Comparing the hardness of MAX 2-SAT problem instances for quantum and classical algorithms

Puya Mirkarimi,[1,*] Adam Callison,[2,3] Lewis Light,[1] Nicholas Chancellor,[1] and Viv Kendon[1,4]

[1]*Department of Physics, Durham University, Durham DH1 3LE, United Kingdom*
[2]*Blackett Laboratory, Imperial College London, London SW7 2BW, United Kingdom*
[3]*Department of Physics and Astronomy, University College London, London WC1E 6BT, United Kingdom*
[4]*Department of Physics, SUPA and University of Strathclyde, Glasgow G4 0NG, United Kingdom*

(Received 8 September 2022; accepted 17 April 2023; published 5 June 2023)

An algorithm for a particular problem may find some instances of the problem easier and others harder to solve, even for a fixed input size. We numerically analyze the relative hardness of MAX 2-SAT problem instances for various continuous-time quantum algorithms and a comparable classical algorithm. This has two motivations: To investigate whether small-sized problem instances, which are commonly used in numerical simulations of quantum algorithms for benchmarking purposes, are a good representation of larger instances in terms of their hardness to solve, and to determine the applicability of continuous-time quantum algorithms in a portfolio approach, where we take advantage of the variation in the hardness of instances between different algorithms by running them in parallel. We find that, while there are correlations in instance hardness between all of the algorithms considered, they appear weak enough that a portfolio approach would likely be desirable in practice. Our results also show a widening range of hardness of randomly generated instances as the problem size is increased, which demonstrates both the difference in the distribution of hardness at small sizes and the value of a portfolio approach that can reduce the number of extremely hard instances. We identify specific weaknesses of these quantum algorithms that can be overcome with a portfolio approach, such their inability to efficiently solve satisfiable instances (which is easy classically).

DOI: 10.1103/PhysRevResearch.5.023151

I. INTRODUCTION

Are the small-sized problem instances typically used for numerical simulations actually difficult enough to solve to provide a useful test of quantum algorithms? We investigate this question in the setting of continuous-time quantum computing (adiabatic quantum computing, quantum walks, and quantum annealing in particular) used to solve hard optimization problems. Here, the word "difficult" refers to the amount of computing resources used to solve one particular instance of a problem, and "hard" refers to the scaling of the computational complexity with respect to input size. Not all instances of hard problems are actually difficult to solve, even when they belong to a problem class that is NP-hard. Complexity classes are concerned with the asymptotic behavior of computational complexity as a function of input size, and even for uniformly hard problem classes, there are instances that are much less difficult to solve than the others, although they may form a vanishingly small subset in the large size limit. However, for the small sizes we have to use for numerical simulations, the less difficult instances could form a significant fraction of the instances being processed, and this could significantly skew the results of the simulations.

Despite the above caveats, prior work by some of the authors [1] found surprisingly good scaling for as few as five qubit spin glass ground state problems, using quantum walk computation. This contrasts with the search problem [2,3], where finite size effects are apparent up to 20 or so qubit problem sizes [4] for all continuous-time quantum computing methods. Crosson *et al.* [5] identified 20 qubit sized instances of MAX 2-SAT that are difficult for quantum annealing (low success probabilities at anneal time $t_f = 100$). However, this does not guarantee that these instances are also difficult for classical algorithms, or for other continuous-time quantum computing methods, such as quantum walk computation.

If different problem instances are more or less difficult for different algorithms, the best solution method can be a hybrid approach. Portfolio solvers for the Boolean satisfiability problem (SAT) are an example of this. These solvers take a set of different core solvers or different configurations of the same core solver, called a portfolio, and they run the solvers in parallel on different computing cores. Heuristics and machine learning techniques can be used to determine smart resource allocations that assign more computing cores to solvers that are likely to perform better, based on the features of an instance. Portfolio-based SAT solving was introduced in 2008 with ManySAT [6] and solvers using the portfolio approach have outperformed all other solvers in the parallel track of recent SAT competitions [7].

*puya.mirkarimi@durham.ac.uk

In this work, we consider the difficulty of the MAX 2-SAT instances from Ref. [5], which are difficult for coherent quantum annealing, for other quantum and classical algorithms. We compare these difficult instances with typical MAX 2-SAT instances, and also compare the performance of a good classical algorithm. We then evaluate parallel approaches that combine two quantum algorithms and approaches that combine a quantum algorithm and a classical algorithm, to identify portfolio-based strategies that could outperform a single algorithm used alone.

The paper is organized as follows. In Sec. II, we give an introduction to methods in continuous-time quantum computing and the MAX 2-SAT problem. In Sec. III, we outline the datasets and methods we have used in our numerical analysis. Then, we present the results of our work in Sec. IV, where we make comparisons between the difficulty of instances for different algorithms and study the behavior of these algorithms on satisfiable instances, which are classically easy to solve. Finally, we give an overview of the results and present our conclusions in Sec. V.

II. BACKGROUND

In this section, we introduce the definitions and concepts used in this work. This includes an overview of continuous-time quantum computing in the coherent regime, a description of the MAX 2-SAT problem, and a mapping of MAX 2-SAT to a problem Hamiltonian. Definitions given in this section will be used when quantifying the success of algorithms in later sections. The contents of this section are not new work. Rather, this section is intended to be a brief review of key concepts and prior work for readers that are new to this literature.

A. Continuous-time quantum computing

Continuous-time quantum computing is a model for computing that offers an intuitive approach to solving combinatorial optimization problems on quantum hardware. In this approach, the problem is encoded in a problem Hamiltonian H_{problem} such that the ground state of H_{problem} corresponds to the desired solution. The computation is performed by initialising a set of qubits to a state $|\psi(0)\rangle$, applying a time-dependent Hamiltonian, and measuring the final state of the qubits after a time t_f has passed. Assuming that the system stays in a fully coherent regime for the full length of the computation, the evolution of the system between the initialization and measurement steps can be described by the Schrödinger equation. Typically, the Hamiltonian is expressed in the form

$$H(t) = A(t)H_{\text{driver}} + B(t)H_{\text{problem}}, \tag{1}$$

where $A(t)$ and $B(t)$ are the control functions, which are generally time-dependent real numbers, and H_{driver} is a Hamiltonian that drives state transitions.

The continuous-time quantum walk (QW) [8] is a form of fully coherent continuous-time quantum computing where the control functions are time-independent. The initial state is chosen to be the ground state of H_{driver}, which is known in advance and is easy to prepare. The QW Hamiltonian is given by

$$H_{\text{QW}}(\gamma) = \gamma H_{\text{driver}} + H_{\text{problem}}, \tag{2}$$

where we set $B(t) = 1$ and $\gamma = A(t)$ is called the hopping rate. QW is a quantum analog of the classical continuous-time random walk, which is a stochastic process that describes the path of a walker as it takes random steps on a mathematical space. The hopping rate γ can be interpreted as the probability per unit time that the walker will move to an adjacent site. For discussions of the connection between QW and other forms of continuous-time quantum computing, see Refs. [1,4,9].

Another coherent form of continuous-time quantum computing is adiabatic quantum computing (AQC) [10]. In AQC, the system is prepared in the ground state of H_{driver}, and the Hamiltonian is slowly varied from H_{driver} to H_{problem} by varying the control functions from $A(0) = 1$ and $B(0) = 0$ to $A(t_f) = 0$ and $B(t_f) = 1$. If $A(t)$ and $B(t)$ are smoothly and monotonically decreasing and increasing respectively and the minimum gap between the energy of the ground state and the first excited state is nonzero, then the adiabatic theorem [11] ensures that the system will have a high probability of staying in the instantaneous ground state throughout the computation, provided that t_f is long enough. In AQC, t_f is always long enough to be close to the adiabatic limit. We refer to protocols with time-dependent control functions where coherence and/or adiabaticity are not guaranteed as quantum annealing (QA). Note that AQC and QA are sometimes defined differently elsewhere in the literature.

In practice, the problem Hamiltonian is typically expressed as an Ising Hamiltonian on n qubits, which takes the form

$$H_{\text{problem}} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} J_{ij}\sigma_i^z\sigma_j^z + \sum_{i=1}^{n} h_i\sigma_i^z, \tag{3}$$

where the couplings $J_{ij} \in \mathbb{R}$ and field strengths $h_i \in \mathbb{R}$ are used to encode the problem, and $\sigma_i^z = \mathbb{1}_2^{\otimes i-1} \otimes \sigma_z \otimes \mathbb{1}_2^{\otimes n-i}$ is the Pauli operator σ_z acting on qubit i and identities acting on all other qubits. A common choice for the driver Hamiltonian in both QW and AQC is the transverse-field Hamiltonian

$$H_{\text{driver}} = -\sum_{i=1}^{n} \sigma_i^x, \tag{4}$$

where σ_i^x is defined similarly to σ_i^z as $\sigma_i^x = \mathbb{1}_2^{\otimes i-1} \otimes \sigma_x \otimes \mathbb{1}_2^{\otimes n-i}$. The ground state of this driver Hamiltonian, which in QW and AQC is the initial state of the system, is the equal superposition of the computational basis states,

$$|\psi(0)\rangle = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} |j\rangle = |+\rangle^{\otimes n}, \tag{5}$$

where $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$.

