Load balancing for high performance computing using quantum annealing

Omer Rathore [1,*] Alastair Basden [1,†] Nicholas Chancellor [1,2,‡] and Halim Kusumaatmaja [1,3,§]

[1]*Department of Physics, Durham University, Durham DH1 3LB, England, United Kingdom*
[2]*School of Computing, Newcastle University, Newcastle upon Tyne NE4 5TG, England, United Kingdom*
[3]*Institute for Multiscale Thermofluids, School of Engineering, The University of Edinburgh, Edinburgh EH9 3FB, Scotland, United Kingdom*

(Received 14 August 2024; accepted 9 November 2024; published 17 January 2025)

With the advent of exascale computing, effective load balancing in massively parallel software applications is critically important for leveraging the full potential of high-performance computing systems. Load balancing is the distribution of computational work between available processors. Here, we investigate the application of quantum annealing to load balance two paradigmatic algorithms in high-performance computing. Namely, adaptive mesh refinement and smoothed particle hydrodynamics are chosen as representative grid and off-grid target applications. While the methodology for obtaining real simulation data to partition is application specific, the proposed balancing protocol itself remains completely general. In a grid based context, quantum annealing is found to outperform classical methods such as the round robin protocol but lacks a decisive advantage over more advanced methods such as steepest descent or simulated annealing despite remaining competitive. The primary obstacle to scalability is found to be limited coupling on current quantum annealing hardware. However, for the more complex particle formulation, approached as a multiobjective optimization, quantum annealing solutions are demonstrably Pareto dominant to state of the art classical methods across both objectives. This signals a noteworthy advancement in solution quality which can have a large impact on effective CPU usage.

DOI: 10.1103/PhysRevResearch.7.013067

I. INTRODUCTION

During the initial development of scientific computation, simulations greatly benefited from improvements to clock speeds of individual processors [1]. However, since the early 2000s [2] hardware limitations have resulted in dwindling improvements to sequential programming applications. The result has been a paradigm shift to concurrency in programming software that aims to exploit the multicore architectures which form the bedrock of modern day high-performance computing (HPC). However, the effectiveness of these applications is heavily dependent on the equitable distribution of computational workload across available resources, a concept known as load balancing.

Load balancing encompasses not just fair distribution of the computational tasks across processors, but also doing so in such a way that minimizes the need to communicate data between processors. Motivation for the former is evident considering how all processors need to wait for the slowest one to finish before proceeding to the next step, thus potentially resulting in idling and resource wastage at a much larger scale if even a single processor lags behind. In addition to this, communication bandwidth between processors is usually not as efficient as within the processor itself [3,4] and so the volume of communication ideally should be minimized. The importance of load balancing clearly extends beyond any singular discipline. However, for the purposes of this paper emphasis is placed on applications in the realm of computational fluid dynamics. In particular, this paper explores the viability of quantum annealing as a solution strategy, as demonstrated using representative grid and off-grid applications.

In the context of grid based applications, it is usually not the governing equations themselves that define the load balancing problem but the chosen method of domain decomposition (DD). DD is the art of splitting a computational domain into smaller subdomains. This allows solving each smaller problem individually before subsequently recombining into the global solution. Historically, since its original proposal [5], DD in the realm of fluid simulations has been used to accommodate the inherent challenges of complex geometries [6,7] or higher dimensionality [8]. Extensive research in the last 40 years has led to rapid development, facilitating application to multiphysics problems [9], moving meshes [10], and a wider pool of applications [11]. More importantly in the context of modern HPC, DD is an indispensable tool for transforming the global simulation domain into subcomponents that can be assigned to individual processors.

*Contact author: omer.rathore@durham.ac.uk
†Contact author: a.g.basden@durham.ac.uk
‡Contact author: nick.chancellor@newcastle.ac.uk
§Contact author: halim.kusumaatmaja@ed.ac.uk

A popular choice of DD for large scale simulations in the realms of fluids [12], stress analysis [13], and biological flows [14] among others is adaptive mesh refinement (AMR). AMR operates on the principle that different areas of the computational domain are best solved with different levels of spatial resolution. This can prevent wastage of computational resources in regions with little activity while maintaining precision in zones of interest. The method is particularly useful for applications with large variation of spatial scales or when high resolution data are required in specific regions. This paper considers block structured implementations as a representative class and does not make further distinctions with alternative AMR varieties, although an interested reader is referred to the relevant literature [15,16].

For off-grid applications, the representative method here is smoothed particle hydrodynamics (SPH), a mesh free simulation method for continuum mechanics. Since its original conception [17], its application has spread to encompass a wide variety of fields including but not limited to solid mechanics, engineering, astrophysics, and the food industry [18,19]. Fundamentally, SPH is an interpolation technique that utilizes a collection of unordered points, or particles, to ascertain the value of a function at a specific point. This process is facilitated through the use of a kernel function, which effectively integrates the influence of adjacent particles to compute the value of a function at the desired point.

