EasyChair Preprint

№ 9887

Evaluating Quantum Algorithms for Linear Solver Workflows

Sophia Kolak, Hamed Mohamedbagherpoor, Konstantis Daloukas, Kostas Kafousas, Francois-Henry Rouet, Yorgos Koutsoyannopoulos, Nathan Earnest-Noble and Robert Lucas

March 27, 2023

Evaluating Quantum Algorithms for Linear Solver Workflows

Sophia Kolak [1,2,*], Hamed Mohammadbagherpoor[1], Konstantis Daloukas[3], Kostas Kafousas[3], Francois-Henry Rouet[3], Yorgos Koutsoyannopoulos[3], Nathan Earnest-Noble[1,*], and Bob Lucas[3,*]

[1]IBM Quantum
[2]CMU
[3]Ansys

***Abstract*—We normally think of implicit Mechanical Computer Aided Engineering (MCAE) as being a resource intensive process, dominated by multifrontal linear solvers that scale super linearly in complexity as the problem size grows. However, as the processor count increases, the reordering that reduces the storage and operation count for the sparse linear solver is emerging as the biggest computational bottleneck. Reordering is NP-complete, and the nested dissection heuristic is generally preferred for MCAE problems. Nested dissection in turn rests on graph partitioning, another NP-complete problem. There are quantum computing algorithms for which provide new heuristics for NP-complete problems, and the rapid growth of today's noisy quantum computers and associated error mitigation techniques leads us to consider them as possible accelerators for reordering. This paper reports on the evaluation of the relative merits of several short depth quantum algorithms used for graph partitioning, integration with the LS-DYNA MCAE application, and using the initial results generated on IBM quantum computers to bring to attention critical focus areas based on the methods used within.**

***Index Terms*—quantum computing, simulation, linear equations, graph partitioning**

I. Introduction

In the past few years, quantum computing has had a significant upturn - in both its development, and industrial interest. When looking at the development of quantum systems, it is easy to appreciate how quickly the field has developed in the past decade. For example, the current industry superconducting qubit of choice, the Transmon, was first demonstrated less that two decades ago [1]. In the time since the transmon's inception, it's gate fidelities and coherence times have grown orders of magnitude. Roughly one decade after the realization of the qubit itself, or three years after the transmon coherence grew to 60μsec [2], quantum systems were made available via cloud access in 2016 - being the first point in history when non-specialists could practically test the potential benefits from the systems. Since then, a variety of physical realizations are now available via cloud access, with many of them surpassing a scale in which we can classically simulate these systems. With this rapid growth of quantum hardware, the broader industry has taken an interest in understanding how it may help their clients' needs. The potential for quantum computers to improve computing capabilities in the unknown, and potentially near-term future leads to two important questions: when is the appropriate time to begin to actively invest in a quantum initiative and in what ways?

The answer to this question is different depending on the nature of a given software product. In addition to the type of software a company produces, there is the question of which problems quantum algorithms are currently well equipped to solve. Although quantum computing can in theory provide solutions more efficiently to NP-Complete problems than the classical heuristic algorithms used today [3], the present reality of both quantum hardware and software limits stake-holders to a few specific use cases, which are often much smaller than the scales needed by industry.

We begin the paper by providing context on Ansys as a company, quantifying their relative need for quantum computing according to the size of their bottlenecks and the expected growth in data volume and the performance of quantum systems. In section II we describe the Ansys technological offering and client relevant workflows, seeking where there are computational bottlenecks. We identify graph-partitioning (re-ordering) as the strongest candidate for quantum migration, over presently existing algorithms for solving this problem. In section III we layout some of the different methods available to be used for graph partitioning and give a quick introduction to practical aspects of these algorithms. In section IV we highlight how the quantum algorithms in question are expected to scale with problem size (judging by circuit width and depth), and assess the expected quality in both classical simulations of quantum algorithms and direct quantum experiment on IBM Quantum hardware. We leverage the parameters from noise free simulations to get a rapid assessment of quantum hardware capabilities. Using this information, we finish the paper with a discussion and potential next steps in sections V and VI. Given the rapid development of the underlying hardware, we try to inform our future recommendations based on estimated trends of quantum systems since the invention of the transmon. We highlight the recent growth of quantum systems, in particular those available in the cloud via IBM Quantum [4]. Based on the hardware demo results we highlight some of key milestones which would need to be achieved for full integration, and discuss the best course of action for Ansys in light of these findings.

*Corresponding author: email1@gmail.com

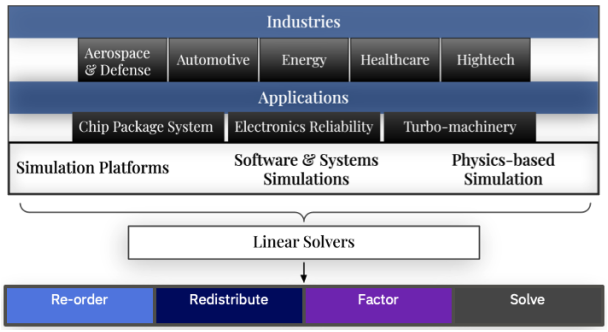

Fig. 1. A breakdown of the main industries Ansys works with, and the services they provide. Across this wide range of industries and simulation types, linear solvers are required. Within the linear solver pipeline, re-ordering (one sub-step) is a ubiquitous bottleneck, limiting the efficiency of the services at the top of the stack. The process is to Reorder, Redistribute, Factor, and then Solve.

