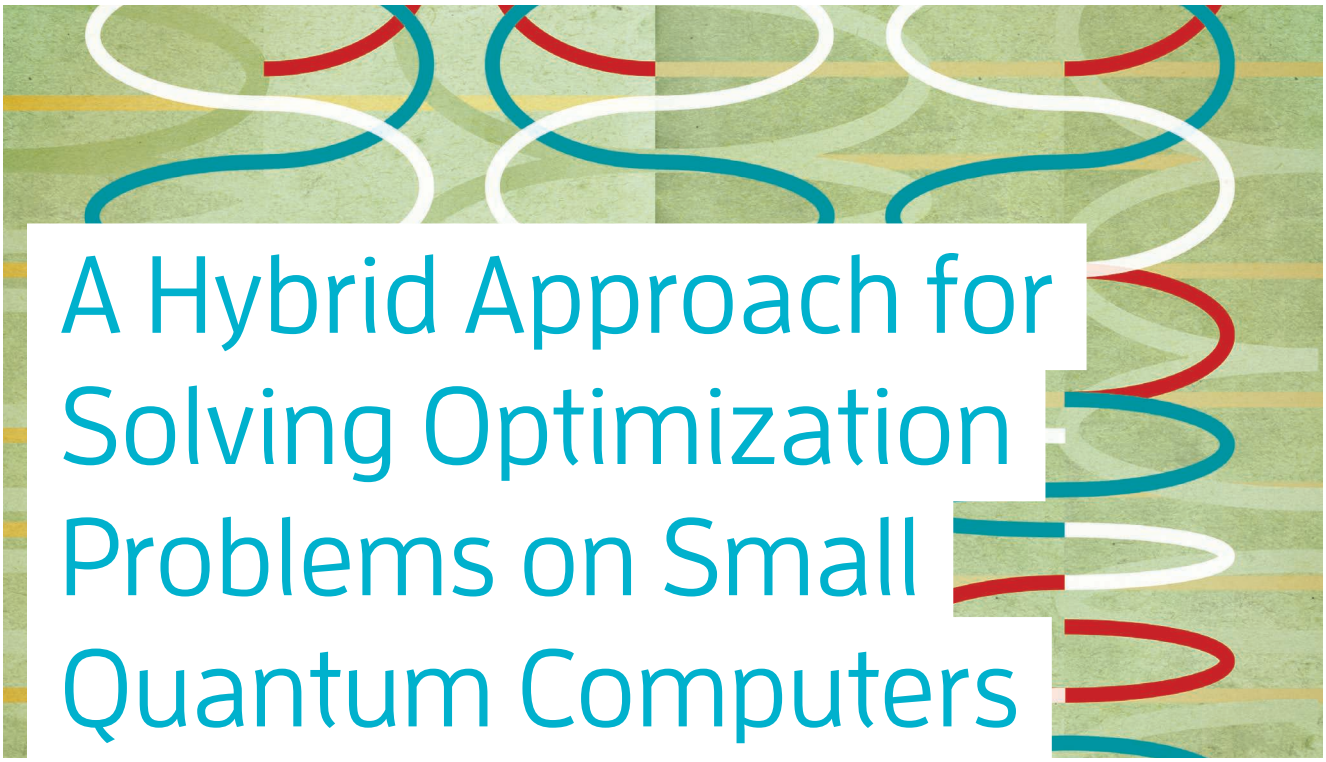

A Hybrid Approach for Solving Optimization Problems on Small Quantum Computers

Ruslan Shaydulin, Clemson University
Hayato Ushijima-Mwesigwa, Fujitsu Laboratories of America
Christian F. A. Negre, Los Alamos National Laboratory
Ilya Safro, Clemson University
Susan M. Mniszewski, Los Alamos National Laboratory
Yuri Alexeev, Argonne National Laboratory

Solving larger-sized problems is an important area of research in quantum computing. Designing hybrid quantum-classical algorithms is a promising approach to solving this. We discuss decomposition-based hybrid approaches for solving optimization problems and demonstrate them for applications related to community detection.

In recent years, quantum devices with up to tens of qubits on universal quantum computers (UQCs) and a few thousand qubits on quantum annealer (QA) devices have become available. This enabled researchers to use real quantum hardware to solve toy problems for the first time. Unfortunately, in the near term, quantum computers are expected to stay very limited, in both the number and quality of qubits, which makes it difficult to use them for practical applications that often require hundreds or even thousands of qubits. Challenges such as qubit connectivity limitations, high noise levels, and full error-correction overhead and concerns about scalability raise questions about near-term ability of quantum hardware to effectively incorporate a larger number of qubits and deliver the theoretical

Digital Object Identifier 10.1109/MC.2019.2908942
Date of publication: 4 June 2019

0018-9162/19©2019IEEE

speedups that have been promised by many algorithms developed since the 1990s.