Whereas in AQC the adiabatic theorem guarantees that one can always attain a near-unity probability of successfully measuring the state corresponding to the optimal solution by using a long enough run time, this is not the case for QW. Therefore, to find the optimal solution with a probability of close to unity using QW, one should perform the computation many times and take the best found solution. For simplicity, in

the rest of this work we will only be considering problem instances that have a unique optimal solution corresponding to a nondegenerate ground state $|\psi_G\rangle$ of the problem Hamiltonian. According to the Born rule [12], the probability of measuring the state $|\psi_G\rangle$ after evolving the system for a time t_f is

$$P(t_f) = |\langle\psi_G \,|\, \psi(t_f)\rangle|^2. \tag{6}$$

In general, the success probability $P(t)$ for QW fluctuates with time in an unpredictable manner. To avoid taking every measurement at a time when $P(t_f)$ happens to be near a local minimum, it is beneficial to use different values of t_f for each measurement. We will consider the same approach as in [1], where t_f is selected uniformly at random from an interval $I = [t_I, t_I + \Delta t_I]$. The average single run success probability is defined as

$$\overline{P}(t_I, \Delta t_I) \equiv \frac{1}{\Delta t_I} \int_{t_I}^{t_I+\Delta t_I} P(t_f)\, dt_f, \tag{7}$$

which is the mean success probability of individual measurements in this approach. The number of repeats required to attain an arbitrarily high probability of measuring the ground state scales as the inverse of $\overline{P}(t_I, \Delta t_I)$ for small $\overline{P}(t_I, \Delta t_I)$.

For both QW and AQC, the choice of control functions is a free parameter that can affect the performance of the algorithms. In QW, this corresponds to the hopping rate γ. A good choice for γ is one that balances the energy between H_{driver} and H_{problem} in the total Hamiltonian [1]. To achieve this, we would like to set γ such that the energy-spread of γH_{driver} is equal to the energy-spread of H_{problem}. For the transverse-field driver Hamiltonian defined in Eq. (4), the energy-spread is $2n$. However, the energy-spread of the problem Hamiltonian, which is the difference between the maximum and minimum number of clauses that can be satisfied, depends on the particular problem instance and it is not possible to calculate it without solving the instance. Therefore we instead consider the average energy-spread of H_{problem} for instances of a given number of variables n, and we calculate a heuristic hopping rate γ_{heur} by setting this equal to the energy-spread of γH_{driver}, giving

$$\gamma_{\text{heur}} = \frac{\langle E_{2^n} - E_1\rangle}{2n}. \tag{8}$$

Here, E_1 and E_{2^n} are the smallest and largest eigenvalues of H_{problem} respectively. For our analysis, the average energy-spread $\langle E_{2^n} - E_1\rangle$ was calculated for each n by diagonalising the problem Hamiltonians of the generated instances (i.e., solving the problems). In practice, when the size of the instances makes this approach too computationally expensive, $\langle E_{2^n} - E_1\rangle$ can be calculated for similar instances with fewer variables and extrapolated to larger n.

The choice of control functions that we have used for all AQC simulations in this analysis is the linear schedule

$$A(t) = 1 - \frac{t}{t_f}, \quad B(t) = \frac{t}{t_f}. \tag{9}$$

While there exist strategies involving QW and AQC with different choices of control functions that can improve performance [13], we will not be exploring them in this work. The simulations in [5] used the same linear schedule as above with a constant duration $t_f = 100$, which is close to the adiabatic limit for most of the instances that were generated in their work. (Over half of the instances had success probabilities of $P(100) > 0.95$.) However, the instances that were selected for being difficult had success probabilities of $P(100) < 10^{-4}$, which puts their simulations far from the adiabatic limit. Hence, these instances are difficult for a coherent QA protocol that relaxes the condition of adiabaticity in AQC.

B. MAX 2-SAT

A Boolean formula $\phi = \phi(x_1, \ldots, x_n)$ consists of n Boolean variables $x_1, \ldots, x_n$, Boolean operators, and parentheses. The Boolean operators we will consider are conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$). Boolean variables can take one of the two possible logical values true (denoted 0) and false (denoted 1). A set of values that are assigned to the n variables in a formula is called an assignment, and for each assignment the Boolean formula ϕ will evaluate to either true or false. We define $2n$ literals $l_1, \ldots, l_n$ and $l_{-n}, \ldots, l_{-1}$ such that the literal l_i is associated with the variable x_i if i is positive, or $\neg x_{|i|}$ if i is negative.

In the maximum satisfiability problem (MAX SAT), a problem instance is specified by a Boolean formula ϕ that is in conjunctive normal form (CNF), which is a formula that is structured as a conjunction of m clauses, where a clause is a disjunction of literals. In this work, we will be studying maximum 2-satisfiability (MAX 2-SAT), which is a special case of MAX SAT where there are two literals in each clause. An example of a valid formula for MAX 2-SAT is

$$(x_1 \vee x_2) \wedge (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_2 \vee \neg x_3), \tag{10}$$

where $n = 3$ and $m = 6$ in this case. In MAX 2-SAT, any possible truth assignment is known as a solution, and we are tasked with finding an optimal solution, which is a solution that maximizes the number of clauses that evaluate to true. A clause that evaluates to true is said to be *satisfied*, and a clause that evaluates to false is said to be *unsatisfied*. The formula above has four optimal solutions which each satisfy five of the six clauses, one of which is the assignment $x_1 = 0,\ x_2 = 0,\ x_3 = 0$.

Although 2-SAT, the decision version of MAX 2-SAT, is in the complexity class P [14], MAX 2-SAT is NP-hard [15]. Nevertheless, MAX SAT solvers have made remarkable advancements over the past three decades, and they are able to solve or approximately solve MAX 2-SAT instances with relatively large input sizes. This progress can in part be attributed to annual competitions in producing the fastest SAT and MAX SAT solvers [7,16] and the large demand for these solvers, which is generated from the ability to efficiently map a wide range of practical problems to satisfiability problems. Examples include integrated circuit design debugging [17,18], cancer therapy design [19], software verification [20,21], and planning [22,23].

Outside the title and abstract of this paper, we avoid using the word "hardness" unless we are referring to the scaling of the computational complexity of a problem. When referring to the amount of resources used by a given algorithm to solve a particular instance of a problem, we use the word "difficulty"

instead. Note that some papers use the word "hardness" instead of "difficulty," which is what we have done in the title and abstract. An instance that is difficult for one algorithm may not necessarily be difficult for another algorithm. The hardness of a problem is typically measured by the worst-case or average-case computational complexity. Due to MAX 2-SAT being NP-hard [15], we expect a worse than polynomial scaling of the worst-case time complexity with n for any algorithm, assuming P $\neq$ NP.

In practice, the average run time scaling for a given set of instances may differ significantly from the worst-case time complexity of the problem and may be polynomial, even if the problem is NP-hard. The way in which instances are sampled is often an important consideration when performing an analysis on run times, especially when the problem is not uniformly hard. It has been shown that the difficulty of random MAX 2-SAT instances increases with the clause density $\rho = m/n$, and that there is a difficulty phase transition at the critical clause density $\rho_c = 1$ [24,25]. This has been demonstrated experimentally for QA [26] and has also been observed in numerical simulations of the quantum approximate optimization algorithm [27,28].

C. Problem mapping

In order to solve instances of MAX 2-SAT with a continuous-time quantum algorithm, a mapping of the problem as a Hamiltonian in the form given by Eq. (3) is required. Such a mapping should assign lower energies to eigenstates corresponding to more desirable solutions. Under the binary encoding of Boolean variables $x_i \in \{0, 1\}$ where 0 corresponds to true and 1 corresponds to false, the disjunction operator is equivalent to multiplication—i.e. $x_i \vee x_j$ can be written as $x_i x_j$. By identifying the variable x_i with the single-qubit basis state $|x_i\rangle$, we observe that

$$\frac{\mathbb{1} - \sigma_z}{2}|x_i\rangle = x_i|x_i\rangle, \tag{11}$$

where σ_z is the Pauli Z operator. For a clause $C_k = l_i \vee l_j$, where the literal l_i is positive (negative) if the number i is positive (negative), the corresponding term in the problem Hamiltonian can be constructed by taking the product

$$H_{C_k} = \frac{\mathbb{1} - \mathrm{sgn}(i)\sigma^z_{|i|}}{2}\frac{\mathbb{1} - \mathrm{sgn}(j)\sigma^z_{|j|}}{2}, \tag{12}$$

where we have used the sign function sgn to extract the sign of the indices. This term contributes an energy equal to $x_i x_j$. The problem Hamiltonian can then be constructed by taking a sum over the terms corresponding to each of the clauses in the Boolean formula ϕ, giving

$$H_{\text{problem}} = \sum_{C_k \in \phi} H_{C_k}. \tag{13}$$

The eigenvalues of this Hamiltonian are equal to the numbers of clauses that are unsatisfied by the assignments corresponding to the eigenstates.

D. Algorithm portfolios

One of the aims of this paper is to investigate the extent to which a portfolio-based strategy could improve the performance of continuous-time quantum algorithms. The portfolio approach is a simple method of achieving parallelism that was inspired by strategies for managing risk while increasing utility in economics [29]. It takes advantage of the lack of correlation in the difficulty of instances between several algorithms (together called a portfolio) by running the different algorithms in parallel. This approach has been applied to SAT solving [6,30], where it typically outperforms all other parallel strategies in competitions [7]. Optimal portfolios of recent SAT solvers and the impact of the portfolio size on performance have been studied [31]. For overviews of classical parallel SAT solving, see Refs. [32,33].