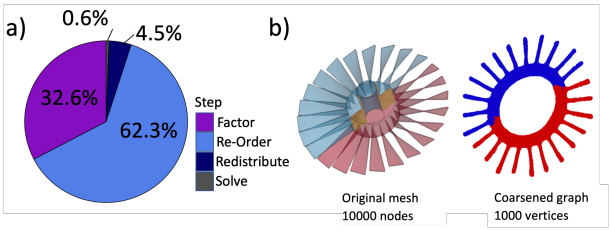

Fig. 2. a) `LS_DYNA` linear solver computing time breakdown for one load step in an implicit finite-element simulation. One first recursively partitions the graph (i.e., reordering), then redistributes the matrix, followed by factoring, and finally triangular solves. b) An impeller, when its graph is coarsened from 10,000 to 1,000 nodes, as an example of how this impacts its representation. In the subsequent analysis, we consider the graph of each object coarsened from 10,000 to as few as 25 nodes.

II. Possible Quantum Integration Pathways

Ansys is a software company that provides engineering simulation solutions that enable organizations to design and test virtual prototypes of products, systems, and materials. Ansys software is used by engineers and designers in a range of industries, including aerospace, automotive, construction, consumer goods, electronics, semiconductors, energy, healthcare, and more. Ansys' software suite includes a range of products that allow engineers to simulate and analyze different physical phenomena, such as structural mechanics, fluid dynamics, electromagnetics, signal/power integrity, and thermal management. These tools enable engineers to optimize the design of their products and systems, reduce physical prototyping costs, and accelerate time-to-market. Figure 1 shows the general breakdown of the Ansys client environment – from the fundamental need for physics-based simulation up to many of the relevant client industries. Furthermore, the figure outlines many of the the relevant computational steps which must be taken for these industrial applications. For the purpose of this study, we remained focused on pathways which impact the entirety of Ansys' clients to maximizes the potential revenue impact. We have highlighted two key challenges which become classical computational bottlenecks across a broad range of Ansys' client workflows, and are thereby strong candidates to explore for quantum integration. Some of these simulations already take days to complete, and are expected to take longer in the coming years as the users' models grow in complexity. A future impact analysis may benefit from considering the needs of specific simulation use cases (e.g. Material Science) to determine other, more tailored, pathways and assess their potential.

A. Linear Solvers

Linear solvers are not only pervasive, but present a super-linear computational bottleneck in Ansys programs. This is unsurprising, as on classical computers the process of linear equation solving (finding the vector $\vec{x}$, for the matrix A such that $A\vec{x} = \vec{b}$), for a sparse A matrix is $O(n^{1.5})$ for planar systems of equations, $O(n^2)$ for three-dimensional ones, and $O(n^3)$ for dense non-Sparse A matrix, where n is the dimension of the matrix [5]. While researchers have devised highly efficient algorithms and parallelization strategies for reducing the work involved, solving linear equations still scales polynomial with the dimension of the matrix. When the matrix A remains relatively small or has some special properties, the linear solver will finish in a reasonable amount of time. Ansys, however, is already dealing with massive matrices (A =1M-50B rows and columns), which may be sparse or dense. Today these operations can take days, but as Ansys' use cases become larger and more complex (as they are expected to in the near future), finding a solution could take weeks, and in some cases, may be effectively uncomputable via classical methods. As such, these solvers were identified early on as the high-level target for quantum migration.

Although there are potential scaling benefits to integrating quantum linear solvers with Ansys' quantum software needs, the quantum algorithms for doing so present major challenges in the both today's noisy quantum computers era and beyond. The two main quantum algorithms for solving linear equations are either fault tolerant [6] or variational [7]. In practice unfortunately, both approaches have problems. When considering fault tolerant approaches (e.g., HHL [6]), we are faced with two main concerns. First, this approach is unlikely to be implemented in the near term due to the lack of existence of a suitable fault tolerant quantum computer. Second, there are issues with experimental implementation of some associated technology (e.g., qRAM and I/O bottlenecks) which bring into question whether these algorithms can be implemented today or in the near future.. Given these points and the inability to implement these algorithms at scale on today's quantum hardware, we exclude this from our main discussion. Potentially nearer-term approaches like Variational Quantum Linear Solver (VQLS) [7] suffer from circuits which still do not transpile well to native quantum architectures, I/O bottlenecks, and even so, do not provide an exact solution (see appendix A for brief discussion around circuit depths of these algorithms). However, this is an on-going field of research [8], where people are seeking to overcome such issues for near

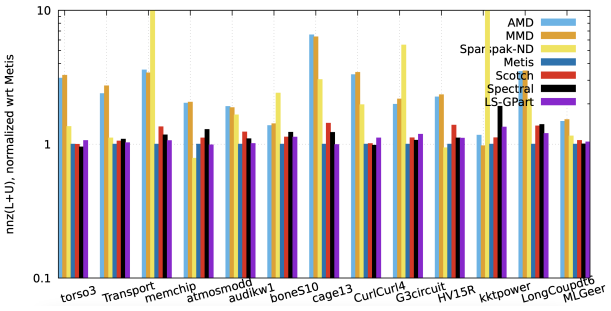

Fig. 3. A comparison of different reordering algorithms. The 14 test matrices are from the SuiteSparse Matrix Collection [11]. The initial data is from P. Ghysels et al. [12]. Ansys' LS-GPart reordering software was added to the comparison [13].