A hybridization of quantum and classical algorithms is one of the expedient answers that researchers suggest to tackle real-life problems with existing quantum hardware. These hybrid algorithms combine both classical and quantum computers in an attempt to acquire the best of both, leveraging the power of quantum computation while using a classical machine to address the limitations of noisy intermediate-scale quantum (NISQ) computers (see Figure 1). This is true not only for optimization algorithms but also for other problems, including quantum simulation,[5] quantum machine learning,[2,12,16] and more.[10] For example, classical computers have large memories and are capable of storing the entire global problem, which is a challenge for NISQ devices that have a small number of qubits. At the same time, quantum algorithms have shown improved performance for certain problems. To distinguish unambiguously between the stages of computation performed on two principally different types of hardware, we will refer to the classical and quantum stages of hybrid algorithms as CPUs (including accelerators as such graphics processing units and field-programmable gate arrays) and quantum processing units (QPUs), including a quantum annealer and a universal quantum computer, respectively.

We primarily focus on two classes of NISQ-era devices, UQCs and QAs, using IBM and D-Wave as exemplars. The IBM devices belong to the class of UQCs that evolve the system by applying gates described using quantum assembly language. Other companies developing UQCs include Rigetti, Google, Microsoft, and IonQ. Alternatively, D-Wave devices are QAs designed to solve computational problems via quantum evolution toward the ground states of the cost Hamiltonians that encode optimization problems. While the two paradigms are very different, they share many limitations and challenges.

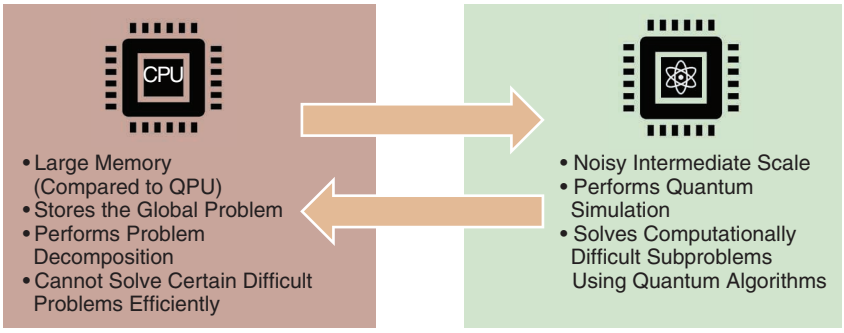

FIGURE 1. The hybrid algorithms combine computations performed in both (a) a classical computer (CPU) and (b) an NISQ quantum computer (QPU). These algorithms are designed to leverage the strengths of each mode of computation while dealing with its weaknesses. For example, CPUs cannot perform quantum simulation efficiently, whereas modern small near-term QPUs cannot compute problems with many variables.

As a problem becomes too large to run directly on quantum computers, decomposition methods are required to split it into pieces of QPU-manageable size, an idea borrowed from high-performance computing and classical numerical methods. Static methods partition the global problem into subproblems that run on the QPU and assemble all the pieces into a final solution on the CPU, while dynamic techniques use data-driven classical processing that produces subproblems that run on a QPU as the solution

QUANTUM COMPUTERS ARE EXPECTED TO STAY VERY LIMITED, IN BOTH THE NUMBER AND QUALITY OF QUBITS, WHICH MAKES IT DIFFICULT TO USE THEM FOR PRACTICAL APPLICATIONS.

Certain instances of graph partitioning[20] and community detection, also known as graph clustering,[11,19] provide examples of practically important NP-hard optimization problems that can already be solved by quantum computers. A small graph problem can be solved directly on a QPU, while larger graph problems of practical interest require hybrid quantum-classical approaches.

evolves. Decomposition can be combined with compression or hierarchical approaches, such as the multilevel method, to better utilize a problem structure before decomposing it for a QPU. Bian et al.[3] and Shaydulin et al.[19] provide examples of such schemes applied to hybrid quantum-classical algorithms.

> DEMONSTRATING THE QUANTUM ADVANTAGE ON NISQ DEVICES IS AN ACTIVE AREA OF RESEARCH.

The decomposition scheme we discuss is a natural evolution of variational quantum algorithms like the variational quantum eigensolver (VQE) algorithm and the quantum approximate optimization algorithm (QAOA). The classical computer not only finds better parameters but also finds an optimal subproblem size to solve on a quantum computer followed by an aggregation step. Given the size of current quantum computers, this approach does not allow for quantum speedup because it is limited to the speedup achieved at the subproblem level. The fundamental difference with this classical scheme is that the subproblem size is chosen to maximize the quantum speedup, which means running calculations on as many qubits as possible. Of course, two n-qubit quantum computers would be less powerful than one $2n$-qubit computer. Using decomposition schemes limits the attainable multiplicative speedup as the size of the global problem increases. When quantum hardware improves enough to be able to tackle a given problem directly, decomposition methods might not be the best approach because of the overhead introduced. However, in the absence of $2n$-qubit computers, decomposition-based methods provide a way to take advantage of the computational power of existing hardware.