An advantage of the portfolio approach is that it has the potential to decrease a strategy's sensitivity to extremely difficult instances. As a simple example of this, consider two algorithms that each find a subset of instances extremely difficult to solve, where these two subsets do not overlap. These instances may not only significantly impact the worst-case performance of the algorithms but also the mean performance. If these algorithms were combined into a portfolio—for example, by running the two algorithms in parallel and allocating each algorithm half of the computing resources—the instances that are extremely difficult for each algorithm would be solved more efficiently by the other algorithm. This speedup comes at the cost of decreased performance for instances that the two algorithms find similarly difficult. Similar discussions of such performance/sensitivity trade-offs have been made in the context of no free lunch theorems for optimization [34–36]. While the no free lunch theorems do not apply to MAX 2-SAT in particular [37], an analysis of the implications of these theorems for portfolios of quantum and classical algorithms would be an interesting direction for future work.

In the context of quantum computing, parallel computing and the portfolio approach in particular have not been studied in significant depth. In Ref. [38], classical and quantum portfolios of quantum algorithms were found to perform better than standalone quantum algorithms for random 3-SAT. In this paper, we build on these results by studying the practical implications of a classical portfolio-based strategy for a selection of quantum and classical algorithms for MAX 2-SAT.

In related works that are not specifically on the subject of the portfolio approach, various hybrid quantum-classical algorithms for near-term quantum devices have been studied [39]. There have been efforts towards integrating quantum processors into modern high performance computing systems [40], which is an important step towards practically implementing a portfolio of quantum and classical algorithms. With regards to parallelism in continuous-time quantum computing, Pelofske *et al.* studied the use of quantum annealers to solve multiple independent problems in parallel on a single device [41,42].

III. NUMERICAL METHODS

A. Datasets

We performed our numerical study on two sets of MAX 2-SAT instances. The first set of instances were generated by

Crosson *et al.* in Ref. [5], and each instance in this set contains $n = 20$ Boolean variables and $m = 60$ unique clauses. Having a constant clause density $\rho = 3$ that is well above the critical clause density ensures that these instances are in the difficult regime. Each of the clauses in these instances were generated by randomly selecting two literals that are associated with distinct variables. Instances with multiple optimal solutions (corresponding to degenerate ground states of the problem Hamiltonian) were discarded. Note that this may lead to a different relation between instance difficulty and clause density compared to the phase transition results discussed in Sec. II B for random MAX 2-SAT. We are not aware of any results related to the phase transition in the context of instances with unique optimal solutions. 202 078 of such instances were generated, but only those with an AQC success probability at time $t_f = 100$ of $P(100) < 10^{-4}$ were selected, meaning that the 137 instances that remained are difficult for QA. For convenience, these instances were transformed by negating all literals corresponding to variables that were set to 1 in the original optimal assignment so that the optimal solution is always the $00\ldots0$ bit string.

The second set of instances were generated in a similar manner as those from Ref. [5]. For each number of variables in the range $5 \leqslant n \leqslant 20$, 10 000 instances containing $m = 3n$ unique clauses were generated with randomly selected pairs of literals corresponding to distinct variables for each clause, and only the instances with unique optimal solutions were kept. The transformation to set $00\ldots0$ as the optimal solution was applied. Unlike the instances from [5], there was no post-selection of a fraction of these instances based on difficulty; hence, we will refer to these as the "typical" instances. It has been shown that instances with an unbalanced ratio of positive to negative literals (or unnegated to negated variables) may be more difficult than balanced instances, where there are an equal number of positive and negative literals [43]. These instances are balanced on average and hence are not maximally difficult.

B. Numerical tests

All of the findings in this paper are results of numerical simulations that were carried out using the PYTHON programming language [44]. The NUMPY [45] and SCIPY [46] libraries were used for computationally intensive calculations, and MATPLOTLIB [47] was used for plotting. The implementation of the least squares method in scipy.optimize.curve_fit was used for obtaining the linear fits in this paper. The unitary time evolutions of quantum systems were simulated using QUIMB [48] as a convenient interface to scipy.integrate.complex_ode and its implementation of DOP853 [49], which is a Runge-Kutta method of order 8 with an adaptive step size. To calculate the average success probabilities in the limit of infinite time interval (shown in Appendix), Hamiltonians were diagonalized using numpy.linalg.eigh. Simulations of QW were run on high performance computers at Imperial College London, and simulations of AQC were run on the Hamilton high performance computing cluster at Durham University.

The quantity of interest for our analysis of QW is the average single run success probability $\overline{P}(t_I, \Delta t_I)$ given in Eq. (7), which is defined over an interval of measurement times $I = [t_I, t_I + \Delta t_I]$. We set $t_I = 0$ and $\Delta t_I = 100$ for our calculations, which produces an interval that is longer than the timescale of the QW dynamics. To demonstrate this, we plot the instantaneous QW success probability in this interval for pairs of randomly selected instances with $n = 5$ and $n = 20$ variables in Fig. 1, and we see that there are many oscillations in the success probability within the time interval in each case. As shown in Appendix, the specific choice of Δt_I does not significantly impact the results as long as it is longer than the timescale of the QW dynamics. The average success probability $\overline{P}(0, 100)$ was approximated by numerically integrating the Schrödinger equation over the full time interval and taking a weighted average of the instantaneous success probabilities of the solutions that were evaluated at each iteration of the integration method, where the weights are given by the amount of time between successive iterations.

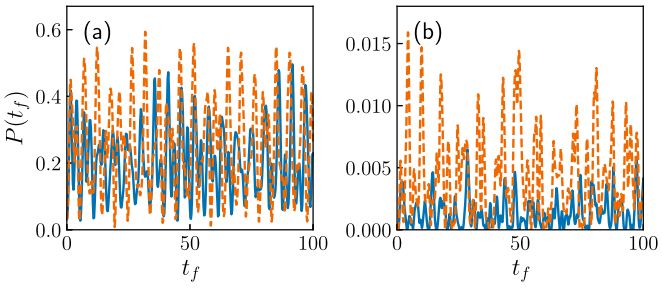

FIG. 1. Instantaneous QW success probability $P(t_f)$ for two randomly selected pairs of instances (solid blue and dashed orange lines) with (a) $n = 5$ variables and (b) $n = 20$ variables plotted against the measurement time t_f. Note the difference in the upper limits of the y axes.

For AQC, the quantity of interest is the minimum evolution duration required to achieve a 99% success probability $t_{0.99}$. To find $t_{0.99}$ for a given instance, the quantum dynamics were simulated for a small duration t_f and the success probability was calculated according to Eq. (6). If the success probability was less than 99%, t_f was doubled and a new success probability was calculated. This was repeated until either a success probability of greater than 99% was found or the simulations became too computationally intensive to continue, in which case $t_{0.99}$ was not found. If the original success probability was greater than 99%, t_f was instead halved each time until a duration with less than 99% success probability was found. The bisection method was then used to search for $t_{0.99}$ within the interval of the last two durations until a precision of at least 1% was reached.

The classical algorithm MIXBANDB [50] was applied to all of the generated MAX 2-SAT instances to compare its performance against the quantum algorithms. MIXBANDB was written to mirror some of the key characteristics of a highly competitive MAX SAT solver known as MIXSAT [51], without including many of the heuristic methods that MIXSAT employs. Like MIXSAT, MIXBANDB is a branch-and-bound algorithm that uses the "Mixing method" [52] as a semidefinite programming solver in order to produce lower bounds, and rounding to produce good guesses. MIXBANDB does not use a dual initialization strategy or any of the data structure or

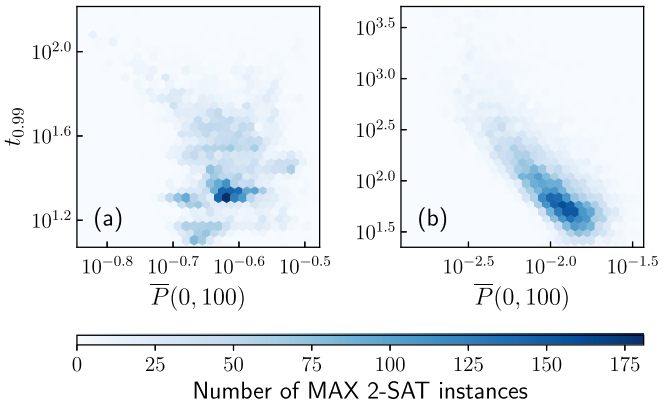

FIG. 2. Heatmaps of the AQC duration $t_{0.99}$ against the QW success probability $\overline{P}(0, 100)$ for typical MAX 2-SAT instances with (a) $n = 5$ and (b) $n = 15$ variables, excluding 138 instances of size $n = 15$ for which $t_{0.99}$ was not successfully calculated. Visually, the plot in (b) looks reasonably well correlated, whereas the plot in (a) does not appear to show much correlation. This is confirmed by Spearman's rank correlation coefficients, which were calculated to be $\approx$0.01 for $n = 5$ and ≈ -0.76 for $n = 15$.

implementation optimizations that are present in MIXSAT. For each instance, we measured the number of times MIXBANDB accessed the problem specification, which we refer to as the number of problem calls N_{calls}. This quantity serves as a proxy for the run time of the algorithm.