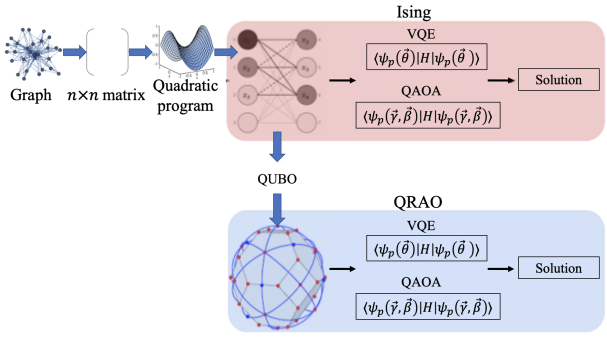

Fig. 4. Method diagram showing the general process one must go through in order to run graph partitioning problems on a gate-based quantum system Both methods take in a graph, in this case, a representation of one of the physical objects simulated by Ansys (described further in section IV-A), and produce solutions of the same form. Note that both methods follow identical steps initially, converting the graph to a matrix, the matrix to a quadratic program, and then finally the quadratic program to a QUBO.The distinction between these methods presents itself in the circuit generated, or the correspond measurements needed.

term devices and should be watched closely.

B. Graph Partitioning / Re-ordering

Since solving linear equations *in its own right* does not have an obvious practical implementation in the near future, we took a closer look at the speed of each step within the solver pipeline, which revealed that the re-ordering of sparse matrices is what dominates highly parallel workloads, as seen in figure 2a. Ansys must perform this reordering in order to efficiently perform sparse Gaussian elimination, a key component of reaching a final solution. Reordering matrices to minimize computational complexity has been an active field of research for over 60 years [9]. Nested dissection of the adjacency graph associated with the matrix has emerged as the best practical algorithm for reordering matrices derived from mechanical MCAE models [10]. Nested dissection involves recursive graph-partitioning, which is itself an NP-complete problem classically solved using heuristic approximations. Notably this is an issue for Ansys customers *today*, not tomorrow.

Since the sparse matrices Ansys factors can range to billions equations, and because linear solvers are ubiquitous bottlenecks across many MCAE codes, an improvement in this major sub-step in their solvers would have a massive effect on the overall performance of their simulations. We therefore choose to investigate the efficacy of near-term quantum approaches to graph-partitioning, a key component in linear solvers, as opposed to the process as a whole.

This presents an exciting opportunity for quantum computing. Progress has stalled on classical heuristics for nested dissection as depicted in figure 3 [13]. The figure compares 7 ordering heuristics for 14 matrices from different applications; the baseline is the Metis nested dissection package from the late 1990's [14]; the figure shows that no significant progress (in terms of quality of the reordering, i.e., number of operations and memory footprint of the linear solver) has been made in recent years. Quantum heuristics are comparatively young, and relevant hardware tests are only beginning. Considering the specific needs of Ansys' client workflows and connecting this to the IBM Quantum Development road-map [15], we next consider some critical focus points which could greatly reduce the time until real user problems can be tested for potential benefits.

III. Quantum Graph-Partitioning Methods

Quantum graph-partitioning works by first formulating the problem as a Quadratic Unconstrained Binary Optimization (QUBO) problem, and then solving this QUBO with a quantum algorithm. In the following sections we explore the options for partitioning Ansys graphs according to the method diagram in figure 4.

A. Encoding for Quantum Optimization

Generally speaking, there exist a variety of methods for encoding optimization problems into a quantum primitive, usually in the form of sampling a circuit distribution [16]–[18]. In figure 4 two encoding options which we applied to an Ansys graph partitioning problem are shown: an Ising Encoding [19] and a Quantum Random Access Encoding (QRAC) [16]. The Ising model is equivalent to Quadratic Unconstrained Binary Optimization (QUBO) [17], and is the traditional method of mapping optimization problems to gate-based quantum hardware [20]. Alternatively one translates a QUBO into a QRAC and get a more efficient measurement basis - the underpinning aspect of the Quantum Random Approximate Optimization (QRAO) algorithm [18]. QRAO in particular allows larger problems to be encoded with fewer qubits, making this solution more useful in the near-term. However, as we will detail in the next section, for the specific realize used in our experiment [18] these encoding benefits are lost for the balanced graph partitioning cost function provided by the Ansys use case. Since the start of our review, researchers have updated the QRAO method to guarantee an encoding benefit of 2x, even when using a balanced cost function [21].

Once an encoding method is chosen, an end-user must ultimately select which quantum circuit they will run. For the

purpose of our demo, we highlight two possible candidates: A hardware efficient ansatz (HEA) approach with the Variational Quantum Eigensolver (VQE) [22], or a QAOA [20] ansatz which is directly made based on the provided graph.