In this overview, we do not focus on analyzing the performance and speedups from quantum optimization algorithms like D-Wave quantum annealing and QAOA. For an overview of these state-of-the-art methods, refer to a recent report by the U.S. Department of Energy's Office of Scientific and Technical Information[13] that shows that QAOA provably outperforms the best-known classical approximation algorithm for especially difficult cases of the MAXCUT problem. Instead, we focus on a different question: if these methods work, how can we use them to solve practical problems (i.e., find solutions of satisfactory quality) under the limitations of NISQ-era hardware? Assuming the methods can deliver speedups on problems that can fit on small near-term quantum computers, how can we leverage these speedups to solve problems of practical importance? All decomposition-based methods described in this article rely on the ability of quantum optimization methods to deliver quantum speedups on NISQ devices. Demonstrating the quantum advantage on NISQ devices is an active area of research.

Only a few frameworks provide an easy way to implement hybrid algorithms. Examples include the XACC framework,[8] which was developed at Oak Ridge National Laboratory for both UQCs and D-Wave QAs, the Hybrid workflow platform[1] by D-Wave, and Rigetti's Quantum Cloud Services for UQCs. These frameworks follow the traditional coprocessor model, i.e., treating QPUs as coprocessors to execute specialized kernel code, while considering the complexity of the interplay between classical and quantum hardware.

OVERVIEW OF ALGORITHMIC APPROACHES

In this section, we provide an overview of the QA and UQC paradigms and their specific hybrid methods for optimization. This is followed by a description of a general hybrid approach using the local search method, an optimization technique, solving subproblems on NISQ QPU devices (referring to both UQC and QA) with the main driving routine working on the CPU.

Quantum annealing

QA devices are available from D-Wave in the form of the 2X (up to 1,152 qubits) and 2,000 Q (up to 2,048 qubits). Using entanglement, they minimize the Ising model objective function, which is composed of biases and strengths that encode the Hamiltonian problem. QAs are able to solve problems that arise in optimization, machine learning, sampling, and simulation. The D-Wave solves problems formulated as a maximization by using the negative of the objective function. On many occasions, it is easier to use 0 and 1 as possible values for variables, which leads to the problem formulated as a quadratic unconstrained binary optimization (QUBO).

Currently, D-Wave annealers have physical constraints such as limited precision, sparse connectivity, and a

limited number of available qubits. Problem variables do not map one to one with the available qubits. Each problem variable is represented by a chain of qubits obtained by embedding or mapping a problem onto the hardware Chimera graph prior to annealing. Quantum-only solutions are limited by the largest number of graph nodes/variables that can be represented on the D-Wave hardware. On the 2000Q, problems up to size 64 can use the same embedding for 64 fully connected nodes/variables. Quantum-classical approaches are required for larger problems.

D-Wave's qbsolv tool is available for solving large QUBO problems using a hybrid quantum-classical algorithm.[4] In this case, an optimization solver drives dynamic decomposition. During each iteration of this algorithm, a large neighborhood local search is performed, in which subQUBOs are solved using the QPU, followed by tabu improvements on the CPU. The size of a subQUBO is limited by the number of variables that can be embedded in the hardware. The qbsolv loop converges to a low-energy solution.

Universal (gate-based) quantum computers

Near-term UQC devices are widely expected to have no more than a few hundred nonerror-corrected qubits.[14] These NISQ computers cannot run many of the most famous QC algorithms with asymptotic speedups, such as Shor's, for problems of practical size.[14] This is due to both the small number of qubits and the limited number of gates that can be executed before the errors accumulate and the output becomes no longer useful due to decoherence. To address this challenge, a number of quantum-classical algorithms were introduced, of which the most famous are QAOA and VQE. QAOA can use either a variational or analytical strategy for finding parameters.

Variational algorithms combine a small QPU and CPU with the goal of finding the ground state of a Hamiltonian problem, which can be purely classical (QAOA) or quantum (VQE). The ansatz is prepared by applying a series of parameterized gates. An ansatz or trial state is prepared on the QPU, and its energy is measured. This process is outlined in Figure 2.