When applied in the context of computational fluids this forms a Lagrangian particle method for solving hydrodynamic equations, thus forming an equivalent counterpart to the Eulerian equations solved by grid based methods. Moreover, when compared with the latter, SPH is inherently more adept at handling free surface flows and complex boundaries and can accommodate large deformations without being hindered by grid distortion. However, these advantages are usually accompanied with caveats when it comes to stability and convergence, as well as limited accuracy [20].

This paper delves into the potential of quantum computing (QC) to address the challenges of load balancing, with a particular focus on the quantum annealing (QA) approach [21] for each of the two cases described above. QA is particularly suited for finding the ground state of an Ising problem, which is essentially analogous to finding the optimal solution to many binary combinatorial optimization problems of interest. This includes the traveling salesman and knapsack problem, as well as more niche applications in computational chemistry or graph theory [22–24]. In light of its range of applicability, the field has attracted considerable interest in recent years, a trend that has only been further fueled by the increasing accessibility of quantum annealers—programmable quantum computers designed for quantum annealing. While the largest gate based quantum computer (IBM) only has 433 quantum bits (i.e., qubits) [25], the largest annealer (D-Wave) has over 12 000 qubits [26].

There is an expectation in the community that QA might outperform classical algorithms for some applications. This has led to many empirical studies into the efficacy of QA for a wide range of problems [27–31], often with mixed results that lack a decisive consensus. As of yet, there may be more promise in using QA for approximate optimization based on a recent demonstration of a scaling advantage over the best classical algorithm, even though this was for a rather artificial optimization problem [32].

As such it seems the actual choice of problem to solve strongly influences any inferences made about the potential (or lack thereof) of quantum annealing. While there are many works in the existing literature that apply QA to a particular problem, the body of literature tackling the more fundamental question of whether QA even should be applied or not is very sparse. A recent attempt [33] to create a rigorous methodology for evaluating the viability of a potential use case for quantum computing aims to shed some light on the latter. However, it should be noted that the discussion remains very open ended, despite being of paramount importance particularly in the context of near term quantum hardware.

So why does this paper consider load balancing as the target application for QA? First, it should be evident that load balancing is a topic of relevance for almost all computational scientists regardless of discipline. This is particularly true as the exascale era of classical computing approaches and programmers strive towards utilizing more and more compute power. It is also still very much an area of active research [34–37], highlighting that the room for potential improvements to be made persists. Furthermore, as will be seen shortly, the problem lends itself to a very natural conversion into an Ising formulation suitable for quantum annealers. Although it is not always inherently apparent *a priori* how complex the solution energy landscape will be, it is at the very least very large and scales drastically with problem size.

Perhaps the most compelling rationale for selecting load balancing as an application for QA emerges when considering the broader HPC landscape. Classical computing, benefiting from several decades of development, significantly outpaces the emergent field of quantum computing in terms of maturity and scale when it comes to hardware. Despite the immense potential of quantum technology, it is unlikely to match the sheer scope of classical systems in the near future. Additionally, there is a growing consensus that classical computing will always retain some relevance, never being completely supplanted by its quantum counterpart. In this context, envisioning quantum computers as complementary accelerators, rather than standalone solutions, is a strategic and viable approach, particularly in load balancing, where recalculations are not constant but occur at regular or semiregular intervals, and hence quantum annealers can effectively augment the process by providing potentially better quality solutions. Moreover, this process is inherently streamlined by virtue of the fact that the classical resolution of the governing equations does not require an input from the load balancing subroutine at every time step and hence does not need to waste time waiting.

The proposal here is thus to integrate quantum annealers with classical HPC systems while identifying applications that are best placed to effectively leverage such combined systems. The primary computational tasks, whether in fluid dynamics, solid-state mechanics, biological flows, or some other algorithm, continue to run on classical HPC infrastructure. Meanwhile, the load balancing component is offloaded to an attached quantum processor operating in parallel. This concept mirrors the synergy between GPUs and CPUs [38], and aligns well with the growing narrative of heterogeneous HPC architectures. Such an approach not only capitalizes on

the strengths of quantum annealers but is also in tune with the trend [39–41] towards diversifying and optimizing computational resources in HPC environments.

The structure of the remainder of this paper is outlined as follows: first, an introductory review of quantum annealing is presented, alongside pertinent details from the classical methodologies under consideration. This encompasses an in-depth discussion on the acquisition of real-world classical data to be subsequently load balanced using QA. After which, the outcomes derived from both grid-based (AMR) and off-grid (SPH) simulations run on actual quantum annealing hardware are explored.

II. METHODS

A. Quantum annealing

Of the myriad possibilities [42] when it comes to implementing a quantum computer, currently the two leading paradigms are gate based QC [43,44] and adiabatic quantum computing (AQC) [45,46]. In principle the two approaches have been shown [47] to be equivalent if allowing a polynomial overhead. Gate based QC can be thought of as the quantum analog to classical computing where instead of logical gates acting on bits, the computation is performed by applying unitary gates to qubits. Alternatively, AQC relies on adiabatic time evolution of a quantum state from an initial, easy to prepare state to a final observed value as modeled by the Schrödinger equation.