B. Quantum Circuits

After generating the QUBO or a QRAC, you have chosen the qubit mapping and measurement basis. From here there is the need to choose the initial state of parameters and the corresponding circuit ansatz. The initial circuit generated for a QAOA circuit is given by repetition of an evolution between your 'problem' and a 'mixer' Hamiltonian in the form:

$$|\gamma, \beta\rangle = U_{\beta,\beta_1} U_{\gamma,\gamma_p} \dots U_{\beta,\beta_1} U_{\gamma,\gamma_1} |\psi_0\rangle$$

where β represents your mixing Hamiltonian and γ your problem Hamiltonian. The depth of the QAOA ansatz can then be increased in the number of repetitions of layers p, where in the asymptotic limit of infinite p, it is shown that this converges to the optimal solution of the desired cost function [20]. For initial circuit analysis, we will consider the impact of the repeating layers p on the circuit depth.

Alternately one can use different HEAs which are heuristic in their nature and one can design any sort of ansatz which is best mapped to the quantum hardware. These ansatzes have the benefit of being able to be extremely shallow in their circuit depth, but when using these methods, there is another level of optimization which goes beyond the parameter optimization of the ansatz, but choosing the ansatz itself. There are some early studies which try to characterize these approaches, using various forms of entropy [23]. Furthermore, these methods are prone to getting stuck in local minima and even having the parameter gradient vanish exponentially – either from circuit-induced [24] noise-induced [25].

C. Warm-Starting Quantum Circuits

Quantum variational algorithms such as QAOA and VQE are utilized to solve the combinatorial optimization problems by finding the minimum energy to the corresponding formulated Hamiltonian. These algorithms combined with classical optimizers are always dependent of the initial values for the optimizer and the initial states for the quantum circuit on the Hilbert space which effect the algorithm's performance to find the minimum energy to the problem. Applying a warm-start technique that can initiate the initial value/sate to the quantum circuit would be able to improve the convergence of these algorithms to solve the problem accurately. A warm-start QAOA technique was presented in [26] in which it replaces the binary variables with the continuous variables to relax the optimization problem and solve it classically that improves the performance ratios in polynomial time. Then it uses the solution to the relaxed problem to find the initial state to the QAOA algorithm. By applying the warm-start technique combined with the QAOA algorithm, a better result with higher fidelity can be achieved. Similar approaches can be used for the VQE algorithm to either initiate the parameters of the variational circuit or modify the variational circuit to help the optimizer to converge to the correct value to find the minimum energy of the problem. A new ansatz modification technique is introduced in [27] where the variational circuit is build in a discrete approach based on the property of the problem Hamiltonian and the corresponding Pauli string. In principle this could be applied to improve our 'unfair VQE Warm-start' (see appendix B), but this is beyond the scope of this paper.

IV. Experimental Setup and Results

A. Experimental Setup

To test this method of graph partitioning, we use real Ansys graphs of physical objects from both mechanical and electromagnetic simulations. Images of the two objects used in the analysis are shown in figure 2b. Note that the number of vertices in these graphs ranges from 0.6M - 34.9M, making their raw size too large for both QAOA or QRAO to run on current quantum hardware. In order to conduct preliminary experiments, we instead use coarsened versions of these graphs, for 25, 50, 100, 1K, and 10K nodes, respectively. The effects of graph coarsening are demonstrated in figure 2b, which shows the difference in resolution when an Impeller is coarsened from 10K nodes to 1K. While coarsening does reduce the level of detail in the object graph, it still maintains the general topology and properties of the original object. This allows us to test smaller versions of real graphs on quantum hardware today, while retaining as much of the object's original integrity as possible.

With these graphs, we then conduct two series of experiments. The first is designed to test the efficacy of the selected algorithm (QRAO) on real Ansys graphs. As mentioned in Section III-A, QRAO uses quantum random access codes (QRACs) to encode a maximum of 3 qubits per node, for a slight trade off in accuracy. In order for this position to be recovered with high probability, however, two nodes that share an edge cannot be encoded on the same qubit. This means that the encoding ratio, which can range from 1-3, is variable depending on graph topology. As such, we study the compression ratio achieved for these graphs, which allows us to measure the number of necessary qubits according to graph size (number of nodes) and observe the impact of adding the balancing constraint. We also measure the required processing time for each step in the graph partitioning, as this is a relevant additive factor in tracking the overall algorithmic run-time.

The second series of experiments involves real-hardware demonstrations, and allows us to evaluate the quality of quantum hardware partitions as we increase the size of the graph being studied. Specifically, we measure how well these algorithms conform to simulated expectations and compare the accuracy of the hardware solutions against the exact solution. However, as hardware sizes continue to increase, these benchmarking approaches will no longer work. To this end, we suggest alternate long term benchmarks to assess the trustworthy-ness of a quantum run, and how to measure the corresponding accuracy against alternative, classical methods such as those found in `METIS` [14].

