Hybrid Quantum-Classical Problem Solving in the NISQ Era

Prashanti Priya Angara
pangara@uvic.ca
University of Victoria
Victoria, British Columbia, Canada

Ulrike Stege
ustege@uvic.ca
University of Victoria
Victoria, British Columbia, Canada

Hausi A. Müller
hausi@uvic.ca
University of Victoria
Victoria, British Columbia, Canada

Mehdi Bozzo-Rey
mbozzore@ca.ibm.com
IBM Canada
Markham, Ontario, Canada

ABSTRACT

Quantum computing has evolved from a field of scientific research to a quantum technology industry. Much progress is still needed to solve real-world problems with quantum technology and achieve quantum advantage. Industries, governments, and universities are experimenting with advanced quantum computing technologies to become quantum ready. One way forward is to combine quantum and classical approaches to form hybrid models, algorithms, and architectures to overcome the limitations of NISQ systems for near-term quantum computations. Many algorithm design, data management, and software engineering challenges have to be addressed for practical hybrid development including problem decomposition, variational circuit design, tool integration, and debugging. This paper presents algorithmic patterns and software infrastructures appearing in the literature for practical hybrid quantum problem solving and software development for selected application domains. Hybrid approaches provide significant opportunities for HPC centers and application developers to tackle hard problems that have been considered intractable using merely classical approaches. The most promising hybrid algorithms and architectures provide an excellent avenue for developers to scale quantum applications gradually and for educators to train the workforce incrementally.

CCS CONCEPTS

• **Computer systems organization → Quantum computing**.

KEYWORDS

Quantum computing, hybrid quantum-classical toolkits, variational algorithms and circuits, QPU, HPC

ACM Reference Format:
Prashanti Priya Angara, Ulrike Stege, Hausi A. Müller, and Mehdi Bozzo-Rey. 2020. Hybrid Quantum-Classical Problem Solving in the NISQ Era . In *Proceedings of ACM Conference (CASCON 2020).* IBM Corp., Riverton, NJ, USA.

1 INTRODUCTION

Many scientists and engineers consider quantum computing as one of the most interesting and challenging topics with enormous potential in many different application areas. The field of quantum computing is not new. Physicists argue it started in the early 1920s when the term quantum mechanics was coined. In 1981, Richard Feynman encouraged the physics community to build a quantum computer with his famous quote: "Nature isn't classical, and … if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly, it's a wonderful problem, because it doesn't look so easy." After delivering his famous lectures on *Simulating Physics with Computers* [27], the field began in earnest with quantum information theory, computing models, and algorithms [54]—culminating in seminal results, including Shor's 1994 factoring algorithm with exponential speedup [68], and Grover's search (1996) with quadratic speedup [30] over classical algorithms.

Over the last 15 years, companies such as IBM, D-Wave, Microsoft, Google, Honeywell, Fujitsu, Rigetti, Xanadu, and many startups have engineered real quantum computers and developed *quantum development kits (QDKs),* such as IBM Qiskit & Aqua, D-Wave Leap & Ocean, Microsoft Q# & QDK, Google Cirq & OpenFermion, Rigetti Forest & Quil, Xanadu's Strawberry Fields & PennyLane, as well as application platforms, such as 1QBit's OpenQEMIST, to program these innovative systems effectively. Thus, in recent years, the field of quantum computing has transitioned into a technology industry [40].

We are now in the noisy intermediate-scale quantum (NISQ) era where quantum computers have 50-200 qubits and their noise places serious limitations on their capabilities [59]. Researchers are investigating innovative ways to solve valuable problems using available NISQ systems and to achieve quantum advantage by demonstrating a significant performance advantage over today's classical computers [10, 61].

Over the last decade, scientists and engineers have developed effective hybrid algorithms and architectures by integrating classical and quantum processing units (QPUs) [14, 51]. Hybrid approaches can be applied at problem-solving and design time (e.g., problem decomposition and algorithm design techniques) as well as implementation and run-time (e.g., parameterizations and variational circuits). Researchers have begun realizing Feynman's dream by decomposing nature problems—simulating molecules using variational algorithms for drug design might be the early killer application of quantum computing [52, 58]. While hybrid algorithms and

platforms may just be the best first step, it is reasonable to assume that quantum applications will always be hybrid [42].

Large industry and government investments are pushing for breakthroughs in qubit counts and fidelities, with quantum advantage being a much-sought-after milestone [49]. Today, quantum computers and simulators are readily accessible and programmable over the internet in several *quantum clouds* [14, 40] and can now be explored by everybody. As a result, the demand for quantum computing education, training, and workforce development is increasing continuously [14].

The combination of these efforts has created significant opportunities for HPC centers and application developers alike, to tackle hard problems that are considered intractable using classical approaches. Moreover, hybrid approaches provide an excellent avenue for developers to grow and scale quantum applications gradually and for educators to train the quantum workforce incrementally.