The advantage of variational algorithms is that the ansatz can be chosen so that the number of gates required is small enough to run on NISQ devices feasibly. This can introduce various tradeoffs in terms of the quality of the trial state, complexity of the classical optimization, and the amount of accumulated error. For example, in our earlier work,[19] we used a recently introduced hardware-efficient ansatz[6] [see Figure 3(b)]. This ansatz uses natural entangling interactions available on the device and therefore introduces fewer errors; however, it is much more difficult for a classical optimizer to find optimal parameters.[9]

VARIATIONAL ALGORITHMS ADDRESS ONLY THE FIRST PART OF THE PROBLEM, THE QUALITY OF THE QUBITS.

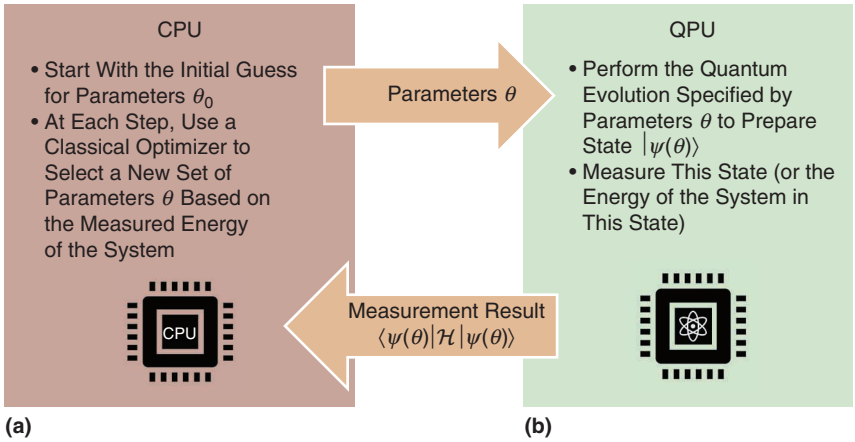

FIGURE 2. The general outline of variational hybrid algorithms. (a) The variational cycle starts with some initial guess θ_0. Then, a trial state $|\psi(\theta)\rangle$ is prepared on (b) the QPU at each step, measured, and (b) the measurement result is read by the CPU. A classical optimization routine uses this measurement to select the next set of parameters θ and the cycle proceeds. This cycle continues until a solution with satisfactory quality is discovered or the classical optimizer converges.

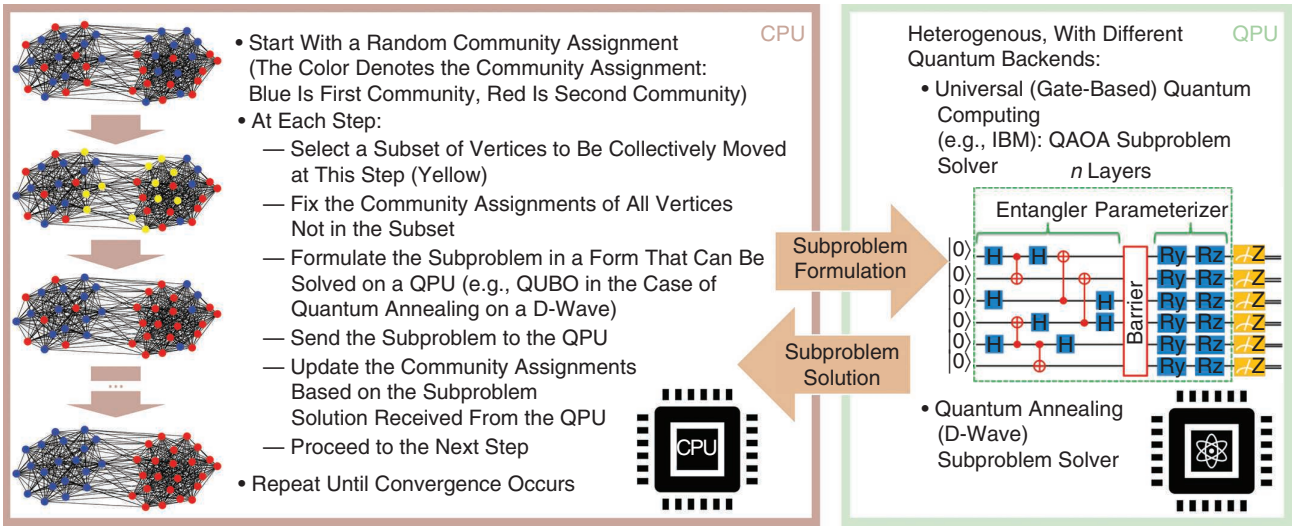

FIGURE 3. An outline of the quantum local search applied to the problem of two-way network community detection (graph clustering with two clusters). Note that algorithms in (b) (the QPU box) can be hybrid themselves, like QAOA described in Figure 2. The (b) set of gates prepares the hardware-efficient ansatz used in QAOA [6].