IV. RESULTS

In this section, we present an analysis of the difficulty of MAX 2-SAT instances for various quantum and classical algorithms based on the results of numerical simulation. The QW success probability $\overline{P}(0, 100)$ is used as a measure of the difficulty of instances for QW, where a higher success probability corresponds to a less difficult instance, and the AQC duration $t_{0.99}$ is used as a measure of difficulty of instances for AQC, where a longer duration corresponds to a more difficult instance. We start by making a cross-comparison between the difficulty of MAX 2-SAT instances for QW and AQC in Sec. IV A, and we make further comparisons that include difficulty for MIXBANDB in Sec. IV B. Where we were able to obtain results for 20-variable instances, a comparison with QA difficulty using the instances from Ref. [5] is also made. In Sec. IV C, we investigate the relative difficulty of satisfiable instances, which are classically easy to solve, for the algorithms that we are considering.

A. QW/AQC difficulty comparison

To characterize the relation between the QW difficulty and AQC difficulty of MAX 2-SAT instances, Fig. 2 shows the joint distribution of $\overline{P}(0, 100)$ and $t_{0.99}$ for typical instances with $n = 5$ variables and for $n = 15$ variables—-the latter being the largest problem size that we could calculate $t_{0.99}$ for. We calculate Spearman's rank correlation coefficient for the distributions to be $\approx$0.01 and ≈ -0.76 for $n = 5$ and 15, respectively. Spearman's rank correlation coefficient has a range of -1 (perfect anticorrelation between the rankings of the two quantities) through 0 (no correlation in rankings) to $+1$ (perfect correlation in rankings). Recalling that smaller $P(0, 100)$ and larger $t_{0.99}$ values indicate more difficult instances, the increasing negative Spearman's rank correlation coefficient values (anticorrelation) indicates a correlation between QW and AQC difficulty that gets stronger with n. This suggests that while a portfolio-based strategy may be able reduce the total run time, it would not produce a huge improvement at higher n, as the instances that are difficult for one algorithm are not likely to be found less difficult by the other algorithm.

In Fig. 2(b), a long tail of instances that are extremely difficult for both QW and AQC can be identified visually. These instances are towards the top left of the heatmap and are far from the location where the heatmap shows the highest density of instances. In comparison, the tail of instances to the bottom-right of the heatmap is much shorter, indicating that the least difficult instances are not as outlying as the most difficult instances for both QW and AQC. It is possible that typical instances of the $n = 15$ problem size are more dominated by less difficult instances than would be found at larger problem sizes. In other words, the top and left sides of the graph may have a higher density of instances when plotted for larger n, due to there being more instances in the "difficult tail" of the distribution. If this is the case, then the most difficult of the typical instances may be a better representation of the types of instances that are typically found at larger problem sizes. On top of this, instances of practical interest may have a significantly different composition of more and less difficult instances than our sample of randomly generated instances. Therefore it would be useful to find out whether the level of correlation between $\overline{P}(0, 100)$ and $t_{0.99}$ changes with the difficulty of the instances, but this is not easy to tell from these heatmaps.

A more detailed analysis of the difficulty of instances is required, to distinguish between the increase in difficulty with n that is simply due to the increase in problem size, from the change in the distribution of the difficulty of the instances for QW and AQC at each n. This is not easy to achieve because we expect the proportion of very difficult instances to increase with n. To accommodate such an analysis, we have followed a similar approach to other authors who have partitioned instances according to a measure of their difficulty [53,54]. Specifically, we have grouped the typical instances of each number of variables n into deciles that are ranked by difficulty. Decile 1 contains the 10% of the instances that are least difficult, decile 2 contains the next least difficult 10% of the instances, and so on, with decile 10 containing the most difficult 10% of the instances. This partitioning is done using the average QW success probability $\overline{P}(0, 100)$ as the measure of difficulty to produce "QW difficulty deciles" and similarly done using the 99% success probability duration $t_{0.99}$ for AQC to produce "AQC difficulty deciles." For AQC, it is assumed that the instances that we were not able calculate $t_{0.99}$ for in a reasonable amount of time are the most difficult instances. To identify the extremely difficult instances, we have also grouped together the most difficult 1% of instances at each n for QW and for AQC using the same measures of difficulty as for the deciles. By defining the QW/AQC "difficulty percentiles" in a similar way as the difficulty deciles, we can refer to the instances that are on the boundaries of the deciles by

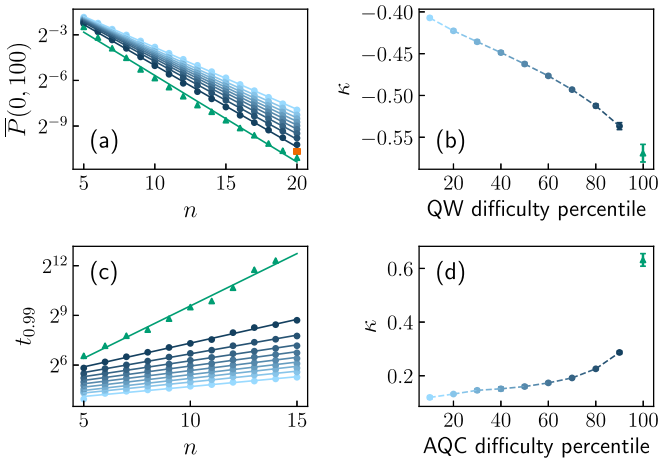

FIG. 3. (a) The QW success probabilities $\overline{P}(0, 100)$ of the 10th, 20th, …, 90th percentile instances (blue circles) and the 99th percentile instances (green triangles) for QW difficulty plotted against the number of variables n on log-linear axes, with a linear fit (solid line) for each percentile. The median QW success probability of the instances from Ref. [5] is also plotted (orange square). Darker shades of blue represent more difficult percentiles. (b) Plot of the values of the scaling exponents κ that have been inferred from the gradients of the linear fits, with error bars indicating standard errors. [(c) and (d)] Similar plots for the AQC duration $t_{0.99}$ and AQC difficulty percentiles. The results go up to $n = 20$ for QW and $n = 15$ for AQC due to time and resource constraints. Hence, there is no data point for the instances from Ref. [5] shown in (c). Also note that we could not calculate $t_{0.99}$ for the 99th percentile instance for AQC difficulty at $n = 15$, so there is no corresponding point in (c).

their percentiles. For example, the most difficult instance in the third QW difficulty decile is the 30th percentile instance for QW difficulty. Similarly, the most difficult instance for AQC that is not one of the most difficult 1% of instances for AQC is the 99th percentile instance for AQC difficulty.

Figure 3(a) shows a log-linear plot of the average success probabilities $\overline{P}(0, 100)$ for the most difficult instances of each QW difficulty decile (excluding the most difficult decile) and the 99th percentile instances for QW difficulty against the number of variables n. A linear fit is shown for each of the decile boundary instances and the 99th percentile instances, and the corresponding scaling exponents for each fit are plotted in Fig. 3(b). These plots show that the scaling of $\overline{P}(0, 100)$ gets progressively worse as the subset of selected instances gets more difficult. Therefore, at larger problem sizes, we can expect a bigger difference between the difficulty of the most and least difficult instances. The inferred scaling exponents also indicate that the tail of difficult instances gets longer as n is increased, which means that the time spent solving many instances would be largely dominated by the most difficult instances. This highlights the value of a portfolio approach, as any efficiency improvement for the most difficult instances would make a significant difference to the total run time.

An orange point indicating the median value of $\overline{P}(0, 100)$ for the instances from Ref. [5] is shown in Fig. 3(a). The placement of this point shows that these instances are also difficult for QW, which suggests that there is a correlation between QW and QA difficulty. These instances were selected to be the most difficult 137 instances for QA out of 202 078 randomly generated instances, meaning that they are all in the top 0.1% of the most difficult instances for QA. However, the median QW difficulty of these instance lies between the 90th and 99th percentiles of the typical instances, which indicates that the instances from [5] are not as extremely difficult for QW as they are for QA. Given that these instances are at least four orders of magnitude more difficult for QA than the median of the randomly generated instances, their lower relative difficulty for QW is substantial, and an approach involving QA would benefit from speeding up these instances by running QW in parallel.

Figures 3(c) and 3(d) show similar plots as above, but this time for the AQC duration $t_{0.99}$ and AQC difficulty deciles/percentiles. The median value of $t_{0.99}$ for the instances from [5] was not calculated, as these instances were too large to run AQC simulations for. Just as with the QW results, we find that more difficult percentiles scale more harshly for AQC, and this effect is even more prominent than for QW. (The decile boundary scaling exponents range from $\kappa = 0.119 \pm 0.006$ to 0.28 ± 0.005 for AQC, as opposed to $\kappa = -0.407 \pm 0.002$ to -0.537 ± 0.004 for QW.) Notably, there is a large jump in κ to $\approx 0.63 \pm 0.02$ between the 90th and 99th percentile instances, compared with a smaller jump to $\approx -0.57 \pm 0.01$ for QW, which implies that a small subset of the instances we are considering are extremely difficult for AQC. This is not surprising, as it is known that AQC performs very poorly on instances of other NP-hard problems when the minimum energy gap between the ground and first excited states is extremely small [55]. The steep gradient of the fit for the 99th percentile instances may be an indication that the number of extremely difficult instances is growing with n, which would be consistent with the idea that we would see a larger fraction of instances in the "difficult tail" at larger n. Therefore we are further motivated to analyze the relation between QW and AQC difficulty for these extremely difficult instances.