Three problem types are at the core of many quantum applications: *simulation, machine learning,* and *optimization* [71]. Hybrid quantum-classical algorithms and full-stack quantum platforms are being developed for all three problem types. This paper characterizes the current *hybrid quantum toolkit* available to quantum scientists and engineers, including core building blocks, selected algorithm design patterns, and platform considerations for hybrid quantum-classical problem-solving in the NISQ era. Lessons we can learn in the short-term experimenting with hybrid quantum-classical computing will be eminently useful in the long-term because we expect that quantum computing will scale and remain hybrid.

Section 2 presents core building blocks for hybrid computing and describes effective techniques of the hybrid toolkit prominently featured in the literature. Section 3 outlines hybrid algorithmic approaches to optimization and decision problems. Section 4 highlights algorithms and architectural patterns in quantum machine learning problems. Section 5 discusses the notion of hybrid variational patterns, which are at the core of hybrid algorithms for quantum optimization, machine learning, and simulation problems. Finally, Section 6 concludes with a call for quantum software engineering skills.

2 THE HYBRID QUANTUM TOOLKIT

In Chapter 3 of the famous *Consensus Study Report* on Quantum Computing [32], the term *basic quantum building block* refers to quantum algorithms with exponentially better performance than classical algorithms. Here we expand the term to *hybrid quantum building blocks* (or *hybrid toolkit* for short) to include quantum algorithms that solve problems that lend themselves as sub-problems frequently used in a larger context, and provide an asymptotic speedup to its classical counterparts in general, as opposed to exponential speed-up only. A number of these hybrid quantum building blocks—core library routines—are useful for solving many (quantum) problems.

Building blocks are elements of the hybrid toolkit that also includes algorithmic and architectural design patterns. Examples of such powerful building blocks are *quantum phase estimation (QPE), Grover's search, variational quantum eigensolver (VQE), variational quantum factoring (VQF)* [4], and linear equation solvers [33, 46]. Famous techniques in the hybrid quantum toolkit include *quantum approximate optimization algorithms (QAOA)* [25], hybrid variants of classical algorithm-design techniques or design patterns (e.g., [3, 23]), *hybrid variational algorithms* (Section 5), and the *quantum support vector machine (QSVM)* [60]. Over the past decade, QPE, VQE, and QAOA have emerged as the most significant near-term building blocks and techniques for NISQ systems. [70].

Quantum Phase Estimation. QPE is the workhorse behind many quantum algorithms [55]. QPE, which is enabled by the *quantum Fourier transform (QFT)* [13], approximates the eigenvalue associated with a given eigenvector of a unitary operator. The long coherence times needed for QPE can be mitigated through a hybrid approach. The QPU is used as a state preparation and measurement routine while the CPU processes the measurement updates from the quantum circuit. QPE has numerous applications in the quantum problem-solving realm. For example, it is a key component of the order-finding and factoring algorithms. Shor's algorithm [68] is expected to be a serious threat to popular public-key systems that we rely on to secure information [11]. The VQF algorithm [4] is a promising near-term alternative to Shor's algorithm.

Grover's Search and Variants. One of the most famous building blocks is Grover's search [30]. Because searching is such a fundamental operation, the quantum search algorithm by Grover is used as a subroutine in many different contexts. Given n unstructured elements, it finds a specific item in $O(\sqrt{n})$ time as opposed to the classical $\Theta(n)$ algorithm, a quadratic speedup. The quantum algorithm is based on a black-box transformation and phase kick-back—both are concepts that are used in other foundational quantum algorithms including the Deutsch-Jozsa algorithm [19] and QPE. Important variants of Grover's search are the quantum algorithm for finding the minimum in an unstructured set [24], quantum partial search [31], and quantum random walk algorithms [67].

Hybrid Variational Quantum Eigensolver. VQE is a hybrid algorithm that computes the lowest eigenvalue of a Hermitian matrix or a Hamiltonian H [39, 58]. VQE algorithms are based on the variational method of quantum mechanics, which states that for a given H its expectation value must be greater than or equal to the lowest possible eigenvalue. VQE approximates solutions by decomposing an exponential-sized optimization problem and executing a polynomial number of quantum sub-problems. Hamiltonian H can be expressed as a set of terms that are realized as separate quantum circuits. The expectation values of its parts are then summed up classically to compute the expectation value of H. The initial states for these circuits are selected from a set of states based on an ansatz and are generated by a parameterized circuit. The parameters for the state preparation circuit are computed in a classical optimization loop that minimizes the expectation value of H. Compared to QPE circuits, the depths of the VQE circuits are considerably smaller, which is a big advantage in the NISQ era. Progress in computational quantum chemistry and optimization has been driven largely by VQE [39]. Many VQE applications are in molecule simulation [39, 50], where the lowest possible eigenvalue represents the ground state energy of the molecule. VQE also applies to hybrid machine learning and combinatorial optimization.

Hybrid Harrow Hassidim Lloyd. Another useful quantum algorithm is the *Harrow Hassidim Lloyd* algorithm (*HHL*) that solves a system of n linear equations in $O(\log n)$ steps [33]. Solving a system of linear equations is central to many optimization problems. Aaronson outlines ways of using the HHL algorithm in machine learning [2].