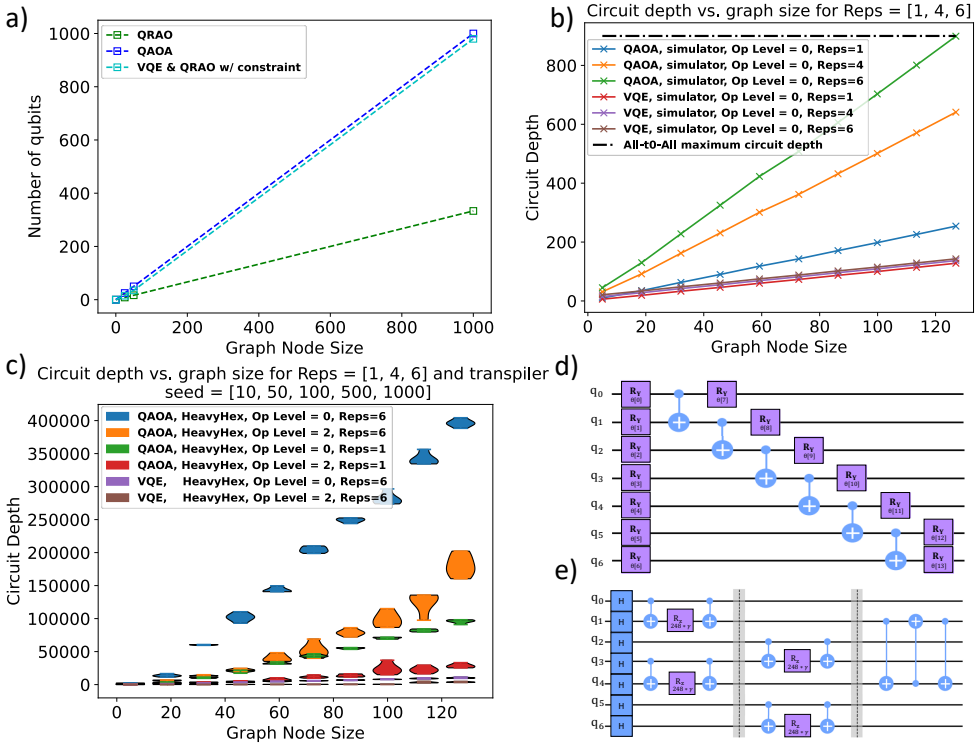

Fig. 5. a) Required number of qubits for a given graph size, for different encoding parameters. Of particular note: Ansys' desired graph partitioning with constraints does not benefit from the present QRAO encoding. b) Circuit Depth as a function of different graph node sizes, with varying levels of compilation and increased circuit depth. As an example of the importance of the problem – this figure took approximately 1 day to complete with a recent Apple MacBook Pro. c) The repeated structure of a HAE and QAOA ansatz used for our VQE and QAOA demonstrations. (This sort of repeated structure particular benefits from dynamic circuit transpilation capabilities). Due to its heuristic nature, the HEA can nicely conform to hardware topology constraints but also has issues when scaling to larger sizes (e.g. barren plateaus). d) Example structure in a VQE circuit. e) Example structure in a QAOA circuit.

B. Results

1) Encoding and Scaling Results: To begin, we startup by first determining theoretical scaling for the given approaches, as seen in figure 5. Firstly, we measure the relative encoding efficiency of QRAO against QAOA, both with and without our constraints in figure 5a. Since QAOA requires a 1:1 mapping between nodes and qubits, it requires the same number of qubits to compute a solution for a graph as the number of nodes in the graph. In contrast, QRAO requires only 334 qubits to solve the same problem for Ansys graphs, when a balancing constraints is not included. However, as the reordering process requires a graph to be bisected by considering the balancing constraint, the corresponding encoding benefits are lost. In the outlook, we highlight what this means for an Ansys integration but choose to focus on the Ising encoding for initial hardware testing and validation.

With the encoding method selected, we now wish to evaluate which quantum circuit will be ideal to run for the given problem on a given hardware architecture. To do this, we evaluate how a quantum circuit depths depend on both the graph size and the transpilation method used for HEA [28] or the graph-informed ansatz of QAOA [20]. Specifically, we use a HEA circuit with $TwoLocal$ entanglement, or a p=1 realization of a QAOA circuit. As seen in figure 5b, the circuit depth grows much more rapidly for a QAOA circuit compared to the HEA ansatz, making it more unlikely to have a successful evaluation on quantum hardware.

2) Hardware Results: Results from hardware demos, figure 6, will measure accuracy, and max size computable on current quantum hardware. For the purpose of our hardware comparison, we want to note that we choose to focus on a 'quality' metric for comparative result. Though the required time for computation is quite important for ultimate comparison, the practical nuances of getting an accurate required compute time (utilizing the state-of-the-art techniques vs what is available via cloud providers) makes this comparison difficult to properly judge. To this end we refrain from elaborate in this work have keep this for future investigation.

V. Outlook

From our exploration, we have found that for beneficial quantum graph partitioning (reordering) at larger scales, current circuit fidelities require more extensive error mitigation protocols than used here. Though we limited our use of error mitigation to readout error mitigation [29], this is more a matter of the availability of certain software features, moreso than fundamental limitation and could be address in subsequent work. Furthermore, we showed that using an warm start was essence for our QAOA circuits to converge to the