To make a cross-comparison between QW and AQC, we examine the QW difficulty of instances when grouped by AQC difficulty, and vice versa. In Fig. 4(a), we plot the median values of the QW success probability $\overline{P}(0, 100)$ for the instances in each AQC difficulty decile against n on log-linear axes. The median values of $\overline{P}(0, 100)$ for the most difficult 1% of instances for AQC are also plotted. A straight line is fit to the points for each of these groups of instances, and the corresponding scaling exponents are plotted in Fig. 4(b). The fact that more difficult AQC deciles tend to correspond to smaller median values of $\overline{P}(0, 100)$ implies that AQC difficulty is a good indicator of QW difficulty. Note that this is not the case at the lowest values of n, but the correlation becomes more clear as n is increased. This agrees with the correlations we found in Fig. 2. The implied scaling exponents show that this increase in QW difficulty with AQC difficulty becomes more prominent across all of the deciles as n is increased. In particular, the correlation between QW difficulty and AQC difficulty seems to remain strong at the "difficult tail" of the distribution. However, there is not a large jump in the scaling for the most difficult 1% of instances for AQC on QW, which shows that the extremely difficult instances for AQC are not as extremely difficult for QW. This indicates a significant

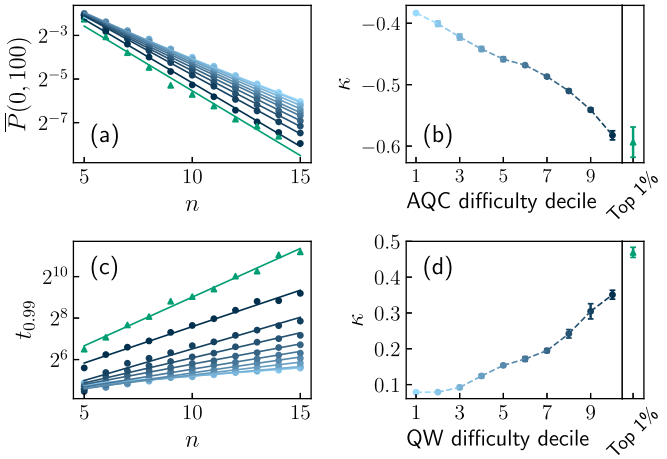

FIG. 4. (a) The median value of the QW success probability $\overline{P}(0, 100)$ of each of the AQC difficulty deciles (blue circles) and the most difficult 1% of instances for AQC (green triangles) plotted against the number of variables n on log-linear axes, with a linear fit (solid line) for each category. Darker shades of blue represent more difficult deciles. (b) Plot of the values of the scaling exponents κ that have been inferred from the gradients of the linear fits, with error bars indicating standard errors. [(c) and (d)] Similar plots for the median values of the AQC duration $t_{0.99}$ for instances organized by QW difficulty. Note that since there were more than 100 instances that we could not calculate $t_{0.99}$ for at $n = 15$, the plot in (a) does not include a point at $n = 15$ for the median of $\overline{P}(0, 100)$ for the most difficult 1% of instances for AQC.

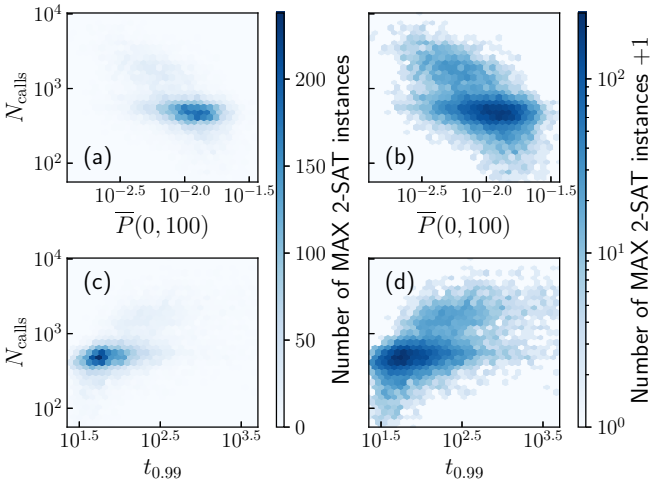

FIG. 6. Heatmaps of the number of problem calls N_{calls} against [(a) and (b)] the QW success probability $\overline{P}(0, 100)$ and [(c) and (d)] the AQC duration $t_{0.99}$ on log-log axes for typical instances with $n = 15$ variables. 138 instances for which $t_{0.99}$ was not successfully calculated are excluded in (c) and (d). (a) and (c) show the heatmaps with a linear color scale, and (b) and (d) show the same distributions with a logarithmic color scale, where the value for each cell has been increased by one to remove the zeros. Visually, some correlation can be seen in both distributions. Spearman's rank correlation coefficients are ≈ -0.47 for the distribution in (a) and ≈ 0.52 for the distribution in (c).

advantage of the portfolio approach for solving the instances that AQC finds extremely difficult.

We make a similar comparison in Figs. 4(c) and 4(d), where we plot the median AQC durations $t_{0.99}$, and the corresponding scaling exponents, for the instances in QW difficulty deciles and the 99th percentile instances for QW difficulty. In accordance with the previous results, these plots indicate that QW difficulty is a good indicator of AQC difficulty, though this is again more clearly the case for larger n. The trend of worsening scaling exponent with increasing difficulty decile in both sets of comparisons indicates that the QW-AQC difficulty correlation is likely to continue to larger n, even for the tail of difficult instances. Therefore a simple portfolio-based strategy is unlikely to produce a drastic speedup, though there is still room for some speedup, especially for the instances that are extremely difficult for AQC. Assuming that performing runs of both QW and AQC for each instance incurs a cost of roughly a factor of two to the total run time, even a small scaling advantage obtained from this approach would make up for this cost at large problem sizes.

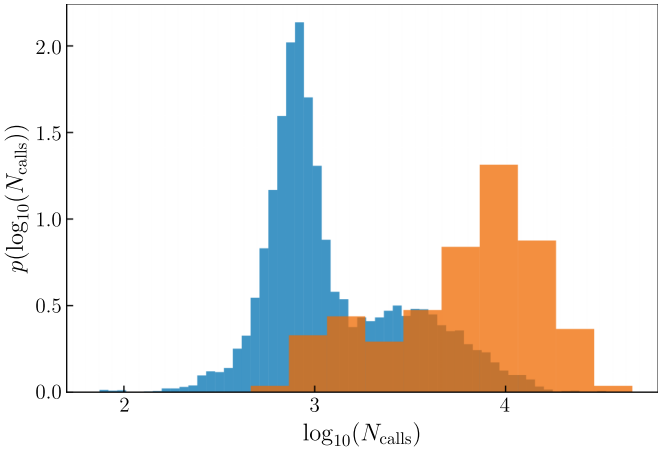

FIG. 5. Histogram showing the approximate probability density $p(\log_{10}(N_{\text{calls}}))$ of the logarithm of the number of problem calls N_{calls} made by the MIXBANDB algorithm when solving the typical $n = 20$ instances (blue) and the instances from Ref. [5] (orange), which are difficult for QA.

B. Quantum/classical difficulty comparison

To quantify the classical difficulty of MAX 2-SAT instances, we have measured the number of problem calls N_{calls} made by the classical algorithm MIXBANDB when solving each MAX 2-SAT instance. While there exist many classical algorithms that perform much better than MIXBANDB, they typically employ heuristic methods to gain a speed advantage, which make a significant difference at small problem sizes. This is undesirable for our analysis because the quantum algorithms we are comparing them to do not use such heuristics. MIXBANDB is based on MIXSAT, which is a powerful MAX SAT solver, but it does not incorporate the heuristic optimizations that MIXSAT uses. In this sense, MIXBANDB is a good classical comparison to QW and AQC. However, MIXBANDB is an exact solver, meaning that it always returns an optimal solution at the end of a run, whereas QW and AQC cannot give such guarantees. Since the purpose of this work is not

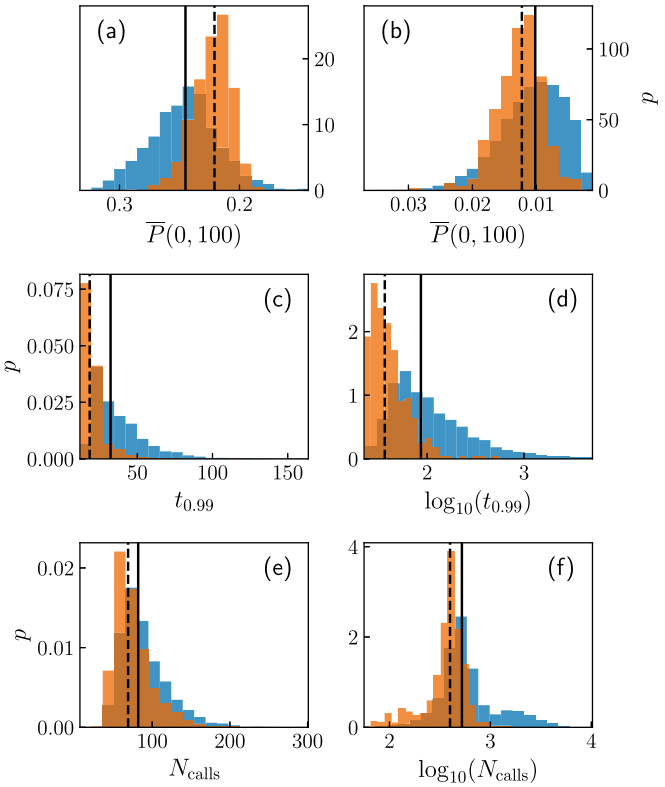

FIG. 7. Histograms of the approximate probability density p of [(a) and (b)] the QW success probability $\overline{P}(0, 100)$, (c) AQC duration $t_{0.99}$, (d) logarithm of $t_{0.99}$, (e) MIXBANDB calls N_{calls}, and (f) logarithm of N_{calls} for typical satisfiable (orange) and unsatisfiable (blue) MAX 2-SAT instances with [(a), (c), and (e)] $n = 5$ variables and [(b), (d), and (f)] $n = 15$ variables. The x axes have been reversed in [(a) and (b)] so that instance difficulty increases from left to right for all plots. Median values of the distributions are indicated by dashed lines for satisfiable instances and solid lines for unsatisfiable instances. Note that the distributions in (d) do not include 138 instances for which $t_{0.99}$ was not successfully calculated.

