Applications of the Quantum Approximate Optimization Algorithm to Cybersecurity DDoS Graph Partitioning

Casimer DeCusatis, Meghan OLoughlin, Alex Baida, and Chrisopher Pellerito
Marist College
Poughkeepsie, NY USA
casimer.decusatis@marist.edu

Abstract— **While the theoretical principles of quantum computing have been known for decades, practical quantum computers have only recently been developed. These systems are currently limited to a small number of qubits, but are sufficient to develop working proof of concept implementations that can eventually be scaled to much larger applications. In this paper, we investigate a near term application of the quantum approximate optimization algorithm (QAOA) to perform graph partitioning on cybersecurity attack patterns in a security operations center. Specifically, we analyze cyberattack data from honeynets represented as a hive plot, and apply QAOA in an effort to sandbox network nodes affected by a distributed denial of service (DDoS) attack. Using the IBM Q System One, we demonstrate a QAOA solution for the Max Cut problem (written in Qiskit) and a proof-of-concept application to DDoS attack hive plot data sets taken from our honeynet. Experimental results of this analysis and extensions of this work to larger systems will also be discussed.**

Keywords—Quantum, QOAO, hive, DDoS

I. Introduction

There has been a remarkable increase in both the number and severity of cybersecurity attacks in recent years, which are expected to cost over $2 Trillion to the global economy [1]. One of the most serious examples is the use of massive botnets for malware delivery and distributed denial of service (DDoS) attacks. In a recent 3-month period, botnet attacks increased 29% to nearly 17,000 per day [2]; over the past year, the average DDoS attack size has increased over 540%, with the maximum attack size exceeding a terabit/second. There is a significant need for faster, novel techniques which enable intrusion detection, visualization, and response to botnet attacks in a security operations center (SOC). While this is a multi-pronged effort, in this paper we consider the application of quantum computing techniques to the analysis of cyberattack graphs in the SOC.

Quantum computing is an emerging field which incorporates fundamentals of both quantum physics and computer science. This approach holds the potential to solve certain exponential execution time problems which are beyond practical limits of current digital computers. While the theoretical basis of this field has been understood for many decades, only within the past few years have working quantum computers become available. One of the largest near-term quantum computers is the IBM Q System One, which is programmed using the Qiskit language [3]. Much of the near-term research in this field involves proof of concept implementations, which are limited by the scalability of current quantum computer hardware (currently the largest publicly available systems from IBM are only about 5-8 qubits). Nevertheless, it's critical to study these applications at a small scale now, in preparation for larger quantum computers becoming available within the next few years. (IBM roadmaps plan for a 1,000 qubit machine by 2023 [4]).

In this paper, we discuss the application of a quantum computing co-processor to parse hive plot graphs of honeynet cyberattack patterns. We describe an implementation of the Max Cut problem in Qiskit capable of creating binary graph node classification. With each node in the attack graph represented by one qubit, we demonstrate this work using data sets previously collected from our own honeynets (these data sets have previously been employed for developing and training adversarially resistant machine learning classifiers [5]). Due to the limitations of near-term available quantum computing hardware, we demonstrate this approach for a relatively small attack graph; simulations suggest that the algorithm will be scalable to larger DDoS attacks on future quantum computing systems.

II. QAOA and Max Cut

The QAOA algorithm is an example of combinatorial optimization [6], a class of algorithms which attempts to find an optimal solution by maximizing/minimizing a cost function of a discrete variable, C(x). We encode the optimization problem as a Hamiltonian operator, H (a Hermitian matrix which describes the total energy of a quantum system), such that the lowest energy state corresponds to the optimal solution. The energy of a quantum system in a state ψ is given by the expectation value with respect to H, which can be expressed in bra-ket notation:

$$\text{Energy } (|\psi >) = < \psi \mid H \mid \psi \ >$$

The lowest energy or ground state ψ* of a quantum system is the value of ψ for which the expectation is minimized:

$$|\psi^* > = \text{argmin Energy} |\psi >)$$

A variational method such as QAOA may be used to approximate the ground state ψ * and the minimum energy of a quantum system. First, we choose a trial state or ansatz parameterized by some value θ (in other words, we only consider a subset of the entire Hilbert solution space). Next, we vary θ in order to minimize the energy value. We seek the value θ* for which the energy of the trial state is lowest,

$$\text{Energy } (\theta^*) = <\psi(\theta) \mid H \mid \psi(\theta) >$$

By running QAOA on a quantum computer, we calculate an energy value, which is then passed to a classical optimizer (such as used in the *numpy* library in Qiskit) that computes updated values of θ for successive iterations of QAOA until we converge on an approximate solution.

A classic example is the Max Cut problem, which was the first application described in the original QAOA paper [7]. Max Cut is a quadratic unconstrained binary optimization (QUBO) problem. Consider an n-node non-directed graph G = (V, E) where V is the set of vertices (such that |V| = n) and E is the set of edges between nodes i and j which may be assigned some weight w(i, j) > 0 and for which w(i, j) = w(j,i) . A cut is defined as partitioning the original set V into two subsets, so that each node is a member of either the first or second subset. The cost function C(x) is the sum of the weights of edges connecting points in the two different subsets (i.e. we want to cut the graph in such a way that the value of the edge weights crossing the cut is maximized). Each node i is assigned a value of either x(i) = 0 or x(i) = 1 to indicate which of the two subsets contains that node. In this way, we can generate a binary bit string of i values, representing the assignment of each note to either the first of second subset. Then C(x) is given by a sum over indices 0 to n-1 as follows:

$$C(\mathbf{x}) = \sum_{i,j=1}^{n} w_{ij} x_i (1 - x_j).$$

The traditional, brute-force approach to this problem requires that we exhaustively try all the possible binary assignments for each node (in each binary assignment, the entry of a vertex is classified as belonging to either the first or second partition) and check the weight of the cut associated with each graph partition. For a graph with N nodes, the total number of graph partitions grows as 2^N and the problem cannot be solved in less than exponential time with respect to the number of nodes. In general, there are two approaches for dealing with such problems. First, we can use an approximation algorithm which is guaranteed to find a solution of specified quality in polynomial runtime. Second, we can use a heuristic algorithm which doesn't have a polynomial runtime guarantee, but appears to perform well on some instances of the problem. The QAOA is an example of the second approach. For classical algorithms, it has been shown that it is NP-hard (i.e. no classical algorithm with polynomial runtime exists) that achieves a better approximation ratio than 0.941 [8]; in fact, the best classical approximation achievable is 0.878 [9 - 11]. This would still represent a significant performance advantage for identifying and blocking DDoS attacks. Further, computing the Max Cut of nonplanar graphs (i.e. graphs whose edges cross each other, such as those generated by honeynets) is known to be an exponential execution time problem for classical algorithms, but may be solvable in polynomial time using quantum algorithms for at least some specific cases.

We can encode Max Cut instances as Hamiltonian operators, and then use variational methods such as QAOA to find the ground state, which should be a good approximation of the optimal solution. Our preferred implementation of QAOA is a layerized quantum circuit (with p layers) based on an adiabatic process [9]. This form is chosen because, although there is no performance guarantee for QAOA in general, there is a guarantee that we will obtain the best possible solution as p tends to infinity. Thus, we expect that this form may yield good solutions for large values of p. In fact, it can be shown that QAOA solutions to Max Cut for graphs with bounded degree and a circuit for which p=1 achieve an approximation ratio of about 0.692 [6, 7, 9]. Further, since our result will be improved by the choice of a good ansatz, we can begin by preparing a quantum system in the ground state of a simple Hamiltonian. Then we transition to a more complex Hamiltonian whose ground state corresponds to the desired optimization solution. The adiabatic theorem tells us that we can remain in the ground state while transitioning to the more complex Hamiltonian, which yields the solution to optimization problem. A good way to implement this is to break up the problem Hamiltonian into the sum of two different Hamiltonians, known as a cost Hamiltonian (parameterized by γ) and mixer Hamiltonian (parameterized by β). Thus we need to optimize for two parameters at each of the p layers used in QAOA. This formulation means that we can implement QAOA as an adiabatic schedule alternating between the cost and mixer Hamiltonians. Note that since QAOA is a variational algorithm, the mixer layers are needed to perturb the quantum state between successive cost layers. The time evolution of the quantum state is obtained by exponentiating the sum of the cost and mixer Hamiltonians (while a Hamiltonian represents the energy of a quantum system, time evolution of the quantum state is also governed by a Hamiltonian per Schrodinger's equation). This can be computed to a very good approximation by the Trotter-Suzuki formula [6]. It turns out that the layers of QAOA correspond to Trotterized segments of the time evolution operator for this adiabatic evolution [9].

