. In *MFCS98 workshop on Randomized Algorithms*, pages 90–100, 1998.

[24] Mikko Möttönen, Juha J Vartiainen, Ville Bergholm, and Martti M Salomaa. Quantum circuits for general multiqubit gates. *Physical review letters*, 93(13):130502, 2004.

[25] Lyle Pursell and S. Y. Trimble. Gram-schmidt orthogonalization by gauss elimination. *The American Mathematical Monthly*, 98(6):544–549, 1991.

[26] Pooya Ronagh. Quantum algorithms for solving dynamic programming problems. *arXiv preprint arXiv:1906.02229*, 2019.

[27] Amit Saha, Avik Chongder, Sudhindu Bikash Mandal, and Amlan Chakrabarti. Synthesis of vertex coloring problem using grover's algorithm. In *2015 IEEE International Symposium on Nanoelectronic and Information Systems*, pages 101–106. IEEE, 2015.

[28] Amit Saha, Debasri Saha, and Amlan Chakrabarti. Circuit design for k-coloring problem and its implementation on near-term quantum devices. In *2020 IEEE International Symposium on Smart Electronic Systems (iSES)(Formerly iNiS)*, pages 17–22. IEEE, 2020.

[29] Kazuya Shimizu and Ryuhei Mori. Exponential-time quantum algorithms for graph coloring problems. In *Latin American Symposium on Theoretical Informatics*, pages 387–398. Springer, 2021.

[30] Kazuya Shimizu and Ryuhei Mori. Exponential-time quantum algorithms for graph coloring problems. In *Latin American Symposium on Theoretical Informatics*, pages 387–398. Springer, 2021.

[31] Alok Shukla and Prakash Vedula. Trajectory optimization using quantum computing. *J. Global Optim.*, 75(1):199–225, 2019.

[32] Olawale Titiloye and Alan Crispin. Graph coloring with a distributed hybrid quantum annealing algorithm. In *KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*, pages 553–562. Springer, 2011.

[33] Olawale Titiloye and Alan Crispin. Quantum annealing of the graph coloring problem. *Discrete Optimization*, 8(2):376–384, 2011.

[34] Juha J Vartiainen, Mikko Möttönen, and Martti M Salomaa. Efficient decomposition of quantum gates. *Physical review letters*, 92(17):177902, 2004.

[35] Wei Wang and Xin Liu. List-coloring based channel allocation for open-spectrum wireless networks. In *VTC-2005-Fall. 2005 IEEE 62nd Vehicular Technology Conference, 2005.*, volume 1, pages 690–694, 2005.

[36] Yushi Wang and Marek Perkowski. Improved complexity of quantum oracles for ternary grover algorithm for graph coloring. In *2011 41st IEEE International Symposium on Multiple-Valued Logic*, pages 294–301. IEEE, 2011.