Lee et al. [46] propose a hybrid variational variant of the HHL algorithm to solve linear equations on NISQ systems by optimizing the quantum circuit. While the quantum circuit for the HHL algorithm consists of a quantum phase estimation part and an ancilla quantum encoding part, followed by an inverse quantum phase estimation, the proposed hybrid algorithm first repeatedly performs QPE to obtain k-bit classical information of eigenvalues. Then, it follows up with a classical routine that analyzes measurement outcomes from the first step. Based on the analyzed data; it then determines which simpler circuit implementation of the original ancilla quantum encoding part is applicable. Finally, the algorithm performs the original HHL algorithm with the reduced ancilla quantum encoding part instead of the original one.

Infrastructure for Hybrid Approaches. Basic linear algebra subprograms (BLAS) comprises a set of tools describing subroutines for basic linear algebra such as matrix multiplication, dot product, and cross-product. Biamonte et al. [8] refer to quantum algorithms for linear algebra tasks as *qBLAS* or quantum BLAS. These subroutines are provided by quantum software libraries, such as *IBM Qiskit & Aqua* [36] or *Microsoft Q# & QDK*, which include implementations of fundamental quantum algorithms such as QPE, Grover, VQE, and HHL, and support for techniques such as *QAOA* and *QSVM*. These are eminently useful in developing practical hybrid solutions. Xanadu supports photonic computing through *Strawberry Fields* and quantum machine learning through *PennyLane* [7]. Rigetti offers *quantum cloud services (QCS)* including parametric compilation, active qubit reset for improving performance, and co-located CPUs and QPUs, which are useful for optimizing hybrid and variational algorithms [40]. D-Wave offers access to hybrid quantum annealing through the software frameworks Ocean SDK & Leap allowing developers to work on quantum and classical systems in parallel. D-Wave also provides quantum variational autoencoders (QVAEs) to accelerate machine learning tasks [42]. Honeywell and IonQ provide access to their trapped-ion computers through Microsoft Azure Quantum and Amazon Braket.

Hybrid quantum-classical computations are increasingly supported by hybrid architectures. In many ways, the development of hybrid architectures with dedicated *quantum processing units* (QPUs) is similar to the evolution of graphics hardware accelerators such as *graphical processing units* (GPUs), which are tightly integrated into HPC platforms to tackle computationally data-intensive problems. The race is on to develop hardware-agnostic software models at different levels of abstraction to program hybrid solutions effectively. One of the most prominent frameworks is eXtreme-scale ACCelerator programming (XACC) [51], an open-source, hybrid programming model developed at Oak Ridge National Laboratories (ORNL), designed to enhance classical software workflows with near-term QPUs.

3 DESIGNING HYBRID ALGORITHMS

Since the 1980s (e.g. [17]) many computational optimization and decision problems have been studied with the goal to design quantum algorithms that exhibit a significant speedup over their best classical solutions. Today, in addition to quantum algorithms for computational problems, which produce exact solutions with high probability but are not practical to run on today's NISQ systems for even moderate input sizes, there are hybrid approaches that combine quantum and classical computing. The approaches to optimization or decision problems range from heuristic hybrid algorithms that quantum annealing and are designed for special-purpose quantum computers, to the design of specific hybrid algorithms that produce solutions that are, with high probability, exact solutions.

Approximate & heuristic hybrid techniques. Quantum annealing is an optimization technique that was formulated in 1998 by Kadowaki and Nishimori [38]. It is a heuristic used for finding a global minimum of a given objective function using quantum fluctuation based computation. Adiabatic quantum computing (based on the adiabatic theorem [9]) is a form of quantum computing that relies on a particular kind of quantum annealing. D-Wave announced the first quantum annealer in 2011 [37]—a special-purpose quantum computer that, through quantum annealing, can only be used for problems defined as energy minimization. Santoro and Tosatti [62] outline applications of quantum annealing in the area of hard optimization problems, including the traveling salesperson (TSP) and the Boolean satisfiability problem. Tran et al. [72] propose a hybrid heuristic framework, which uses quantum annealing as part of a complete search when tackling scheduling optimization problems.

QAOA embodies a technique using the variational method that enables approximation algorithms for combinatorial optimization problems [25]. One of its main features is the ability to control the depth of the quantum circuits required by the algorithm, so that, in principle, one can restrict the algorithm to shallow quantum circuits while sacrificing solution quality.

Speeding up Classical Algorithms for Optimization Problems. Good candidate problems for potential speed-up are members of classes of computational problems for which no classical algorithm is known to solve the problem faster than a certain exponential time. As example serve the NP-complete problems for which exact solutions are desired. While there is no expectation to solve an NP-complete problem using a quantum algorithm in polynomial time [1], it is worthwhile investigating how to combine quantum routines with classical problem-solving strategies to speed up solutions for such problems and in turn enable faster practical implementations.