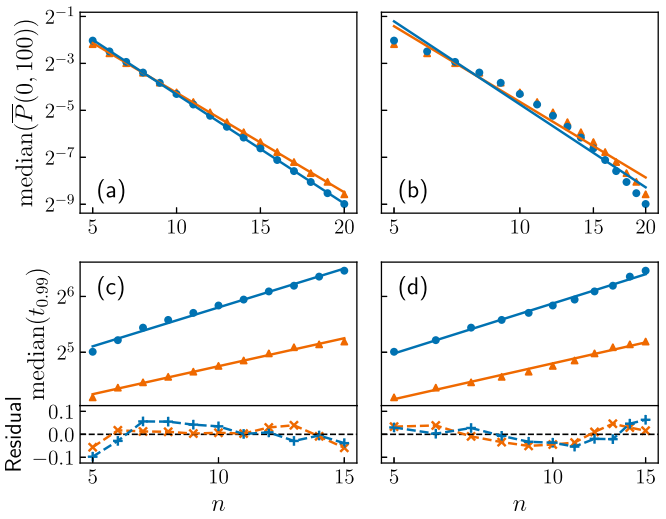

FIG. 8. Median QW success probability $\overline{P}(0, 100)$ for all satisfiable (orange triangles) and unsatisfiable (blue circles) typical instances against the number of variables n on (a) log-linear and (b) log-log axes. Linear fits are shown for both the satisfiable and unsatisfiable instances. [(c) and (d)] The same as above, but for the AQC duration $t_{0.99}$. Residuals, which are calculated logarithmically as $\log_2(\text{median}(t_{0.99})) - \log_2(y(n))$, where $y(n)$ is the corresponding fit, are also shown in [(c) and (d)] for both the satisfiable (orange crosses) and unsatisfiable (blue pluses) instances.

to benchmark the scalings of these algorithms but to compare the underlying mechanisms they use to solve problems, this difference is not important for our analysis.

Figure 5 shows a histogram of the approximate probability density of the logarithm of the number of calls N_{calls} made by MIXBANDB for typical instances with $n = 20$ variables and the instances that are difficult for QA. It can be seen that the typical instances form a bimodal distribution, where there is a peak of instances that required relatively few calls and a long tail of more difficult instances that form another peak at a higher number of calls. This suggests that the typical instances can be roughly divided into two sets for their difficulty classically, with the majority of instances being in the less difficult set. The instances that are difficult for QA also form a bimodal distribution but with a larger peak of difficult instances than less difficult instances. The center of this distribution is located on the "difficult tail" of typical instances, suggesting that difficult instances for QA tend to be difficult for MIXBANDB too.

In Fig. 6, we plot the joint distribution of N_{calls} and $\overline{P}(0, 100)$ for the typical instances with $n = 15$ variables, and the joint distribution of N_{calls} and $t_{0.99}$ for the 9862 typical instances that we successfully calculated $t_{0.99}$ for. The former distribution has Spearman's rank correlation coefficient of ≈ -0.47 and the latter has Spearman's rank correlation coefficient of ≈ 0.52. These imply that correlations exist between difficulty for MIXBANDB and difficulty for both QW and AQC, although the correlations are not as strong as what we previously observed between QW difficulty and AQC difficulty at the same problem size. Therefore it seems that a portfolio-based strategy would be more effective when applied to a quantum and classical algorithm, as opposed to QW and AQC. This does not rule out the possibility of attaining a better performance from other forms of (hybrid) quantum algorithms. Examples of other techniques in continuous-time quantum computing that can be incorporated in a hybrid approach include: Preannealing [13]; local quantum searches, which can be performed with the addition of a biased Hamiltonian in coherent annealing [56–59] or with just the driver and problem Hamiltonians in dissipative reverse annealing [60]; annealing schedules that interpolate between QW and AQC [4]; and a variety of approaches that fall under the umbrella of diabatic quantum computing, which are reviewed in Ref. [61]. For a detailed review of hybrid approaches involving quantum and classical algorithms, see Ref. [39].

We note that since different classical algorithms will have varying levels of correlation with each other, our measure of "classical difficulty" cannot be extended to represent difficulty for classical algorithms as a whole, since such a thing does not exist. However, the fact that some correlation exists between difficulty for the quantum algorithms and difficulty for MIXBANDB is still interesting, as it indicates that there are some characteristics of instances that make them typically

more difficult for both the quantum algorithms and some "good" classical algorithms. Further work is required to find out if there are other good classical algorithms with lower levels of correlation with the quantum algorithms.

C. Satisfiable instances

Satisfiable instances of MAX 2-SAT are easy to solve classically because the optimal solution can be found with a 2-SAT solver, and 2-SAT is known to be in P; a linear time algorithm for 2-SAT was found in Ref. [14], which is based on finding the strongly connected components of the problem's implication graph. Some of the typical instances that we are analyzing are satisfiable, the proportion of which decreases with n. A good classical MAX SAT solver will take advantage of this to solve satisfiable instances efficiently, for example by calling a 2-SAT solver at the start of the algorithm. However, since the quantum algorithms we are studying do not explicitly check for the satisfiability of formulas, it is unclear whether they will solve satisfiable instances efficiently. In this subsection, we analyze the difficulty of satisfiable instances for QW and AQC and discuss the implications of this for a portfolio approach.

The plots in Figs. 7(a) and 7(b) show the approximate probability density of the QW success probability $\overline{P}(0, 100)$ for satisfiable and unsatisfiable instances with $n = 5$ and $n = 15$ variables. The median value of each distribution is indicated by a vertical line. We find that the median $\overline{P}(0, 100)$ for satisfiable instances is larger than for unsatisfiable instances at $n = 15$, but at $n = 5$, the satisfiable instances have a smaller median $\overline{P}(0, 100)$ than the unsatisfiable instances. This indicates that QW finds satisfiable instances less difficult on average except in the case of very small n, although even at $n = 5$ the most difficult instances still tend to be unsatisfiable. We note that the long tail of difficult instances cannot be as easily identified in Fig. 7(b) as it can in Fig. 2, which is because we are using a linear x axis in Fig. 7 and QW difficulty is inversely proportional to $\overline{P}(0, 100)$. The tail of difficult instances can be clearly seen in both of the $n = 15$ distributions when plotting $1/\overline{P}(0, 100)$ or using a logarithmic x axis.

In practice, the satisfiable instances are less difficult for QW than what can be inferred from the plots in Fig. 7. This is due to the fact that the QW success probability is generally much less than 1, so repeat runs are needed to increase the probability of finding the optimal solution. From Eq. (12), we can see that each unsatisfied clause adds an energy contribution of 1. A simple improvement to a QW strategy would be to measure the energy of the final state or classically evaluate the formula with the solution given by QW to efficiently determine the number of clauses that are left unsatisfied by the corresponding solution. If the optimal solution of a satisfiable instance is found, it would become immediately obvious that there are no more clauses that can be satisfied, so repeat runs would no longer be necessary. For unsatisfiable instances, we cannot be certain that we have found an optimal solution based on the information gained from QW alone, so there will typically be "wasted" runs conducted after finding the optimal solution. The number of extra runs depends on the specific strategy used to determine when to stop doing repeats. There has been previous work on applying sophisticated methods of determining the stopping point to quantum annealing [62].

Figures 7(c) and 7(d) show the approximate probability density of the AQC duration $t_{0.99}$ for typical satisfiable and unsatisfiable instances with $n = 5$ variables, and similarly for the approximate probability density of the logarithm of $t_{0.99}$ for $n = 15$. The satisfiable instances have shorter median durations than the unsatisfiable instances for both $n = 5$ and $n = 15$, indicating that AQC finds these problems less difficult on average. Since AQC achieves a high success probability in a single run, we do not need to do repeat runs as in the case for QW, so these plots are a good representation of the difference in AQC difficulty between satisfiable and unsatisfiable instances. Figures 7(e) and 7(f) show similar results for the MIXBANDB algorithm, which like the quantum algorithms does not check for satisfiability. Satisfiable instances are found less difficult on average, but the distributions for satisfiable and unsatisfiable instances overlap.

The significant overlap between the distributions of satisfiable and unsatisfiable instances in Fig. 7 either imply that QW and AQC do not find the difficulty of satisfiable instances to be as low as their difficulty for the best classical algorithms, or that a large fraction of the unsatisfiable instances at these problem sizes have just as low difficulties as the satisfiable instances, which can be solved efficiently. To determine which of these two cases is true, we can check whether the difficulty of satisfiable instances scales exponentially for the quantum algorithms, which would support the former as it would indicate that QW and AQC cannot solve them efficiently.