Our implementation of QAOA represents each node in the cyberattack graph as a qubit. First, we apply Hadamard gates to all qubits to create an equal superposition state. Next, we apply p repetitions of alternating cost and mixing layers, represented by unitary quantum gates that form the exponentiation of the cost and mixer Hamiltonians, as shown in figure 1.

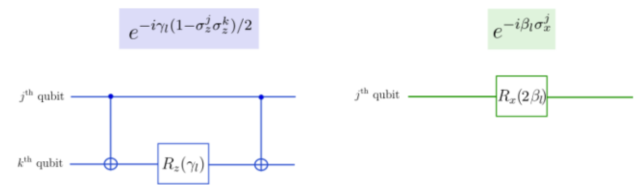

Figure 1: unitary gate for cost (left) and mixer (right)

The cost function can be re-written with a variable reassignment such that it maps to a cost Hamiltonian that can be expressed as a weighted sum of Pauli gates, each acting on a single qubit [6]. Exponentiating the cost Hamiltonian yields controlled single rotation gates [9], which can further be decomposed into two CNOT gates and one rotational gate. The mixer layer is defined as a sum over all Pauli X operators, and the exponentiation of the mixer Hamiltonian yields a rotational X gate. So, the mixer layer just consists of rotational X gates applied to each qubit, and parameterized by β. The QAOA circuit can contain as many layers as we like, each characterized by a single parameter for cost and a single parameter for mixing. The output is measured in the computational z basis, and consists of a series of binary bit strings corresponding to possible solutions of Max Cut. Our full quantum circuit is shown in figure 2. We then use the *numpy* classical optimizer [6] to evaluate the value of the cuts and compute its mean, which serves as the cost function value. The classical optimizer then updates the parameters so we can re-run the quantum circuit, until we converge on a good approximate solution. In other words, we can think of the bit strings that correspond to the Max Cut of a graph as the ground state of a Hamiltonian encoding the cost function. The QAOA algorithm arrives at a good approximation to this ground state by evolving from a reference state. The reference state is given by the ground state of a Hamiltonian that couples all 2^N states forming the basis of the cost Hamiltonian (i.e. the diagonal basis for the cost function, or the Z computational basis). We prepare an approximation to the ground state of this Hamiltonian, and perform a measurement of that state in the Z basis. Performing a measurement on the N-qubit quantum state return a bit string corresponding to the max cut with high probability.

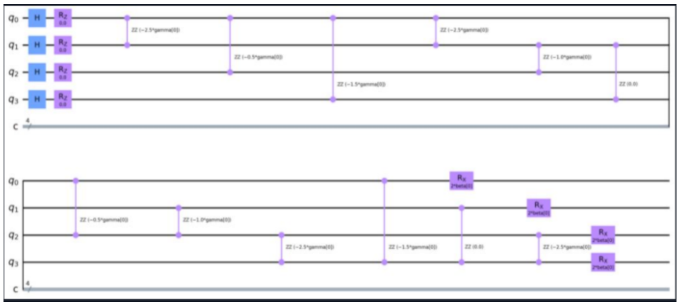

Figure 2 – Qiskit circuit diagram for QAOA with p=1

III. Experimental Results

Our implementation of QAOA utilizes the Qiskit optimization library module COBYLA [12]. This is a numerical optimization method for constrained problems where the derivative of the objective function is not required. It directly extends the Variational Quantum Eigensolver (VQE) class and inherits VQE's hybrid organizational structure. Unlike VQE, however, QAOA employs its own unique computational approach which is configured using the single integer parameter, p, that describes the depth of the variational form.