Tran et al. [72] advocate that *"an effective approach to solving complex problems is to decompose them and integrate dedicated solvers for those sub-problems."* While hybrid algorithms can be used to realize such strategy, by decomposing the problem on the CPU, and invoking quantum building blocks on the QPU, or by solving sufficiently small sub-problems directly. This approach is highly appropriate for NISQ systems. Also, classical algorithm design-techniques such as divide-and-conquer and dynamic programming make use of the idea to solve smaller sub-problems and then combining those to solutions for the problem instance. Also, subroutines such as a search or sorting are taking advantage of solving smaller sub-problems as

part of a larger algorithm. Can such ideas be realized for designing hybrid algorithms for NP-complete problems?

Since NP-complete problems typically can be solved using a brute-force search, Grover's search algorithm can be applied to achieve a quadratic speedup of the brute-force algorithm. More efficient strategies involve exact methods to solve NP-complete problems including dynamic programming [5, 35, 74], divide and conquer [3, 29], and other methods in the toolkits of exact algorithms [28] and fixed-parameter tractability [16, 21].

For a number of fixed-parameter algorithms that consist of a kernelization step (i.e., a polynomial-time pre-processing algorithm that reduces the given decision problem to a (smaller) one that has a size that depends on a given fixed-parameter instead of the original input size), followed by a brute-force Grover's search, would achieve a speedup over the classical running time. However, in many cases applying Grover's search to speed up an existing sophisticated exponential-time algorithm is difficult.

Ambainis et al. [3] describe how to apply Grover's search [30] or its variant of finding a minimum item in a set [24] to speedup particular exponential-time dynamic-programming algorithms, for example for the NP-complete problems Hamiltonian Circuit, TSP [5, 35] or Minimum Set Cover [28]. What these dynamic-programming approaches have in common is that they solve subproblems corresponding to subsets of an n-element set. Such a dynamic programming approach in its most basic form typically runs in exponential time in the size of the input. The achieved dynamic programming quantum speed-up relies on pre-computing solutions for sub-problems of a specific size (e.g., say, a quarter of the original instance size) using the classical dynamic programming algorithm, followed by searching—on the remaining subsets—for an answer to the problem using a quantum search in the pre-computed answers for the sub-problems. With their approach, they can improve upon the famous algorithm by Bellman, Held, and Karp from 1962, which is still state of the art for general TSP instances.

While in quantum dynamic programming algorithms as described by the approach above, "quantum power" is used in a *top down* manner by applying Grover's search to large sets of precomputed solutions of sub-problems, a *bottom up* approach is used in a framework of hybrid divide and conquer algorithms: Ge and Dunjko [29] describe the technique by generalizing the specific quantum algorithm [22] that solves the problem 3SAT by introducing a quantum version of Schöning's algorithm [63], and is designed specifically to work on small quantum devices. They show that applying a quantum algorithm to small sub-problems of a specific size (e.g., a quarter of the original input size) in the divide-and-conquer method yields a polynomial speed-up that is achievable using quantum computers with only a few qubits, and therefore such an algorithm can be run on NISQ systems. Note that for this general hybrid divide-and-conquer approach, a quantum algorithm solving the problem in question is necessary to be executed on sufficiently small sub-instances.

4 HYBRID QUANTUM MACHINE LEARNING

In 2001, Gupta et al. proposed a mathematical model of computation called *Quantum Neural Network (QNN)*, which is based on Deutsch's model of computational networks [18]. Since then, the fields of quantum computing and classical machine learning have exploded [8, 75]. With the advent of NISQ systems, using quantum computing for machine learning has been revisited in the past decade. Conversely, the well-established classical machine learning methods help extend and improve quantum information theory [66]. Schuld and Killoran [65] outline similarities between kernel methods in machine learning and quantum computing and lay the theoretical foundations of *quantum machine learning (QML)*. *Classical machine learning (CML)* is classified into *supervised learning*, *unsupervised learning*, and *reinforcement learning*. Quantum computing can play a role in different parts of the ML pipeline.

Recently, deep learning techniques have lead to breakthroughs in the analysis of vision, text, audio, and speech [45]. These techniques rely on vectors and tensors that are transformed from one representation to a high dimensional representation where complex functions can be learned. These computational units are continuous and are currently only approximated on classical computers. One area of research is investigating whether the continuous nature of quantum computers can be leveraged to perform computations with continuous tensors and vectors. Unlike qubit-based quantum computers, the continuous model leverages the wave-like properties of nature. Killoran et al. describe quantum neural networks that use the Continuous-Variable (CV) model [43]. Here, instead of encoding quantum states as qubits, quantum states are encoded as electromagnetic fields. Strawberry Fields [44] is a software framework that is based on the CV model of quantum computing.

Supervised Machine Learning. Supervised ML uses labeled data to identify labels for unlabeled data. Supervised ML techniques, such as classification, have largely satisfactory algorithms in classical computing [15, 48, 76]. Havlíček et al. address the limitations to classification problems that have large feature spaces, where kernel functions are computationally expensive to solve classically [34]. Their methods take advantage of exponentially large quantum states. Rebentrost et al. [60] introduce a *quantum support vector machine (QSVM)* for big-data classification that relies on exponentiation techniques for non-sparse matrices for training with a complexity that is logarithmic in feature size and the number of training data points (the complexity of the classical SVM is linear in comparison). Farhi and Neven [26] describe a generic framework for implementing *QNNs* for supervised learning with classical simulation and show that these networks can indeed be used to classify data but were restricted to about 17 qubits for simulation.