A general hybrid approach

Since emerging NISQ devices have a very limited quality and number of qubits, they cannot directly tackle many problems of practical size. Variational algorithms address only the first part of the problem, the quality of the qubits. By using shallow-depth ansatzes, we can reduce the number of gates and bypass the issue of quickly accumulating errors. However, that still leaves the issue of problem size.

A natural way to address this challenge is to decompose the problem (statically or dynamically), solve the computationally difficult subproblems on the QPU (either UQC or QA), and combine them on the CPU to obtain a global solution. The recently introduced quantum local search (QLS)[18,19] is one method that utilizes this approach. The QLS is inspired by the success of numerous local-search heuristics that have been applied to a variety of computationally difficult problems (such as the satisfiability and traveling salesman problems) that otherwise are problematic for global solvers to tackle in a reasonable amount of time. In QLS, the entire (global) problem is stored on the CPU. The QLS starts from a random initial solution, and a neighborhood of the current solution is searched to improve the objective function at each step. Since the neighborhood can be restricted to be small, this search can be performed on the QPU. If a better solution is found, the current one is updated, and the QLS proceeds. This step is repeated until the objective function can no longer be improved. Similar approaches have been applied to mapping optimization problems to quantum annealers.[3,7,17]

The QLS has the advantage in that, by storing the entire (potentially, very large) problem on the CPU and only solving the computationally intensive subproblems on the QPU, we are able to tackle large problems using limited hardware. The subproblems offloaded to the QPU are not trivial; as larger quantum hardware becomes available, algorithms like QA and QAOA have the potential to demonstrate a quantum advantage on these subproblems. We were able to cluster graphs of up to 400 nodes using only a small 16 qubit QPU.[19]

We observe that the QLS reaches the global optimum (confirmed by Gurobi) for these problems, i.e., finds the optimal clustering of nodes. Other heuristics with runtimes much faster than the QLS exist. We do not compare the running time of the methods since it falls outside the scope of this work. However, this method has an obvious downside: it is prone to being stuck in a local optima. If, at a given iteration, all neighborhoods we consider were too small to find an improvement, the algorithm would not be able to climb out of a local optima.

This issue is not unique to the QLS. Generally, iterative local improvement methods are not sufficient to obtain high-quality solutions for large problems unless they are combined with other global search algorithms. This limitation has led to the creation of multilevel (or multiscale) optimization in which scale interactions of the given problem play an important role. The idea behind this is to create a hierarchy of smaller problems, which are easier to solve, and then work backward toward the solution of the original problem by using a solution inherited at the coarser level of the hierarchy to initialize the next-finer level. The hierarchy forms a basis to make global decisions for a given problem. More precisely, with respect to the graph-partitioning problem, a graph is gradually coarsened until a partition can be computed efficiently, which is then projected back onto the original graph while being refined at all levels of the hierarchy.

In the context of hybrid quantum-classical algorithms, the main driving routine that creates a hierarchy and assembles the final solution at all levels is performed on the CPU, and a costly refinement that solves subproblems is performed on the QPU. There is a variety of fast classical multilevel heuristics for combinatorial optimization problems. Outperforming them is one of the most challenging tasks that face quantum optimization methods in general.

APPLICATIONS

After introducing the concepts and ideas related to the application of NISQ devices on large-scale problems, we focus on two applications: 1) the network community detection for which we apply the quantum local search[18,19] with both IBM Q and D-Wave architectures and 2) imidazole glycerolphosphate synthase (IGPS) protein sub-structure discovery using D-Wave and *qbsolv*.

Network community detection on IBM Q and D-Wave

The network community detection aims to group vertices based on their similarity, which is often expressed in the number of shared immediate and distant neighbors between vertices. Such groupings frequently lead to dense and sparse link connectivity within and between the groups, respectively. Both the densification and sparsification of inner and outer connections serve as objectives for a big class of community detection approaches. One of the most commonly used methods is modularity maximization.

In modularity maximization, the goal is to split the set of vertices of an underlying graph into two subsets (communities) such that the difference (modularity) between the actual number of edges within a community and the expected number of edges, if the edges are distributed uniformly at random over the same number of vertices, is maximized. In other words, modularity maximization looks for a statistically surprising distribution of edges. We focus on finding an optimized assignment for two communities.[19] However, there are several approaches to extend the problem to cases with more than two communities.[11]

Solving community detection on NISQ computers is challenging because networks from practical applications are too large to be fully mapped onto the near-term hardware, which justifies an application of the QLS to address this issue. Figure 3 is an outline. The QLS starts with some initial solution, which is a community assignment in the context of community detection. We start with a random community assignment, but in general it is possible to start with some other initial guess that is a result of a heuristic with acceptable time/quality tradeoff.[19]

At each step of the QLS, a neighborhood of the current solution is explored. In two-way community detection, the solution space is all possible community assignments. Therefore, a current solution's neighborhood is community assignments similar to it, one that only differs by a small number of vertices. The neighborhood search is then performed at each step as follows. A subset of vertices to be moved between communities is selected based on some optimization criteria (we choose the highest gain change in the objective[19]). The assignments of all other vertices remain invariant, and a new optimization subproblem is formulated by encoding these invariant vertices as a boundary condition.