We validated our implementation of QAOA using a simple four node graph with a known Max Cut solution. We performed 100 iterations on random four node graphs to ensure that our algorithm consistently yields the expected results, before attempting this algorithm on live honeynet data graphs. An existing honeynet was used to collect a large dataset of real world attack patterns, which were used to study both DoS and DDoS attacks. A a subset of this data combining DoS and normal traffic is shown in figures 3 and 4. As in prior research [5], we visualize the cyberattack data as a hive plot, which has the advantage of not introducing artifacts into the data set. In order to test our approach using a near-term quantum computer, we were limited to about 10 qubits as of this writing. The edge weights for DoS traffic are set higher than for normal traffic. Edge weights can be assigned in several different ways, for example weights may be proportional to the trust level associated with the assets being accessed, the trust level associated with the incoming data source, or the bandwidth of the network connection, among others. Figure 4 shows the edge weights for normal traffic set equal to 1 and for attack traffic equal to 5. Such a large separation is not necessary for our approach to work. We determined experimentally that for our implementation, QAOA is not very sensitive to the relative values of the edge weights; a difference of only 0.01 between normal and attack edge weights yielded the same results. We also verfied the graph in Figure 3 to be nonplanar using a depth-first search algorithm (i.e. the *networks* library in Python) based on Kuratowski's Theorem and Wagner's Theorem [10], so it isn't possible to find a Max Cut solution for this graph in polynomial time using classical algorithms.

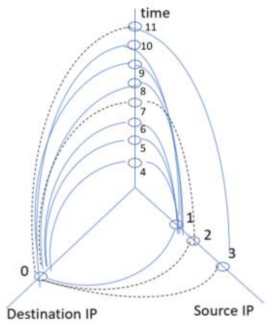

Figure 3 – Sample honeynet graph, mixing DoS traffic (solid edges) and normal traffic (dashed edges)

	0	1	2	3	4	5	6	7	8	9	10	11
0	0	5	1	1	5	5	5	1	5	5	5	1
1	5	0	0	0	5	5	5	1	5	5	5	1
2	1	0	0	0	0	0	0	1	0	0	0	0
3	1	0	0	0	0	0	0	0	0	0	0	1
4	5	5	0	0	0	0	0	0	0	0	0	0
5	5	5	0	0	0	0	0	0	0	0	0	0
6	5	5	0	0	0	0	0	0	0	0	0	0
7	1	0	1	0	0	0	0	0	0	0	0	0
8	5	5	0	0	0	0	0	0	0	0	0	0
9	5	5	0	0	0	0	0	0	0	0	0	0
10	5	5	0	0	0	0	0	0	0	0	0	0
11	1	0	0	1	0	0	0	0	0	0	0	0

Figure 4 – edge weights for data in figure 3, with nodes identified in bold along the top and left edge

Results of the QAOA solution for this graph are shown in figure 5. As expected, the result does not isolate all the attack nodes for p = 1, but after multiple iterations it successfully isolates about 69% of attack nodes. The ability to isolate this fraction of attack nodes would be a significant benefit to cyberdefenders. Improvements in the QAOA result are expected for higher values of p, with a tradeoff of slightly longer execution times.