Schuld et al. [64] describe a low depth variational quantum-classical training scheme for a weak non-linear classification. They compare this against a purely classical implementation, provide the mapping of quantum gates to layers of a neural network, and introduce a quantum dropout regularization scheme. The quantum-classical implementation (using simulators) performs well on standard classical benchmark datasets, such as MNIST256 and SEMEION. However, it is prone to overfitting without regularization.

Unsupervised Machine Learning. Unsupervised ML looks for patterns in a dataset with no pre-existing labels. Perdomo-Ortiz et al. [57] describe some of the challenges and opportunities for quantum-assisted ML and argue that the promising "killer" applications in quantum computing might not come from the well-researched area of supervised ML, but rather from the areas the classical ML faces

challenges, such as unsupervised and semi-supervised learning. Otterbach et al. [56] describe a hybrid approach to solving clustering problems using QAOA. They leverage the statistical distributions that are available on quantum computers to improve performance in clustering. QAOA has also been used for generative models of ML. Verdon et al. describe such a low-depth hybrid algorithm for generative neural network learning [73]. Related research includes the use of quantum annealers for the implementation of QML, which is limited by the sparse connectivity among qubits in the physical hardware. Benedetti et al. address this challenge demonstrating the feasibility of using quantum annealers for implementing unsupervised ML models [6]. Lloyd et al. provide an overview of some hybrid paradigms for cluster assignment and cluster finding [47]. The authors describe algorithms where quantum speedups can occur in different parts of the ML pipeline especially for "big quantum data." They provide algorithms for exponential speedups (compared to their classical counterparts) in estimating distances and inner products. For unsupervised techniques, such as clustering, their clustering algorithm exhibits an exponential speedup as opposed to the classical counterpart.

Reinforcement Learning. Reinforcement learning is an ML paradigm that studies how (software) agents take actions to maximize the cumulative reward. Reinforcement learning differs from supervised and unsupervised techniques—the focus is on balancing exploration (discovering unchartered territory) and exploitation (usage of the agents' current knowledge). Using hybrid algorithms for reinforcement learning is a recent field of study, due to Dong et al. [20], who showed promising results in this area—they found a good trade-off between exploration and exploitation that speeds up the learning process. Dunjko et al. [23] describe quantum-accessible reinforcement learning for certain game playing scenarios where the games have a recursive structure and the agent can learn by playing against itself. Examples of such games include AlphaGo and AlphaGo Zero, which have been studied extensively in classical reinforcement learning [69].

5 HYBRID VARIATIONAL ALGORITHMS

Over the past decade, key quantum algorithms emerged for optimization, machine learning, and simulation problems. Popular gems include VQE [58], QAOA [25], and identifying classifiers in machine learning [64].

At the core of these algorithms are the *variational principle & method* and the notion of a *hybrid quantum-classical variational algorithm*, which together represent one of the most promising and practical algorithmic patterns or frameworks for near-term NISQ systems.

As for any other hybrid approach, *variational quantum algorithms (VQA)* involve iterating between a CPU and a QPU. The variational quantum circuit on the QPU gets parameters as inputs from the CPU and delivers expectation values as outputs back to the CPU. A classical optimization loop, minimizing a cost function and executing on the CPU, drives the hybrid algorithm by repeatedly feeding revised parameters to the variational quantum circuit.

VQAs have been proposed for many different applications [12], including computational quantum chemistry and molecule simulation [39, 50], computationally hard combinatorial optimization problems [41, 72], such as MaxCut, Scheduling, TSP, and Knapsack, linear algebra problems [77] including solving systems of linear equations [46], integer factoring [4], and machine learning [64].

6 CONCLUSIONS

Now that a quantum technology industry has emerged in the NISQ era with QPUs, QDKs, algorithmic toolkits including practical variational algorithms and circuits for the near-term, it is imperative that we also focus on quantum software engineering before being overwhelmed with a quantum software crisis as we faced for classical computing in 1968 when the term software engineering was coined [53].

In their inspiring 2017 paper, Rigetti engineers [78] argued convincingly that *"we need a new breed of quantum programmer to study and implement quantum software—with a skillset between that of a quantum information theorist and a software engineer."* Scientist and engineers, who implement hybrid quantum-classical toolkits, algorithms, frameworks, architectures, and platforms, need not only quantum information skills but also software engineering and domain-specific application skills.

This paper aimed to outline the landscape of hybrid quantum-classical problem solving for NISQ systems. After presenting fundamental building blocks for hybrid problem solving and algorithms, we described hybrid approaches for solving quantum optimization, simulation, and machine learning problems. The most promising and practical avenue in the quest to achieve quantum advantage in the NISQ era is called *hybrid quantum-classical variational algorithms and circuits*. In summary, this paper provides solid starting points for researchers interested in experimenting with practical hybrid quantum-classical problem-solving approaches, especially variational ones, for different application domains. Software engineering skills are critical for building and generating the infrastructure for highly scaleable hybrid quantum-classical ecosystems.