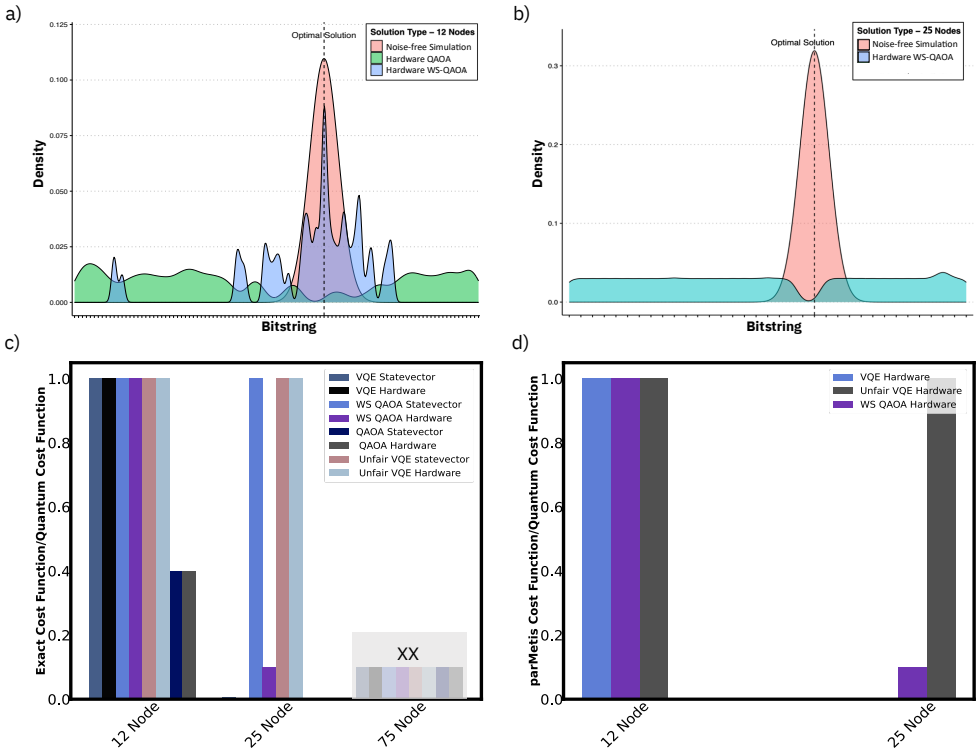

Fig. 6. Quantum hardware results comparison to noise free simulations. From this we can see that quantum hardware has an increase count of bit strings. a) The result of noise-free simulation, QAOA and WS-QAOA for 12 node graph. b) The result of noise-free simulation and WS-QAOA for 25 node graph. c) A plot comparing the solutions of hardware to exact solution calculated by CPLEX. *= this is a benchmark of the circuit and is not a valid comparison for VQE since we do not yet have a proper iterative test. Specifically, we manually tuned a one rep=1 HEA ansatz (in the assumption such an ansatz may find the minimum). Due the the slowness of statevector simulations, we are opting to run the VQE algorithm on the quantum hardware directly. In some sense, this is an accomplishment in its own right – though it does not provide direct value for someone wishes to use this for an application. XX = We cant do these - which necessitates a longer term benchmark. Statevector simulations are impossible, regardless of what compute power you put behind it which makes a 75 node exact solution to become impractical. Namely, it took more than 5 hours to try and simulate this on a laptop. d) A more appropriate long-term in which we compare our solution to Metis. This is the benchmark which can be easily used beyond 50 node graphs. For the method of comparison we compare both the Metis and Quantum Hardware cost functions.

proper solution. However, as we increase the size of the graph the QAOA circuit scale sufficiently fast to make our initial attemps at 25 node paritioning non-trivial. Furthermore, we show that in principle one can get substantially shorter circuit depths but still achieve good results with our 25 node 'unfair warmstart'. Though this cannot be practically used at larger scales, there may be means, such as the VQE Hot Start [27], which can enable larger graph partitioning. Furthermore, though the Quantum Random Access Code implementation used in our testing did not have encoding benefits when including the balancing constraints – a recent update to the algorithm makes it possible to get at least a factor of two of savings. Each of these factors will be considerably important when trying to understand how to achieve useful quantum advantage for linear solvers. For immediate next steps, we will seek to achieve an integration to enable a more complete characterization of quantum workflows and to fully evaluate expected runtimes as quantum systems increase in size and quality.

VI. Acknowledgements

The authors acknowledge use of the IBM Quantum devices for this work. The authors are also thankful to Bryce Fuller, Stefan Woerner, Rudy Raymond, Paul Nation and Antonio Mezzacapo for insightful discussions and Stefan Woerner and Paul Nation for a careful read of the manuscript.

References

[1] A. Wallraff, D. I. Schuster, A. Blais, L. Frunzio, R.-S. Huang, J. Majer, S. Kumar, S. M. Girvin, and R. J. Schoelkopf, "Strong coupling of a single photon to a superconducting qubit using circuit quantum electrodynamics," *Nature*, vol. 431, no. 7005, pp. 162–167, 2004.

[2] J. B. Chang, M. R. Vissers, A. D. Córcoles, M. Sandberg, J. Gao, D. W. Abraham, J. M. Chow, J. M. Gambetta, M. Beth Rothwell, G. A. Keefe, *et al.*, "Improved superconducting qubit coherence using titanium nitride," *Applied Physics Letters*, vol. 103, no. 1, p. 012602, 2013.

[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," *In Proceedings of the 28th ACM Symposium on the Theory of Computing*, pp. 212–219, 1996.

[4] "https://quantum-computing.ibm.com."

[5] V. Pan, "Complexity of algorithms for linear systems of equations," in *Computer Algorithms for Solving Linear Algebraic Equations* (E. Spedicato, ed.), (Berlin, Heidelberg), pp. 27–56, Springer Berlin Heidelberg, 1991.

[6] S. Lloyd, "Quantum algorithm for solving linear systems of equations," in *APS March Meeting Abstracts*, vol. 2010, pp. D4–002, 2010.