We plot the scaling of the median QW success probability and AQC duration with n for satisfiable and unsatisfiable instances on log-linear and log-log axes in Fig. 8. An exponential scaling would fit better to a straight line on log-linear axes, whereas a polynomial scaling would have a better linear fit on log-log axes. For QW, we can see that the log-linear axes produce better fits for both sets of instances, meaning that the median of $\overline{P}(0, 100)$ appears to scale exponentially with n for both satisfiable and unsatisfiable instances. For AQC, it is unclear which fits are better. We note that these results cannot be used to make statements about the form of the scalings with certainty, as the problem sizes are very small compared to practically relevant instances and the scalings may change at larger sizes.

Given the linear worst-case scaling of good classical 2-SAT solvers and the apparent exponential scaling of QW (and potentially AQC) on satisfiable instances, we can conclude that a classical algorithm should most likely be used to efficiently check the satisfiability of instances at the start of any portfolio-based strategy involving QW and/or AQC, as these instances cannot be solved efficiently by the quantum algorithms. This will speed up the time to solution for satisfiable instances while only contributing a linear overhead to the run time for unsatisfiable instances. An added benefit of this approach is that instances with exactly one clause left unsatisfied by an optimal assignment can be solved faster by QW. This is because by running a classical 2-SAT solver in advance, it would become known when a formula is unsatisfiable and that any assignment satisfying all but one clause is optimal. Therefore there would be no need for repeat runs of

QW after an optimal solution is found, for the same reasons as we mentioned for satisfiable instances. This does not apply to AQC as it does not require repeat runs.

V. CONCLUSIONS

We have examined both the relative difficulty of different instances and the correlation in difficulty for a selection of both classical and quantum algorithms. These include algorithms that behave in conceptually different ways, for example quantum walk, which relies on many repeats with a relatively low success probability, versus adiabatic quantum computing, which succeeds with a high probability after a single run. Our work shows that it is important to include a thorough characterization of the problem instances used for numerical studies of the performance of quantum algorithms. We have found that while there is some correlation in MAX 2-SAT instance difficulty between methods, the correlation seems weak enough that a strategy of attempting a portfolio of algorithms in parallel is viable and likely to be desirable in real computation. We also note unique features of specific strategies. For example, the performance of the most difficult instances for AQC is drastically worse than for more "typical" problems, much more so than for quantum walk. This can be attributed to the presence of instances with extremely small spectral gaps, which limit the performance of AQC [63,64]. Extremely small gaps have been observed in other contexts [65]. This catastrophic failure of AQC suggests that a "standalone" adiabatic strategy without attempting others in parallel is likely to be particularly undesirable. We further find that while the performance of quantum algorithms is generally better for satisfiable problems (which can be solved efficiently classically), these problems are still not solved efficiently by either of the quantum algorithms. This strongly suggests that first performing a classical check for satisfiability is useful. In a sense, attempting different algorithms in parallel can be seen as the most trivial case of a hybrid algorithm. If the algorithms are classical and quantum then it is a hybrid quantum-classical algorithm, but there can also be hybrids between two quantum algorithms, such as AQC and QW. While not the topic of this paper, the fact that even such simple hybrid methods are desirable bodes well for more complicated methods of combining algorithms that are likely to lead to further gains, for example, preannealing in [13] as a quantum-quantum algorithm, or various biasing and reverse annealing techniques [60] as examples of quantum-classical hybrids.

The correlations do suggest that while attempting multiple different algorithms in parallel is likely to be fruitful, there is also likely a more fundamental sense of difficulty in terms of being resistant to being efficiently solved by any algorithm, quantum or classical. Furthermore, the double peaked nature of many of the distributions of effort required for problems suggests that the transition toward being predominantly difficult as problems scale toward the large size limit is not simple, at least not in the case of MAX 2-SAT. This behavior is partially explained by the difference between satisfiable and unsatisfiable instances, but this appears not to be the whole story because there is significant overlap between the two in terms of difficulty. While we have made significant steps in understanding relative problem difficulty over different algorithms at sizes relevant for exhaustively simulated quantum computing, there is still much work to be done to fully understand this important topic which underlies many numerical studies.

The data for all MAX 2-SAT problem instances used in this research are openly available at Ref. [66].

ACKNOWLEDGMENTS

We thank Elizabeth Crosson for providing data for difficult MAX 2-SAT instances. We also thank the members of Algorithms and Complexity in Durham for helpful discussions. N.C. and V.K. were supported by UKRI EPSRC (EP/L022303/1) and impact acceleration funding associated with this grant. N.C. was supported by UKRI EPSRC fellowship EP/S00114X/1. P.M. was supported by UKRI EPSRC Doctoral Training Funds awarded to Durham University (EP/T518001/1) in partnership with dunnhumby. AC was funded by UKRI EPSRC Grant No. EP/L016524/1 via the Imperial College London Centre for Doctoral Training in Controlled Quantum Dynamics, and the UKRI EPSRC UK Quantum Technology Hub in Computing and Simulation (EP/T001062/1).

APPENDIX

Our numerical analysis of QW is based on the average success probability $\overline{P}(0, 100)$, which is taken over an arbitrary time interval $\Delta t_l = 100$. Since this time interval is longer than the timescale of the QW dynamics (see Fig. 1), we expect that the success probability would not be significantly affected by a different choice of Δt_l that is also sufficiently large. To confirm this, we consider the infinite time interval limit of the average success probability, $P_\infty \equiv \lim_{\Delta t_l \to \infty} \overline{P}(0, \Delta t_l)$.

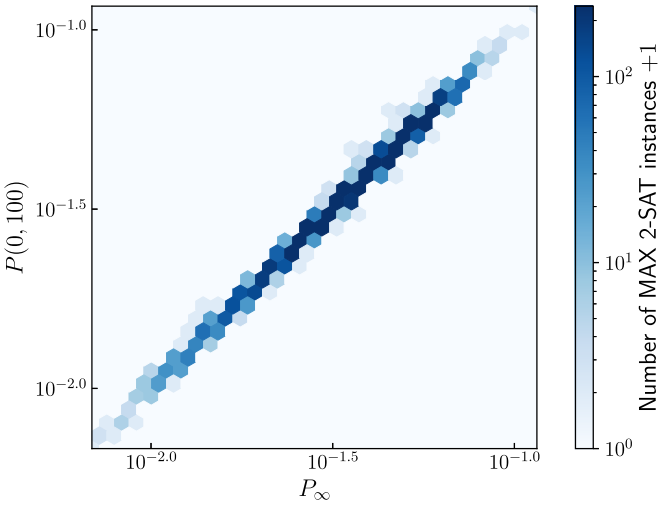

FIG. 9. Heatmap of the QW success probability $\overline{P}(0, 100)$ averaged over a time interval $\Delta t_l = 100$ against the same average probability in the limit of infinite time interval, P_∞, for the typical MAX 2-SAT instances with $n = 11$ variables. A logarithmic color scale is used for better visibility, and the zeros have been removed by adding 1 to the number of instances for each cell. Spearman's rank correlation coefficient between the two quantities is ≈1.00, indicating an almost perfect correlation.

We followed the procedure outlined in Ref. [1] to calculate P_∞ by numerically diagonalising the QW Hamiltonian. Figure 9 shows that for the typical instances with $n = 11$ variables, P_∞ and $\overline{P}(0, 100)$ are in very good agreement. Therefore we can assume that the results of this paper are not dependent on our specific choice of Δt_1.

[1] A. Callison, N. Chancellor, F. Mintert, and V. Kendon, Finding spin glass ground states using quantum walks, New J. Phys. **21**, 123022 (2019).

[2] J. Roland and N. J. Cerf, Quantum search by local adiabatic evolution, Phys. Rev. A **65**, 042308 (2002).

[3] N. Shenvi, J. Kempe, and K. Birgitta Whaley, Quantum random-walk search algorithm, Phys. Rev. A **67**, 052307 (2003).

[4] J. G. Morley, N. Chancellor, S. Bose, and V. Kendon, Quantum search with hybrid adiabatic–quantum-walk algorithms and realistic noise, Phys. Rev. A **99**, 022339 (2019).

[5] E. Crosson, E. Farhi, C. Y.-Y. Lin, H.-H. Lin, and P. Shor, Different strategies for optimization using the quantum adiabatic algorithm, arXiv:1401.7320 (2014).

[6] Y. Hamadi, S. Jabbour, and L. Sais, ManySAT: A Parallel SAT Solver, J. Satisf. Boolean Model. Comput. **6**, 245 (2009).

[7] SAT Competition, https://satcompetition.github.io/ (2023) [online; accessed 14-April-2023].

[8] E. Farhi and S. Gutmann, Quantum computation and decision trees, Phys. Rev. A **58**, 915 (1998).

[9] T. G. Wong and D. A. Meyer, Irreconcilable difference between quantum walks and adiabatic quantum computing, Phys. Rev. A **93**, 062313 (2016).

[10] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, Quantum computation by adiabatic evolution, arXiv:quant-ph/0001106.

[11] M. Born and V. Fock, Beweis des Adiabatensatzes, Z. Phys. **51**, 165 (1928).

[12] D. J. Griffiths, *Introduction to Quantum Mechanics*, 2nd ed. (Pearson Prentice Hall, Upper Saddle River, NJ, 2004).

[13] A. Callison, M. Festenstein, J. Chen, L. Nita, V. Kendon, and N. Chancellor, Energetic perspective on rapid quenches in quantum annealing, PRX Quantum **2**, 010338 (2021).

[14] B. Aspvall, M. F. Plass, and R. E. Tarjan, A linear-time algorithm for testing the truth of certain quantified boolean formulas, Inf. Process. Lett. **8**, 121 (1979).