REFERENCES

[1] S. Aaronson. 2013. *Quantum computing since Democritus*. Cambridge Univ. Press.
[2] S. Aaronson. 2015. Read the fine print. *Nature Phys.* 11, 4 (2015), 291–293.
[3] A. Ambainis, K. Balodis, J. Iraids, M. Kokainis, K. Prūsis, and J. Vihrovs. 2019. Quantum Speedups for Exponential-Time Dynamic Programming Algor.. In *Proc. 30th Ann. ACM-SIAM Symp. on Discr. Algor. (SODA)*. ACM, 1783–1793.
[4] E. R. Anschuetz, J. P. Olson, A. Aspuru-Guzik, and Y. Cao. 2018. Variational Quantum Factoring. arXiv:1808.08927
[5] R. Bellman. 1962. Dynamic Programming Treatment of the Travelling Salesman Problem. *J. ACM* 9, 1 (1962), 61–63.
[6] M. Benedetti, J. Realpe-Gómez, R. Biswas, and A. Perdomo-Ortiz. 2017. Quantum-assisted learning of hardware-embedded probabilistic graphical models. *Phys. Rev. X* 7, 4 (2017), 52–54.
[7] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, M. S. Alam, S. Ahmed, J. M. Arrazola, C. Blank, A. Delgado, S. Jahangiri, K. McKiernan, J. J. Meyer, Z. Niu, A. Száva, and N. Killoran. 2018. PennyLane: Automatic differentiation of hybrid quantum-classical computations. arXiv:1811.04968
[8] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd. 2017. Quantum Mach. Learn. *Nature* 549, 7671 (2017), 195–202.
[9] M. Born and V. Fock. 1928. Beweis des Adiabatensatzes. *Matrix* 6 (1928).
[10] S. Bravyi, D. Gosset, and R. König. 2018. Quantum advantage with shallow circuits. *Science* 362, 6412 (2018), 308–311.
[11] W. Buchanan and A. Woodward. 2017. Will quantum computers be the end of public key encryption? *J. Cyber Sec. Techn.* 1, 1 (2017), 1–22.
[12] M. Cerezo, K. Sharma, A. Arrasmith, and P. J. Coles. 2020. Variational Quantum State Eigensolver. arXiv:2004.01372
[13] D. Coppersmith. 2002. An approximate Fourier transform useful in quantum factoring. arXiv:0201067
[14] A. D. Corcoles, A. Kandala, A. Javadi-Abhari, D. T. McClure, A. W. Cross, K. Temme, P. D. Nation, M. Steffen, and J. M. Gambetta. 2019. Challenges and

