Carl Klier and Nachiket Patel
CSC 591-001

Project B2 Combinatorial Optimization - Graph Coloring

In graph theory, the idea of graph coloring, specifically a vertex coloring, entails assigning colors to nodes of the graph so that no two adjacent nodes are the same color. A coloring of the vertices that uses at most k colors is a k-coloring and the graph is said to be k-colorable. The motivation to be able to solve the k-coloring problem in the most efficient way comes from the fact that there are a lot of practical applications for this problem. One such application is in scheduling a set of conflicting jobs that interfere with each other and need to be executed at different times. If we use a graph where the vertices correspond to jobs and edges exist between two jobs if the two jobs can't be run at the same time, then a k-coloring tells us that all the jobs can be completed in k time slots. The minimum value for k such that a k-coloring exists, called the chromatic number, is the minimum time required to finish all the jobs [1]. Another application for graph coloring is in compiler theory with register allocation. The goal of register allocation is to efficiently assign many variables to a limited number of cpu registers to optimize the speed of applications. This problem can be converted into a k-coloring problem just as the scheduling problem and solved to conclude that the set of variables needed at one time can be stored in just k number of registers [2].

Although the graph coloring problem is categorized as an NP-hard problem and the k-coloring graph problem is NP-complete for any integer $k \geq 3$, there have been many algorithms and heuristics developed to solve the problem on classical computers. One obvious exact algorithm is brute force search which iterates through k values starting at 1 until a valid coloring is found. However, this is impractical for large graphs. Since no known polynomial time algorithm exists, it is necessary to use heuristics to arrive at a suboptimal solution in polynomial time. One common algorithm is the greedy coloring algorithm which takes a list of vertices in some order and simply assigns each vertex the smallest possible color [3]. While the greedy algorithm succeeds in finding a valid coloring, the order of the vertices determines whether the algorithm finds the chromatic number and finding this optimal ordering is non-trivial [2].

The goal of our project is to solve the graph coloring problem on a quantum computer by implementing the problem in two different ways, first on a quantum annealing machine such as D-wave's system and secondly on a gate based quantum computer such as IBM Q's machine. We want to compare and contrast these two implementations analyzing how the problem was programmed, specifically the effort involved in deriving a hamiltonian for the annealing machine compared to designing a circuit for the gate based machine. We also want to compare the quality and accuracy of the solutions and the computational speedup over classical approaches. Lastly, we want to explore the limitations of current quantum computers in solving the graph coloring problem.

D-Wave Annealing

With the graph coloring problem being a constraint satisfaction problem, it can be represented as a binary quadratic model with unary encoding to represent the colors. The first step would be to formulate the problem as a graph, where each node is represented with c variables, which correspond to the number of possible different colors in the problem. From the initial description of the D-Wave implementation, it is obvious that for large graphs, the problem becomes infeasible for embedding into a unit cell of the D-Wave system because most QUOBOs for such problems are too large [4]. To avoid this problem, either the problem size has to be restricted or a decomposition technique will have to be implemented to allow problems to be scaled up [5].

An interesting property of the graph coloring problem is that it exhibits symmetry, this can be utilized as the number of colors in the graph increases to simplify the problem. Due to this symmetry, scaling up from c colors to c+1 colors can be done systematically without the need for complete reconstruction of the problem [6]. One of the questions to consider with the D-Wave implementation of the graph coloring problem is how the time complexity of classical algorithms compare to the D-Wave algorithm.

IBM Q using QAOA

The generalized quantum method for the K-coloring problem which has N nodes (regions on the map) will require somewhere in the order of O(N^2) and O(N^4) gates to formulate the problem. Like the D-Wave method, the size of the problem is a major consideration when attempting to formulate a graph coloring problem with quantum approximation optimization algorithm (QAOA) due to the physical limitation of quantum computers. The main steps in solving the graph coloring problem using the QAOA method are to first derive the objective function for the graph and determine the constraints to impose on the graph. From this, the hamiltonian matrix can be constructed, and lastly, we can use it to derive the appropriate ising model [2]. Something to consider with QAOA is that it is an approximation algorithm and, as such, the problem to which it can be applied varies depending on the acceptable tolerance/deviation from the absolute optimal solution.