By restricting the neighborhood size (i.e., the number of vertices in the subset), we can reduce the number of variables in an optimization subproblem until it is small enough to fit on an NISQ device. This requires a tradeoff since increasing the size of the neighborhood improves the convergence of the algorithm.[19] However, as new more capable quantum hardware becomes available, the QLS has the potential to outperform the classical state of the art.

An additional benefit of the QLS is that it is fundamentally hardware agnostic. As long as the subproblem can be mapped to a quantum algorithm executable on a given hardware, the QLS can use it. With community detection, the subproblem is in Ising form, which maps directly to both QA and QAOA. We demonstrate that the QLS is able to find optimal solutions using 16 variable subproblems with both the IBM 16 qubit

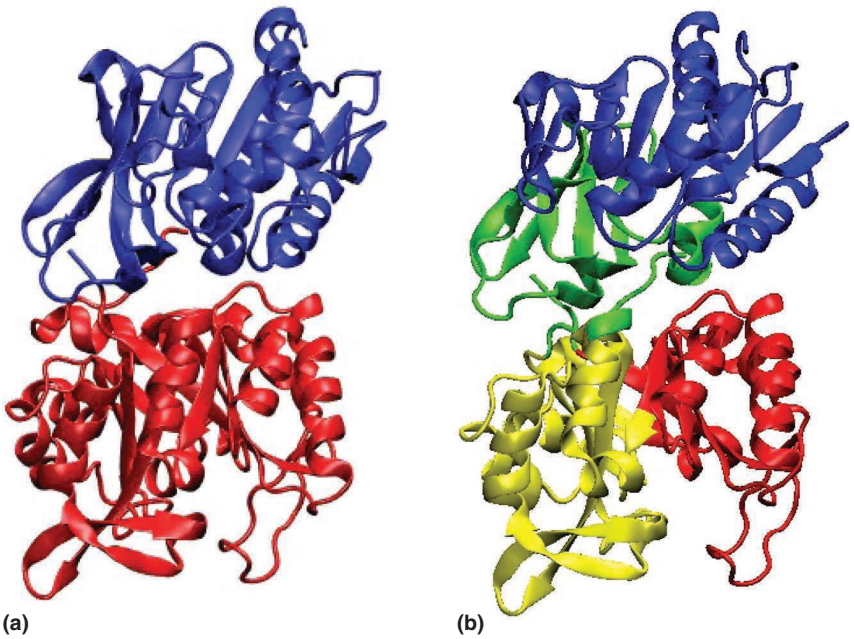

FIGURE 4. The IGPS protein representation showing the split into two communities: (a) two different molecules and (b) four communities as two domains per molecule.

UQC device and the D-Wave 2000Q QA device.[19] Note that the subproblems of up to 64 fully connected variables can be solved on the D-Wave 2000Q using the same embedding. The IBM 20 Q Tokyo, the largest UQC device available, allows for up to 20 variable subproblems.

Multiple community detection using D-Wave

We show an example of the applicability of the all at once (*k*-concurrent) community detection method for the D-Wave QA.[11] In an earlier work, we demonstrated that multiple community detection using the D-Wave QA leads to highly accurate community splits, which, in most cases, match the best-known optimal solutions, particularly when graphs are small.

To show how useful this technique is, we selected an example corresponding to the community detection on dynamically correlated amino acid residues of the IGPS protein. This archetypal system has been extensively studied in the field of biophysics.[15] The modularity matrix is calculated from the α-carbons pair-based correlation matrix obtained from a molecular dynamics simulation.[15] The 454 amino acid residues of the IGPS protein form a graph with edge weights described by the correlation matrix.

The QUBO formulation for the global problem of up to k communities was created with $k \times n$ variables (where k is the maximum number of communities and n is the number of vertices) and served as input to qbsolv. A subproblem size of 64 variables was used when running qbsolv. The split into two communities resulted in naturally revealing the two molecules that compose IGPS (Figure 4). Solving for $k = 4$ communities resulted in identifying four domains (two per molecule) that may be related to the protein functionality[15] (Figure 4).