[7] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles, "Variational quantum linear solver," *arXiv preprint arXiv:1909.05820*, 2019.
[8] A. Montanaro and S. Pallister, "Quantum algorithms and the finite element method," *Physical Review A*, vol. 93, no. 3, p. 032324, 2016.
[9] H. M. Markowitz, "The elimination form of the inverse and its application to linear programming," *Management Science*, vol. 3, no. 3, pp. 255–269, 1957.
[10] A. George, "Nested dissection of a regular finite element mesh," *SIAM Journal on Numerical Analysis*, vol. 10, no. 2, pp. 345–363, 1973.
[11] T. A. Davis and Y. Hu, "The university of florida sparse matrix collection," *ACM Transactions on Mathematical Software*, vol. 38, pp. 1–25, Nov. 2011.
[12] P. Ghysels, X. S. Li, G. Chávez, Y. Liu, M. Jacquelin, and E. Ng, "Preconditioning using rank-structured sparse matrix factorization," in *SIAM Conference on Computational Science and Engineering 2019*.
[13] R. F. Lucas, C. Ashcraft, J. Dawson, E. Guleryuz, R. Grimes, S. Koric, J. Ong, F.-H. Rouet, T. Simons, and T.-T. Zhu, "Implicit analysis of jet engine models on thousands of processors," in *Sparse Days 2019*.
[14] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998.
[15] "https://www.ibm.com/quantum/roadmap."
[16] S. Wiesner, "Conjugate coding," vol. 15, pp. 77–78, 1983.
[17] F. Glover, G. Kochenberger, and Y. Du, "A tutorial on formulating and using qubo models," 2018.
[18] B. Fuller, C. Hadfield, J. R. Glick, T. Imamichi, T. Itoko, R. J. Thompson, Y. Jiao, M. M. Kagele, A. W. Blom-Schieber, R. Raymond, *et al.*, "Approximate solutions of combinatorial problems via quantum relaxations," *arXiv preprint arXiv:2111.03167*, 2021.
[19] A. Lucas, "Ising formulations of many np problems," *Frontiers in physics*, vol. 2, p. 5, 2014.
[20] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," *arXiv preprint arXiv:1411.4028*, 2014.
[21] K. Teramoto, R. Raymond, E. Wakakuwa, and H. Imai, "Quantum-relaxation based optimization algorithms: Theoretical extensions," *arXiv preprint arXiv:2302.09481*, 2023.
[22] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature Communications*, vol. 5, July 2014.
[23] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao, "Expressive power of parametrized quantum circuits," *Physical Review Research*, vol. 2, no. 3, p. 033125, 2020.
[24] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, "Cost function dependent barren plateaus in shallow parametrized quantum circuits," *Nature communications*, vol. 12, no. 1, pp. 1–12, 2021.
[25] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles, "Noise-induced barren plateaus in variational quantum algorithms," *Nature communications*, vol. 12, no. 1, pp. 1–11, 2021.
[26] D. J. Egger, J. Mareček, and S. Woerner, "Warm-starting quantum optimization," *Quantum*, vol. 5, p. 479, 2021.
[27] B. Polina, S. Arthur, Z. Yuriy, Y. Manhong, and l. Dingshun, "Hot-start optimization for variational quantum eigensolver," 2021.
[28] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," *Nature*, vol. 549, no. 7671, pp. 242–246, 2017.
[29] E. v. d. Berg, Z. K. Minev, and K. Temme, "Model-free readout-error mitigation for quantum expectation values," 2020.
[30] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles, "Variational quantum linear solver," 2019.
[31] A. C. Vazquez, R. Hiptmair, and S. Woerner, "Enhancing the quantum linear systems algorithm using richardson extrapolation," *ACM Transactions on Quantum Computing*, vol. 3, no. 1, pp. 1–37, 2022.

VII. Appendix

A. BRISQ Cycles

In reflection of our process of mutual learning, we wanted to try to put down an repeatable process, in the hopes to try and help other interested quantum application practitioners. With a very new technology like quantum, there is an ever increasing amount of discovery happening every day. To this end, we want to shape a process, which we call 'Building Reliable Industry-Scale Quantum' (BRISQ), that accounts for a changing environment of capabilities but helps focus towards an end goal of future deployment for practical value. Each BRISQ cycle is as follows:

- Assess
- Build
- Evaluate
- Discover

Assess the present set of options for integration and pick best candidate. Build a means of testing the feasibility of this candidate on today quantum systems. Evaluate the given algorithm quality and capability to scale to problems of client relevance. (Can do so quickly with rule of thumbs, e.g. circuit depth and fidelity, to rule out candidates with more obvious issues. Use Quantum hardware longer term). In discovery, one summarizes results into few key discoveries (e.g. performance against existing methods in use today and optimal means to improve). Repeat this process with a goal of marching different long term problem scales, starting small and increasing with new capabilities as they develop.