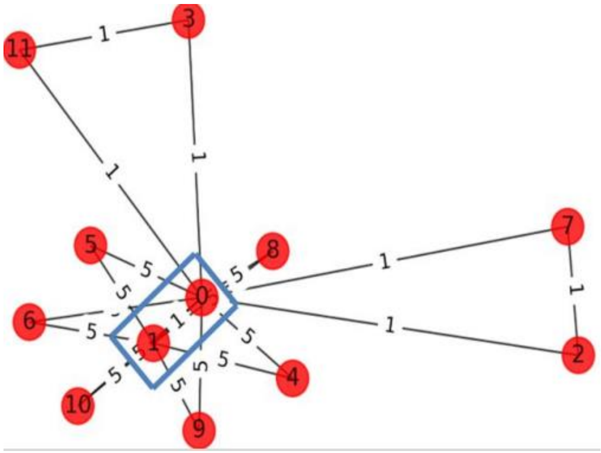

Figure 5 – QAOA solution for the hive plot of Figure 3.

Additional experiments were performed for a DDoS attack are summarized by the hive plot and QAOA solution in figures 6 and 7, respectively. As before, we were able to successfully isolate about 68% of the attack nodes (for example, note that one of the DDoS nodes is on the wrong side of the cut).

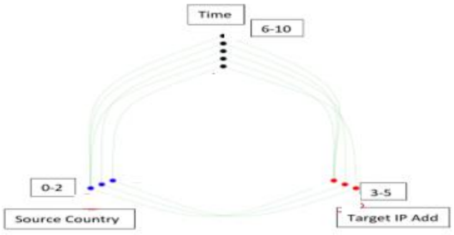

Figure 6 – hive plot for DDoS attack data

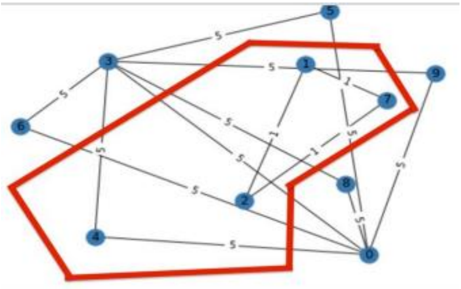

Figure 7 – QAOA Max Cut solution for DDoS attack graph

Due to the near term qubit limitations of our quantum computer, a comparison with brute force methods is possible to demonstrate the accuracy of our results. In practice, a DoS or DDoS attack can consist of several hundred nodes or more, with an aggregate bandwidth on the order of terabytes. Although quantum computers large enough to analyze such attacks are not available as of this writing, we believe that the current algorithms will scale to larger quantum hardware as it becomes available. Further, we note that even the largest Dos or DDoS attacks evolve from smaller attacks over time, so if we can identify an impending attack early enough it would still be possible to block the attack using even a small quantum computer. We compared execution time for QAOA and brute force approaches using the IBM Q System simulator; results are shown in figure 5.

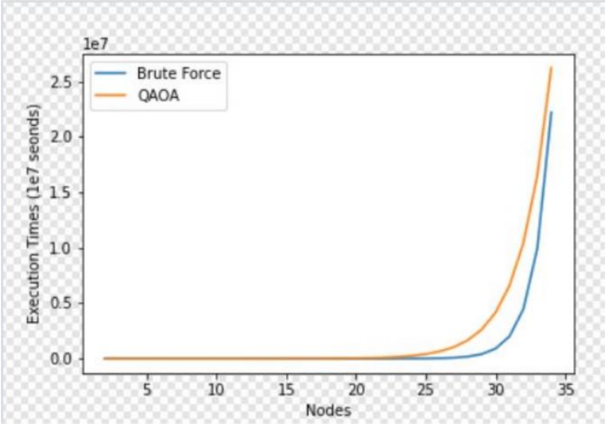

Figure 5 – Simulation of QAOA vs brute force

As expected, the simulation exhausted the memory capacity of an x86 server for large numbers of nodes, and execution times beyond the range of this graph are unreliable (our server crashed after about 2 hours attempting to run larger attack simulations). However, these results predict a performance advantage for QAOA over brute force techniques for attacks with more than 26 nodes. Future research with larger quantum computers will continue to investigate the limitations of this approach.