The results obtained with this example are not surprising and serve to draw some useful conclusions. They demonstrate that quantum-classical algorithms using QAs applied to graph problems are moving out of the toy model exploratory phase. Allowing more communities in the split automatically leads to the need for more qubits, hence justifying the need for hybrid quantum-classical algorithms.

In this article, we discussed the hybrid quantum-classical approaches to quantum computing. Due to the limitations of NISQ QPU devices, using them as accelerators or coprocessors for solving domain-specific problems in combination with a CPU is one of the most promising approaches. Classical computers are used to prepare and postprocess data from quantum computations but also in critical steps to find parameters for variational algorithms or produce subproblems using decomposition methods.

We demonstrated a few practical applications designed to work for such setups on IBM and D-Wave quantum computers. These applications are formulated as combinatorial optimization problems, but hybrid computing is not limited only to them. It is important to note that the hybrid quantum-classical approach will likely continue to be relevant even as quantum machines scale in the foreseeable future. First, there is no clear path to tens of thousands, not to mention millions, of qubits. Second, it is clear that classical computers will remain better at certain things (for

example, it is much more efficient to store large amounts of classical data on the classical computer), something that hybrid algorithms address by trying to leverage the best of both. Finally, there are always bigger and more complicated scientific problems to solve.

The hybrid decomposition-based approaches we advocate are not a silver bullet. They allow us to leverage small quantum computers to solve practical problems, but they are still limited by the capabilities of NISQ devices. If small quantum computers demonstrate quantum speedup, the decomposition-based methods will benefit from it and also show speedup on large optimization problems with a modest overhead. If, on the other hand, there is no quantum speedup on NISQ devices, the decomposition methods will not be able to create it. The same considerations apply to other hybrid algorithms. If the quantum part of a quantum-classical algorithm cannot show any speedup on an NISQ device, then hybridization alone would not be able to provide it.

We believe that these major issues can be solved and that quantum computing has a promising future. We are currently seeing results comparable to or better than state-of-the-art specialized classical methods in terms of quality of the solution, although the runtime performance is lacking. Over the next 10 years, we expect improvements in qubits (quality, count, and connectivity), error correction, and quantum algorithms that will make improvements in runtime possible. Improvements in gate error rates will allow for the more precise manipulation of qubits and enable more advanced computation. These improvements will provide the missing ingredients for decomposition and hybrid quantum-classical methods, which rely on robust quantum computers that are capable of exhibiting a quantum advantage. As hardware evolves beyond the NISQ era, we believe decomposition methods will evolve with the abilities of quantum devices.

ABOUT THE AUTHORS

RUSLAN SHAYDULIN is pursuing a Ph.D. in computer science at Clemson University. His research interests include quantum and classical algorithms for combinatorial optimization and machine learning. Contact him at rshaydu@g.clemson.edu.

HAYATO USHIJIMA-MWESIGWA is a research scientist with Fujitsu Laboratories of America. Ushijima-Mwesigwa received a Ph.D. in computer science from Clemson University. Contact him at hushiji@g.clemson.edu.

CHRISTIAN F. A. NEGRE is a staff scientist at Los Alamos National Laboratory. His research interests include computational physical chemistry, quantum molecular dynamics, and molecular electronics. Negre received a Ph.D. in physical chemistry from the National University of Córdoba, Argentina. Contact him at cnegre@lanl.gov.

ILYA SAFRO is an associate professor of computer science and faculty scholar in the School of Health Research, Clemson University. His research interests include graph algorithms, machine learning, combinatorial scientific computing, and quantum-classical methods. Safro received a Ph.D. in applied mathematics and computer science from the Weizmann Institute of Science. Contact him at isafro@g.clemson.edu.

SUSAN M. MNISZEWSKI is a senior scientist at Los Alamos National Laboratory. Her research interests include quantum computing, neuromorphic computing, tensor networks, and quantum molecular dynamics on high-performance computing architectures. Mniszewski received a B.Sc. in computer science from the Illinois Institute of Technology. Contact her at smm@lanl.gov.

YURI ALEXEEV is a principal project specialist at Argonne National Laboratory. His research interests include computational quantum chemistry, computational biology, molecular dynamics, quantum computing, and high-performance computing. Alexeev received a Ph.D. in physical chemistry from Iowa State University. Contact him at yuri@anl.gov.

ACKNOWLEDGMENTS

This research used the resources of the Argonne Leadership Computing Facility, which is a U.S. Department of Energy (DOE) Office of Science User Facility supported under contract DE-AC02-06CH11357. The

authors would also like to acknowledge the National Nuclear Security Administration (NNSA) Advanced Simulation and Computing program at Los Alamos National Laboratory (LANL) for use of their Ising D-Wave 2000Q quantum computing resource. LANL is operated by Triad National Security for the NNSA of DOE under contract 89233218NCA000001. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. government. Yuri Alexeev and Ruslan Shaydulin were supported by the DOE Office of Science.