An alternative technique for the gate model is VQE (Variational Quantum Eigensolver) which is a hybrid algorithm which attempts to find eigenvalues of the hamiltonian matrix for the problem. However, the advantage of QAOA is that it is guaranteed to arrive at the approximate solution for a combinatorial problem such as the graph coloring problem. Furthermore, if the depth of the circuit goes to infinity then QAOA will converge to the exact solution [2].

Proposed timeline for completing the project:

Week of 10/4 - Initial problem description due
Week of 10/11 - Ask questions and receive more clarification from professor
Week of 10/18 - Implementation of problem for D-wave annealing machine and for IBM Q machine
Week of 10/25 - Analysis of results and first draft due for review
Week of 11/1 - Make necessary changes and explore suggested ideas
Week of 11/8 - Finalize the project and receive any last feedback
11/17 - Turn in final project

Project Page: https://github.ncsu.edu/nnpatel5/Project-B2-GraphColoring/wiki

Sources

[1] Marx, D., “Graph Colouring Problems and their Applications in Scheduling”. Periodica Polytechnica Ser. El. Eng. Vol. 48, No. 1, PP. 11-16, Dec. 2003.

[2] Oh, Y., Mohammadbagherpoor, H., Dreher, P., Singh, A., Yu, X., & Rindos, A, “Solving Multi-Coloring Combinatorial Optimization Problems Using Hybrid Quantum Algorithms,” 2019, https://arxiv.org/abs/1911.00595.

[3] Kosowski, A., & Manuszewski, K. “Classical Coloring of Graphs,” 2008. http://fileadmin.cs.lth.se/cs/Personal/Andrzej_Lingas/k-m.pdf

[4] Dwave. https://arcb.csc.ncsu.edu/~mueller//qc/qc18/readings/dwave4-map.pdf [Accessed 8 October 2020]

[5] Wang, Yang & Lü, Zhipeng & Glover, Fred & Hao, Jin-Kao. (2012). A Multilevel Algorithm for Large Unconstrained Binary Quadratic Optimization. 395-408. 10.1007/978-3-642-29828-8_26.

[6] Dahl, E. D., D-Wave Systems, “Programming with D-Wave:Map Coloring Problem,” 2013, https://www.dwavesys.com/sites/default/files/Map%20Coloring%20WP2.pdf

Do, M., Wang, Z., O'Gorman, B., Venturelli, D., Rieffel, E., & Frank, J, “Planning for Compilation of a Quantum Algorithm for Graph Coloring” 2020, https://arxiv.org/abs/2002.10917.

“Map Coloring,” Map Coloring - Ocean Documentation 3.1.1 documentation. [Online]. Available: https://docs.ocean.dwavesys.com/en/latest/examples/map_coloring.html. [Accessed: 08-Oct-2020].

Docs.ocean.dwavesys.com. 2020. Large Map Coloring — Ocean Documentation 3.1.1 Documentation. [online] Available at: <https://docs.ocean.dwavesys.com/en/stable/examples/map_kerberos.html> [Accessed 8 October 2020].

Maurice Clerc. A general quantum method to solve the graph K-colouring problem. 2020. Ffhal02891847v2f

Shimizu, K., Mori, R. “Exponential-time Quantum Algorithms for Graph Coloring Problems,” 2019, https://arxiv.org/pdf/1907.00529.pdf

Banerjee, Asmita & Behera, Bikash & Das, Kunal & Panigrahi, Prasanta. (2019). Checking and Coloring of Graphs Through Quantum Circuits: An IBM Quantum Experience. 10.13140/RG.2.2.20371.22568.