IV. Summary and Conclusions

There is a need for new techniques to mitigate DDoS attacks through analysis of honeynet graph data, such as hive plots. Max Cut may be a useful approach to partitioning DDoS attack traffic from normal network traffic. However, classical implementations of Max Cut for nonplanar graphs require exponential execution time as the number of attack nodes increases. There is no classical algorithm with polynomial run time that can address this problem. We propose analyzing cyberattack graphs using QAOA, a heuristic algorithm which doesn't have a polynomial runtime guarantee, but whose variational form appears to perform well on some instances of the problem. In particular, we implement QAOA as a layerized quantum circuit based on a Trotterized adiabatic process. For a single layer we achieve a significant separation between attack and non-attack traffic, and performance improves as the number of layers increases. Results suggest that the value of QAOA lies in its ability to solve Max Cut significantly faster than brute force approaches while still yielding an approximate solution that is good enough for many practical applications. After validating our implementation of QAOA in Qiskit, we demonstrate that QAOA can separate attack traffic to a high approximation level using real world honeynet DDoS data on available near-term quantum computers (10 qubits). Simulations of larger attacks suggest that QAOA has a performance advantage over brute force classical techniques for 26 nodes or more. In addition to continuing this work as larger quantum computers become available and increasing the number of layers and shots in the algorithm, future research may include investigating custom Hamiltonians, using conditional value-at-risk methods to speed up the optimization process, warm-starting QAOA from a classical optimization point, and investigating optimal parameter concentrations over different problem instances.

References

[1] S. Oriyano, "*Hacker techniques, tools, and incident handling*", Jones and Bartlett, Burlington, MA (third edition, 2021)

[2] C. DeCusatis, J. Bavaro, T. Cannistraci, B. Griffin, J. Jenkins, and M. Ronan, "Red-blue team exercises for cybersecurity training during a pandemic", Proc. IEEE CCWC conference, January 2021, New York, NY

[3] C. DeCusatis and E. McGettrick, "Near term implementation of Shor's Algorithm using Qiskit", Proc. IEEE CCWC conference, January 2021, New York, NY

[4] A. Cho, "IBM promises 1000 ubit computer by 2023", Science Insider, Feb. 2020 https://www.sciencemag.org/news/2020/09/ibm-promises-1000-qubit-quantum-computer-milestone-2023 (last accessed December 8, 2021)

[5] M. Guarino, P. Rivas, and C. DeCusatis, "Towards adversarially robust DDoS attack classification", Proc. IEEE UEMCON 2020, New York, NY

[6] *Learn quantum computation using Qiskit*, published by IBM Corporation, https://qiskit.org/textbook/preface.html (last accessed March 4, 2022)

[7] E. Farhi, J. Goldstone, and S. Guttman, "A quantum approximate optimization algorithm", Arxiv 1411:.4028 [quant-ph], https://arxiv.org/abs/1411.4028 (last accessed December 8, 2021)

[8] J. Håstad, Some Optimal Inapproximability Results, J. ACM 48, 798 (2001)

[9] J. Weidenfeller, "QAOA and its applications", Section 5.2 from Introduction to the Quantum Approximate Optimization Algorithm and its Applications, IBM Qiskit Global Summer School (October 5, 2021) https://www.youtube.com/watch?v=YpLzSQPrgSc and https://learn.qiskit.org/summer-school/2021/lec5-2-introduction-quantum-approximate-optimization-algorithm-applications (last accessed March 14, 2022)

[10] J.A. Bondy and U.S.R. Murty, *Graph Theory*, Graduate Texts in Mathematics vol 244, Springer , NY (2008)

[11] A. Bouland, "Power and limitations of QAOA", Simons Institute, February 2020 https://www.youtube.com/watch?v=jDl8n7wMSWk (last accessed March 13, 2022)

[12] Constrained optimization by linear approximation (COBYLA) API documentation, https://qiskit.org/documentation/stubs/qiskit.algorithms.optimizers.COBYLA.html (last accessed March 16, 2022)