REFERENCES

1. D-Wave, "D-Wave releases hybrid workflow platform to build and run quantum hybrid applications in leap quantum application environment," Quantum Computing Company, Burnaby, BC, Dec. 2018. [Online]. Available: https://www.dwavesys.com/press-releases/d-wave-releases-hybrid-workflow-platform-build-and-run-quantum-hybrid-applications/
2. J. Biamonte et al., "Quantum machine learning," *Nature*, vol. 549, no. 7671, p. 195, 2017.
3. Z. Bian, F. Chudak, R. B. Israel, B. Lackey, W. G. Macready, and A. Roy, "Mapping constrained optimization problems to quantum annealing with application to fault diagnosis," *Frontiers ICT*, vol. 3, p. 14, July 2016. [Online]. Available: https://doi.org/10.3389/fict.2016.00014
4. M. Booth, S. P. Reinhardt, and A. Roy, "Partitioning optimization problems for hybrid classical/quantum execution," D-Wave, Quantum Computing Company, Burnaby, BC, Tech. Rep. 14-1006A-A, 2017. [Online]. Available: https://www.dwavesys.com/sites/default/files/partitioning_QUBOs_for_quantum_acceleration-2.pdf
5. S. Bravyi, G. Smith, and J. A. Smolin, "Trading classic and quantum computational resources," *Phys. Rev. X*, vol. 6, no. 2, p. 021043, 2016.
6. A. Kandala et al., "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," *Nature*, vol. 549, no. 7671, p. 242, 2017.
7. H. Karimi, G. Rosenberg, and H. G. Katzgraber, "Effective optimization using sample persistence: A case study on quantum annealers and various Monte Carlo optimization methods," *Phys. Rev. E*, vol. 96, no. 4, p. 043312, 2017.
8. A. J. McCaskey, E. F. Dumitrescu, D. Liakh, M. Chen, W. Feng, and T. S. Humble, "A language and hardware independent approach to quantum–classic computing," *Software X*, vol. 7, pp. 245–254, Jan.–June, 2018.
9. J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," *Nature Commun.*, vol. 9, Nov. 2018. [Online]. Available: https://arxiv.org/abs/1803.11173
10. J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classic algorithms," *New J. Phys.*, vol. 18, no. 2, p. 023023, 2016.
11. C. F. Negre, H. Ushijima-Mwesigwa, and S. M. Mniszewski, Detecting multiple communities using quantum annealing on the D-Wave system. 2019. [Online]. Available: https://arxiv.org/abs/1901.09756
12. D. O'Malley, V. V. Vesselinov, B. S. Alexandrov, and L. B. Alexandrov, "Nonnegative/binary matrix factorization with a D-Wave quantum annealer," *PLOS One*, vol. 13, no. 12, p. e0206653, 2018.
13. O. D. Parekh, C. Ryan-Anderson, and S. Gharibian, *Quantum Optimization Approximation Algorithms*, vol. 1. Albuquerque, NM: Sandia National Lab, 2019. [Online]. Available: https://doi.org/10.2172/1492737
14. J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, 2018. doi: 10.22331/q-2018-08-06-79.
15. I. Rivalta, M. M. Sultan, N.-S. Lee, G. A. Manley, J. Patrick Loria, and V. S. Batista, "Allosteric pathways in imidazole glycerol phosphate synthase," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 109, no. 22, pp. E1428–E1436, 2012.
16. J. Romero, J. P. Olson, and A. Aspuru-Guzik, "Quantum autoencoders for efficient compression of quantum data," *Quantum Sci. Technol.*, vol. 2, no. 4, p. 045001, 2017.
17. G. Rosenberg, M. Vazifeh, B. Woods, and E. Haber, "Building an iterative heuristic solver for a quantum annealer," *Comput. Optim. Appl.*, vol. 65, no. 3, pp. 845–869, 2016.
18. R. Shaydulin, H. Ushijima-Mwesigwa, I. Safro, S. Mniszewski, and Y. Alexeev, "Community detection across emerging quantum architectures," in *Proc. 3rd Int. Workshop on Post Moore's Era Supercomputing*, 2018, pp. 12–14.
19. R. Shaydulin, H. Ushijima-Mwesigwa, I. Safro, S. Mniszewski, and Y. Alexeev, Network community detection on small quantum computers. 2018. [Online]. Available: https://arxiv.org/abs/1810.12484
20. H. Ushijima-Mwesigwa, C. F. Negre, and S. M. Mniszewski. "Graph partitioning using quantum annealing on the D-Wave system," in *Proc. 2nd Int. Workshop on Post Moore's Era Supercomputing*, 2017, pp. 22–29.