Opportunities of Near-Term Quantum Comp. Systems. *Proc. of the IEEE* (2019), 1–15.
[15] C. Cortes and V. Vapnik. 1995. Support-vector networks. *Machine learning* 20, 3 (1995), 273–297.
[16] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. 2015. *Parameterized Algorithms*. Springer.
[17] D. Deutsch. 1985. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proc. R. Soc. Lond.* 400 (1985), 97–117.
[18] D. Deutsch. 1989. Quantum computational networks. *Proc. R. Soc. Lond.* 425 (1989), 73–90.
[19] D. Deutsch and R. Jozsa. 1992. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond.* 439 (1992), 553–558.
[20] D. Dong, C. Chen, H. Li, and T. J. Tarn. 2008. Quantum reinforcement learning. *IEEE Trans. Systems, Man, and Cybern.* 38, 5 (2008), 1207–1220.
[21] R. Downey, M. Fellows, and U. Stege. 1998. Parameterized Complexity: A Framework for Systematically Confronting Computational Intractability. *DIMACS Ser. in Discr. Mathem. and TCS* 49 (1998).
[22] V. Dunjko, Y. Ge, and J. I. Cirac. 2018. Computational Speedups Using Small Quantum Devices. *Phys. Rev. Lett.* 121, 25 (2018).
[23] V. Dunjko, Y.-K. Liu, X. Wu, and J. M. Taylor. 2017. Exponential improvements for quantum-accessible reinforcement learning. (2017). arXiv:1710.11160
[24] C. Dürr and P. Høyer. 1996. A Quantum Algorithm for Finding the Minimum. *CoRR* 9607014 (1996).
[25] E. Farhi, J. Goldstone, and S. Gutmann. 2014. A Quantum Approximate Optimization Algorithm. arXiv:1411.4028
[26] E. Farhi and H. Neven. 2018. Classification with quantum neural networks on near perm processors. (2018). arXiv:1802.06002
[27] R. P. Feynman. 1982. Simulating physics with computers. *Int. J. Theor. Phys.* 21, 6–7 (1982), 467–488.
[28] F. V. Fomin and D. Kratsch. 2010. *Exact Exponential Algorithms*. Springer.
[29] Y. Ge and V. Dunjko. 2020. A hybrid algorithm framework for quantum computers with application to finding Hamiltonian cycles. *J. Math. Phys.* 61 (2020).
[30] L. K. Grover. 1996. A fast quantum mechanical algorithm for database search. In *Proc. 28th Ann. ACM Symp. on Theory of Computing (STOC 1996)*. 212–219.
[31] L. K. Grover and J. Radhakrishnan. 2005. Is Partial Quantum Search of a Database Any Easier?. In *Proc. 17th Ann. ACM Symp. Parallel. in Alg. and Arch.* 186–194.
[32] E. Grumbling and M. Horowitz (Eds.). 2019. *Quantum Computing: Progress and Prospects (2019)*. National Academies of Sciences, Engineering, and Medicine. The National Academies Press. https://doi.org/10.17226/25196
[33] A. W. Harrow, A. Hassidim, and S. Lloyd. 2009. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* 103, 150502 (2009).
[34] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta. 2019. Supervised learning with quantum-enhanced feature spaces. *Nature* 567, 7747 (2019), 209–212.
[35] M. Held and R. M. Karp. 1961. A Dynamic Programming Approach to Sequencing Problems. In *Proc. 16th ACM National Meeting*. 201–204.
[36] IBM Quantum Experience. [n.d.]. Qiskit Aqua: Algorithms for quantum computing applications. https://qiskit.org/aqua/
[37] M. W. Johnson, M. H. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk, E. M. Chapple, C. Enderud, J. P. Hilton, K. Karimi, E. Ladizinsky, N. Ladizinsky, T. Oh, I. Perminov, C. Rich, M. C. Thom, E. Tolkacheva, C. J. Truncik, S. Uchaikin, J. Wang, B. Wilson, and G. Rose. 2011. Quantum annealing with manufactured spins. *Nature* 473, 7346 (2011), 194–198.
[38] T. Kadowaki and H. Nishimori. 1998. Quantum annealing in the transverse Ising model. *Phys. Rev. E* 58, 5 (1998), 5355–5363.
[39] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta. 2017. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature* 549, 7671 (Sep 2017), 242–246.
[40] P. J. Karalekas, N. A. Tezak, E. C. Peterson, C. A. Ryan, M. P. da Silva, and R. S. Smith. 2020. A quantum-classical cloud platform optimized for variational hybrid algorithms. *Quantum Sci. Techn.* 5, 2 (2020), 024003.
[41] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev, and P. Balaprakash. 2019. Learning to optimize variational quantum circuits to solve combinatorial problems. arXiv:1911.11071
[42] A. Khoshaman, W. Vinci, B. Denis, E. Andriyash, H. Sadeghi, and M. H. Amin. 2018. Quantum variational autoencoder. *Quantum Sci. Techn.* 4, 1 (2018), 014001.
[43] N. Killoran, T. Bromley, J. Arrazola, M. Schuld, N. Quesada, and S. Lloyd. 2019. Continuous-variable quantum neural networks. *Phys. Rev. Res.* 1, 3 (2019), 1–21.
[44] N. Killoran, J. Izaac, N. Quesada, V. Bergholm, M. Amy, and C. Weedbrook. 2019. Strawberry Fields: A Software platform for photonic quantum computing. *Quantum* 3 (2019), 129.
[45] Y. Lecun, Y. Bengio, and G. Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.
[46] Y. Lee, J. Joo, and S. Lee. 2019. Hybrid quantum linear equation algorithm and its experimental test on IBM Quantum Experience. *Nature Scient. Reports* 9 (2019).
[47] S. Lloyd, M. Mohseni, and P. Rebentrost. 2013. Quantum algorithms for supervised and unsupervised machine learning. arXiv:1307.0411
[48] M. E. Maron. 1961. Automatic indexing: An experimental inquiry. *J. ACM* 8, 3 (1961), 404–417.