1) BRISQ Cycle 1 - VQLS: Our first BRISQ cycle focused on the Variational Quantum Linear Solver [30]. At the point in time, the possible existing Qiskit packages were not properly updated, and thereby required a more involved build cycle if we wished for a proper hardware evaluation. We opted to first consider the quantum circuits of interests and investigate how they might scale. To this end, we present the figure 7 from [31]. This is the underlying quantum circuit which needs to be run for the VQLS algorithm. Notably the circuit include multiple gates which are multi-controlled and multi-targeted. In the case of presently existing superconducting quantum hardware, such gates are not native, and must be instead compiled into the equivalent circuit with only single and two qubit gates. (Though it is possible in smaller scale cases to try and engineer multi-target or multi-control gates more pulse efficiently, this does not scale well and those demonstrated gates do not have ideal gate fidelities with current demonstrations.) As such, when trying to transpile the circuit in figure 7 to the $ibm - sherbrooke$ backend, we get a circuit depth that is on the order of 2,800! This would approximately require a gate fidelity on the order of 0.9999 to achieve reasonable results – and just for a 5 qubit circuit! Scaling prospects with existing methodology used seem unlikely.

2) BRISQ Cycle 2 - Ansys encoding ratio without constraints: With the existing methods for linear solutions with quantum systems (e.g. VQLS), we note rather deep quantum circuits and thereby seem unlikely to be a viable path forward. For our next cycle, we revisited the Ansys workflow and note that the graph partitioning component was a dominate source of present compute time. We chose this for our second brisq cycle to evaluate its feasibility. Initially we want to reduce the potential time to run client relevant workflow on IBM quantum systems via a reduced qubit requirement from the

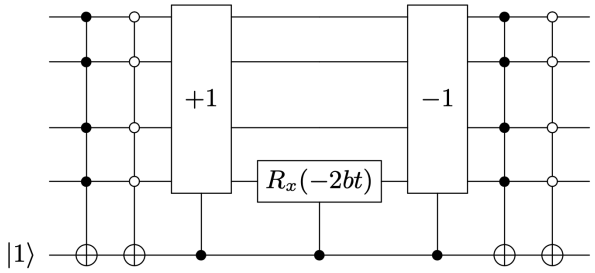

Fig. 7. VQLS circuit. Left Top - not-transpile. Left Bottom –part of the transpiled circuit for backend with all-to-all connectivity. Right – corresponding circuit depth as a function of circuit size

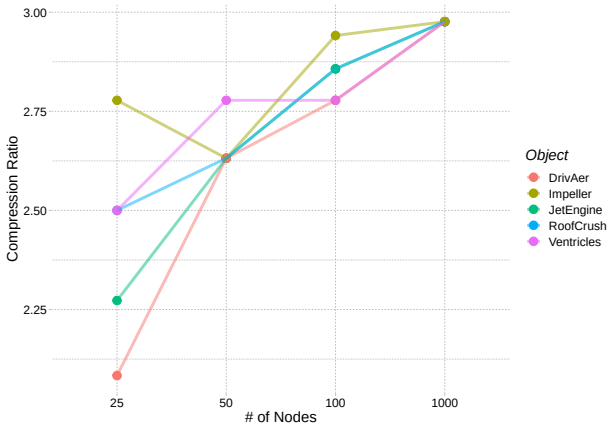

Fig. 8. This graph shows the encoding ratio (number of nodes over number of qubits) vs the size of the provided Ansys graph, when not including the balancing constraint. Of interest at the time, we noticed an improve ratio for larger graphs (likely because of the graph become more sparse at larger sizes).

Quantum Random Access Optimization (QRAO) algorithm. figure 8 shows the how compression ratio for Ansys graphs behaves for each increases graph sizes. Specifically the figure shows that as the coarsened graph approaches 1,000 nodes the compression ratio also climbs closer to 3 (the maximum ratio for the QRAO algorithm). From this it appears QRAO may serve as a method to increase the quantum capabilities of a quantum graph partitioning tool. However, when doing statvector simulations we note that the balancing constraint for this encoding resulted in a densely connected graph and no longer benefitted from the QRAO encoding. This led us to select a more common ising encoding, and pursue further benchmarking to evaluate limitations arising from hardware and/or barren plateaus.

B. Unfair VQE Warm-start

The Variational Quantum Eigensolver (VQE) is one the foremost promising algorithms for tackling combinatorial optimization problems that can be implemented on the near term quantum devices. However, the feasibility of the solution depend on the initial value for the classical optimizer and selecting an appropriate ansatz, which can be arbitrary selected. [27]

In this work we mainly focused on initializing the VQE algorithm so that the algorithm converges to the correct parameters. The warm-Start method applied in this work utilizes the manually selected parameters for the VQE variational circuit to help the optimizer to converge to the correct value and minimize the Hamiltonian's expectation value. The idea was to select the parameters so that the expectation value of the circuit is close to the expected value for the problem. The goal was to analyze the performance of the hardware for graph partitioning of a larger graph (25 nodes). The results show that by applying the initial parameters to the VQE algorithm, the algorithm is able to converge to the correct parameters that can be used to run the circuit on the actual quantum hardware and evaluate the performance of the hardware for partitioning a larger graph. The step for warm-start VQE would be as follows:

1. Manually selection of the parameters for an arbitrary chosen variational circuit (Ansatz) to get the approximation of the correct expectation value for the derived Hamiltonian from the graph partitioning problem.
2. Initiate the VQE algorithm with the selected parameters as an initial value to the classical optimizer.
3. Utilize the converged parameters from the VQE algorithm and bind those parameters to the varaitional circuit.
4. Run the circuit on a hardware through the Sampler primitives with enabling the readout error mitigation technique.
5. The returned quasi binary string with the highest probability would be the solution to the graph partitioning problem.