[15] M. R. Garey, D. S. Johnson, and L. Stockmeyer, Some simplified NP-complete problems, in *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing - STOC '74* (Association for Computing Machinery, Seattle, WA, 1974), pp. 47–63, doi: 10.1145/800119.803884.

[16] MaxSAT Evaluations, https://maxsat-evaluations.github.io/ (2023) [online; accessed 14-April-2023].

[17] S. Safarpour, H. Mangassarian, A. Veneris, M. H. Liffiton, and K. A. Sakallah, Improved design debugging using maximum satisfiability, in *Formal Methods in Computer Aided Design (FMCAD '07)* (IEEE, Austin, TX, 2007), pp. 13–19.

[18] Y. Chen, S. Safarpour, J. Marques-Silva, and A. Veneris, Automated design debugging with maximum satisfiability, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **29**, 1804 (2010).

[19] P.-C. Lin and S. P. Khatri, Application of Max-SAT-based ATPG to optimal cancer therapy design, BMC Genomics **13**, S5 (2012).

[20] E. Clarke, D. Kroening, and F. Lerda, A tool for checking ANSI-C Programs, in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2004)* (Springer, Barcelona, Spain, 2004), pp. 168–176.

[21] F. Ivančić, Z. Yang, M. K. Ganai, A. Gupta, and P. Ashar, Efficient SAT-based bounded model checking for software verification, Theor. Comput. Sci. **404**, 256 (2008).

[22] H. Kautz, B. Selman, and M. Hill, Planning as Satisfiability, in *European Conference on Artificial Intelligence* (Vienna, Austria, 1992), pp. 359–363.

[23] J. Rintanen, Planning as satisfiability: Heuristics, Artif. Intell. **193**, 45 (2012).

[24] D. Coppersmith, D. Gamarnik, M. T. Hajiaghayi, and G. B. Sorkin, Random MAX SAT, random MAX CUT, and their phase transitions, Random Struct. Alg. **24**, 502 (2004).

[25] W. Fernandez de la Vega, Random 2-SAT: Results and problems, Theor. Comput. Sci. **265**, 131 (2001).

[26] S. Santra, G. Quiroz, G. Ver Steeg, and D. A. Lidar, Max 2-sat with up to 108 qubits, New J. Phys. **16**, 045006 (2014).

[27] V. Akshay, H. Philathong, M. E. S. Morales, and J. D. Biamonte, Reachability Deficits in Quantum Approximate Optimization, Phys. Rev. Lett. **124**, 090504 (2020).

[28] J. Golden, A. Bärtschi, D. O'Malley, and S. Eidenbenz, The quantum alternating operator ansatz for satisfiability problems, arXiv:2301.11292.

[29] B. A. Huberman, R. M. Lukose, and T. Hogg, An economics approach to hard computational problems, Science **275**, 51 (1997).

[30] T. Balyo, P. Sanders, and C. Sinz, HordeSat: A massively parallel portfolio SAT solver, in *Theory and Applications of Satisfiability Testing –SAT 2015* (Springer International Publishing, Austin, TX, 2015), pp. 156–172, doi: 10.1007/978-3-319-24318-4_12.

[31] J. Bach, M. Iser, and K. Böhm, A comprehensive study of k-portfolios of recent SAT solvers, in *25th International Conference on Theory and Applications of Satisfiability Testing (SAT 2022)*, Leibniz International Proceedings in Informatics (LIPIcs, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Haifa, 2022), Vol. 236, pp. 2:1–2:18, doi: 10.4230/LIPIcs.SAT.2022.2.

[32] R. Martins, V. Manquinho, and I. Lynce, An overview of parallel SAT solving, Constraints **17**, 304 (2012).

[33] S. Hölldobler, N. Manthey, Van Hau Nguyen, J. Stecklina, and P. Steinke, A short overview on modern parallel SAT-solvers, in 2011 International Conference on Advanced Computer Science and Information Systems (IEEE, Jakarta, 2011), pp. 201–206.

[34] D. H. Wolpert and W. G. Macready, No free lunch theorems for search, Technical Report SFI-TR-95-02-010, 1 (Santa Fe Institute, Santa Fe, NM, 1995).

[35] D. Wolpert and W. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. **1**, 67 (1997).

[36] Y. Ho and D. Pepyne, Simple Explanation of the No-Free-Lunch Theorem and Its Implications, J. Optim. Theory Appl. **115**, 549 (2002).

[37] J. McDermott, When and why metaheuristics researchers can ignore "no free lunch" theorems, SN Computer Science **1**, 60 (2020).

[38] S. M. Maurer, T. Hogg, and B. A. Huberman, Portfolios of Quantum Algorithms, Phys. Rev. Lett. **87**, 257901 (2001).

[39] A. Callison and N. Chancellor, Hybrid quantum-classical algorithms in the noisy intermediate-scale quantum era and beyond, Phys. Rev. A **106**, 010101 (2022).

[40] T. S. Humble, A. McCaskey, D. I. Lyakh, M. Gowrishankar, A. Frisch, and T. Monz, Quantum Computers for High-Performance Computing, IEEE Micro **41**, 15 (2021).

[41] E. Pelofske, G. Hahn, and H. N. Djidjev, Parallel quantum annealing, Sci. Rep. **12**, 4499 (2022).

[42] E. Pelofske, G. Hahn, D. O'Malley, H. N. Djidjev, and B. S. Alexandrov, Quantum annealing algorithms for Boolean tensor networks, Sci. Rep. **12**, 8539 (2022).

[43] P. Austrin, Balanced Max 2-sat might not be the hardest, in *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing (STOC '07)* (ACM Press, San Diego, California, USA, 2007), p. 189, doi: 10.1145/1250790.1250818.

[44] G. van Rossum and F. L. Drake Jr., *Python Tutorial* (Centrum voor Wiskunde en Informatica, Amsterdam, 1995).

[45] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith *et al.*, Array programming with NumPy, Nature (London) **585**, 357 (2020).

[46] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright *et al.*, SciPy 1.0: Fundamental algorithms for scientific computing in Python, Nat. Methods **17**, 261 (2020).

[47] J. D. Hunter, Matplotlib: A 2D Graphics Environment, Comput. Sci. Eng. **9**, 90 (2007).

[48] J. Gray, quimb: A python package for quantum information and many-body calculations, Journal of Open Source Software **3**, 819 (2018).

[49] E. Hairer, G. Wanner, and S. P. Nørsett, *Solving Ordinary Differential Equations I: Nonstiff Problems*, Springer Series in Computational Mathematics (Springer, Berlin, Heidelberg, 1993), Vol. 8, doi: 10.1007/978-3-540-78862-1.

[50] A. Callison, Continuous-time quantum computing, Ph.D. thesis, Imperial College London, 2021.

[51] P.-W. Wang and Z. J. Kolter, Low-Rank Semidefinite Programming for the MAX2SAT Problem, Proceedings of the AAAI Conference on Artificial Intelligence **33**, 1641 (2019).

[52] P.-W. Wang, W.-C. Chang, and J. Z. Kolter, The Mixing method: Low-rank coordinate descent for semidefinite programming with diagonal constraints, arXiv:1706.00476.

[53] D. Wecker, M. B. Hastings, and M. Troyer, Training a quantum optimizer, Phys. Rev. A **94**, 022309 (2016).

[54] S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J.-G. Liu, R. Samajdar *et al.*, Quantum optimization of maximum independent set using Rydberg atom arrays, Science **376**, 1209 (2022).

[55] D. S. Steiger, T. F. Rønnow, and M. Troyer, Heavy Tails in the Distribution of Time to Solution for Classical and Quantum Annealing, Phys. Rev. Lett. **115**, 230501 (2015).

[56] A. Perdomo-Ortiz, S. E. Venegas-Andraca, and A. Aspuru-Guzik, A study of heuristic guesses for adiabatic quantum computation, Quant. Info. Proc. **10**, 33 (2011).

[57] Q.-H. Duan, S. Zhang, W. Wu, and P.-X. Chen, An alternative approach to construct the initial Hamiltonian of the adiabatic quantum computation, Chin. Phys. Lett. **30**, 010302 (2013).

[58] M. Ohkuwa, H. Nishimori, and D. A. Lidar, Reverse annealing for the fully connected p-spin model, Phys. Rev. A **98**, 022314 (2018).

[59] T. Graß, Quantum Annealing with Longitudinal Bias Fields, Phys. Rev. Lett. **123**, 120501 (2019).

[60] N. Chancellor, Modernizing quantum annealing using local searches, New J. Phys. **19**, 023024 (2017).

[61] E. J. Crosson and D. A. Lidar, Prospects for quantum enhancement with diabatic quantum annealing, Nat. Rev. Phys. **3**, 466 (2021).

[62] W. Vinci and D. A. Lidar, Optimally Stopped Optimization, Phys. Rev. Appl. **6**, 054016 (2016).

[63] S. Knysh, Zero-temperature quantum annealing bottlenecks in the spin-glass phase, Nat. Commun. **7**, 12370 (2016).

[64] T. Albash and D. A. Lidar, Adiabatic quantum computation, Rev. Mod. Phys. **90**, 015002 (2018).

[65] N. Chancellor and V. Kendon, Experimental test of search range in quantum annealing, Phys. Rev. A **104**, 012604 (2021).

[66] A. Callison, P. Mirkarimi, N. Chancellor, L. Light, E. Crosson, and V. Kendon, Comparing the hardness of MAX 2-SAT problem instances for quantum and classical algorithms [dataset], Data archive at Durham University, UK, http://doi.org/10.15128/r2x346d419f.