[49] M. Martonosi and M. Roetteler. 2019. *Next Steps in Quantum Computing: Computer Science's Role*. Technical Report. Computing Community Consortium (CCC).
[50] S. McArdle, S. Endo, A. Aspuru-Guzik, S. Benjamin, and X. Yuan. 2020. Quantum computational chemistry. *Rev. Mod. Phys.* 92, 1 (2020), 015003.
[51] A. J. McCaskey, D. I. Lyakh, E. F. Dumitrescu, S. S. Powers, and T. S. Humble. 2019. XACC: A System-Level Software Infrastructure for Heterogeneous Quantum-Classical Computing. *Quantum Sci. Techn.* 5, 2 (2019), 024002.
[52] A. J. McCaskey, Z. P. Parks, J. Jakowski, S. V. Moore, T. D. Morris, T. S. Humble, and R. C. Pooser. 2019. Quantum chemistry as a benchmark for near-term quantum computers. *npj Quantum Inf.* 5 (2019), 99.
[53] P. Naur and B. Randell. 1968. Software Engineering: A Report on a Conference Sponsored by the NATO Science Committee.
[54] M. A. Nielsen and I. L. Chuang. 2011. *Quantum Computation and Quantum Information*. Cambridge University Press.
[55] T. E. O'Brien, B. Tarasinski, and B. M. Terhal. 2019. Quantum phase estimation of multiple eigenvalues for small-scale (noisy) experiments. *New J. Phys.* 21, 2 (2019), 023022:1–28.
[56] J. S. Otterbach, R. Manenti, N. Alidoust, A. Bestwick, M. Block, B. Bloom, S. Caldwell, N. Didier, E. S. Fried, S. Hong, P. Karalekas, C. B. Osborn, A. Papageorge, E. C. Peterson, G. Prawiroatmodjo, N. Rubin, C. A. Ryan, D. Scarabelli, M. Scheer, E. A. Sete, P. Sivarajah, R. S. Smith, A. Staley, N. Tezak, W. J. Zeng, A. Hudson, B. R. Johnson, M. Reagor, M. P. da Silva, and C. Rigetti. 2017. Unsupervised machine learning on a hybrid quantum computer. (2017). arXiv:1712.05771
[57] A. Perdomo-Ortiz, M. Benedetti, J. Realpe-Gómez, and R. Biswas. 2018. Opportunities and challenges for quantum-assisted machine learning in near-term quantum computers. *Quantum Sci. Technol.* 3, 3 (2018), 30502.
[58] A. Peruzzo, J. McClean, P. Shadbolt, M. H. Yung, X. Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien. 2014. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications* 5 (2014).
[59] J. Preskill. 2018. Quantum Computing in the NISQ era and beyond. *Quantum* 2 (2018), 79.
[60] P. Rebentrost, M. Mohseni, and S. Lloyd. 2014. Quantum support vector machine for big data classification. *Phys. Rev. Lett.* 113, 3 (2014), 1–5. arXiv:1307.0471
[61] D. Ristè, M. Silva, C. Ryan, A. Cross, J. Smolin, J. Gambetta, J. Chow, and B. Johnson. 2017. Demonstration of quantum advantage in machine learning. *npj Quantum Information* 3 (2017), 1–5.
[62] G. E. Santoro and E. Tosatti. 2006. Optimization using quantum mechanics: Quantum annealing through adiabatic evolution. *J. Phys.* 39, 36 (2006).
[63] U. Schöning. 1999. A Probabilistic Algorithm for K-SAT and Constraint Satisfaction Problems. In *Proc. 40th Ann. Symp. Found. of Comp. Sci. (FOCS)*. 410.
[64] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe. 2020. Circuit-centric quantum classifiers. *Phys. Rev. A* 101, 3 (2020).
[65] M. Schuld and N. Killoran. 2019. Quantum machine learning in feature Hilbert spaces. *Phys. Rev. Lett.* 122, 4 (2019), 40504:1–12.
[66] M. Schuld, I. Sinayskiy, and F. Petruccione. 2015. An introduction to quantum machine learning. *Contemp. Phys.* 56, 2 (2015), 172–185.
[67] N. Shenvi, J. Kempe, and K. B. Whaley. 2003. Quantum random-walk search algorithm. *Phys. Rev. A* 67, 5 (2003), 052307:1–7.
[68] P. W. Shor. 1994. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proc. 35th Ann. Symp. Found. of Comp. Sci. (FOCS)*. 124–134.
[69] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. Van Den Driessche, T. Graepel, and D. Hassabis. 2017. Mastering the game of Go without human knowledge. *Nature* 550, 7676 (2017), 354–359.
[70] M. Stechly. 2019. Variational Quantum Eigensolver Explained. https://www.mustythoughts.com/variational-quantum-eigensolver-explained
[71] R. Sutor, T. Hickey, and L. Feller. 2018. Taking the quantum leap. https://www.ibm.com/thought-leadership/institute-business-value/report/quantumleap
[72] T. T. Tran, M. Do, E. G. Rieffel, J. Frank, Z. Wang, B. O'Gorman, D. Venturelli, and J. C. Beck. 2016. A Hybrid Quantum-Classical Approach to Solving Scheduling Problems. In *Proc. 9th Int. Symp. Comb. Search (SoCS)*. 1–9.
[73] G. Verdon, M. Broughton, and J. Biamonte. 2017. A quantum algorithm to train neural networks using low-depth circuits. (2017). arXiv:1712.05304
[74] X. Wang and J. Tian. 2011. Dynamic Programming for NP-Hard Problems. *Procedia Engineering* 15 (2011), 3396–3400.
[75] P. Wittek. 2014. *Quantum machine learning: What quantum computing means to data mining*. Academic Press.
[76] X. Wu, V. Kumar, Q. J. Ross, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z. H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg. 2008. Top 10 algorithms in data mining. *Knowledge and Inf. Syst.* 14, 1 (2008), 1–37.
[77] X. Xu, J. Sun, S. Endo, Y. Li, S. C. Benjamin, and X. Yuan. 2019. Variational algorithms for linear algebra. arXiv:1909.03898
[78] W. Zeng, B. Johnson, R. Smith, N. Rubin, M. Reagor, C. Ryan, and C. Rigetti. 2017. First quantum computers need smart software. *Nature* 549 (09 2017), 149–151.