Originally proposed as a way to tackle satisfiability problems [48], AQC has since received significant attention due to the ease with which many combinatorial optimization problems can be represented in Hamiltonian form [22]. The protocol operates under the premise of the adiabatic theorem [49], which states that a quantum system in its ground state remains in its ground state when acted upon by a perturbation if the changes to the Hamiltonian are slow enough and there is an energy gap between the ground and excited states. Therefore if an initial Hamiltonian is appropriately evolved into the problem Hamiltonian the final ground state should encode the solution to the desired problem. This is usually done by interpolating between a Hamiltonian with an easy to directly prepare ground state (H_A) and the problem Hamiltonian (H_B) to give the instantaneous value as described in Eq. (1):

$$H(t) = A(t)H_A + B(t)H_B. \tag{1}$$

The temporal prefactors ensure a smooth transition between the two Hamiltonians. Initially H_A dominates ($A(0) = 1, B(0) = 0$), prior to ceding to H_B over time ($A(T) = 0, B(T) = 1$). The rate of this change is crucial in preventing disruptive, diabatic changes to the system and the acceptable limits are dependent on the minimum energy gap of the problem [50,51].

A common choice for the initial Hamiltonian is a transverse field in the x direction:

$$H_A = \sum_{i \in V} \sigma_i^x, \tag{2}$$

with σ_i^x being the x-Pauli matrix acting on qubit i. This is eventually replaced by the problem/final Hamiltonian as described by

$$H_B = \sum_{i \in V} h_i \sigma_i^z + \sum_{(i,j) \in E} J_{ij} \sigma_i^z \sigma_j^z, \tag{3}$$

where $G(V, E)$ is the graph consisting of qubit nodes V and connective edges between neighboring qubits E. This Hamiltonian is characterized by the local field at the ith qubit, represented by h_i, as well as the interaction couplings between connected qubits denoted by J_{ij}. The instantaneous Hamiltonian during a computation, $H(t)$, of Eq. (1) is thus the well known transverse-field Ising Hamiltonian [52]. As the contribution of H_A fades over time, the quantum dynamics fade out and eventually result in a purely classical system where the qubits are measured in order to obtain the solution. The classical Ising model can be readily obtained by replacing the σ^z Pauli operators with classical spin variables:

$$H_{\text{Ising}} = \sum_i h_i s_i + \sum_{ij} J_{ij} s_i s_j. \tag{4}$$

Moreover by operating under the umbrella of the adiabatic theorem, the protocol is guaranteed to obtain the ground state solution, at least in theory. However, in practice the requirements of true adiabaticity and changing the Hamiltonian slowly enough can be excessively stringent. This is clearly evident in current architecture where thermal coupling with the environment is strong enough such that the timescales are of the order ≈10–100 ns [53]. Furthermore the requirements on how quickly the Hamiltonian can be evolved (i.e., the annealing schedule) are dependent on the gap between the ground state and first excited solution, which is not usually known *a priori*.

QA is thus a relaxation of these conditions that follows the same principles as AQC, but forms a distinctly separate algorithm, with the annealing schedule determined heuristically and strictly adiabatic conditions not guaranteed. The same time dependent Hamiltonian of Eq. (1) is used to evolve the quantum state, however the system can no longer accurately be modeled by the Schrödinger equation, but would require master-equation based descriptions. As a result, QA forms a heuristic quantum algorithm that should be viewed as a statistical sampler rather than a deterministic solver, with the intended goal of maintaining a relatively high probability of remaining in the ground state or to at least be sufficiently close. The theory of quantum annealing is much less developed than AQC, partially due to the more complex setting. However, some progress has been made, for example in the topic of diabatic computing which considers rapid quenches far from the adiabatic limit [54]. In particular, in this regime a mechanism related to energy conservation provides an important guarantee on average optimality [55].

There are actually many similarities between QA and a well known [56] classical counterpart known as simulated annealing (SA). The latter aims to mimic the physical process of a solid being slowly cooled so that the eventual "frozen" state contains the lowest energy solution to a desired cost function. In essence the approach relies on thermal fluctuations for exploration of the solution landscape while avoiding local minima. Meanwhile QA takes a very thematically similar approach but replaces thermal fluctuations with quantum

ones. H_A introduces disorder into the system with respect to the ground states of H_B and in doing so provides qubits with energetic variations during the anneal.

The expectation that QA might surpass classical algorithms is tied to the potential impact of quantum mechanical phenomena such as superposition, tunneling, and entanglement [31,45,57–59]. Fundamentally, quantum superposition and tunneling can enable transitions between states, even those separated by high energy barriers [60]. This suggests that a search algorithm utilizing QA could overcome local minima more easily by tunneling through these energy barriers as illustrated in Fig. 1.

QA is thus grounded in a model that more closely reflects the behavior of real quantum systems when compared to AQC. However, the potential for nonadiabatic effects complicates the distinction between its computational complexity and that of conventional computing. QA often involves the nonadiabatic evolution of mixed quantum states within an open system, and this approach is generally considered heuristic. Thus, it lacks the provable computational advantages that AQC provides [61].

The general approach to employing a quantum annealer shares many similarities across different architectures. This research, however, utilizes D-Wave systems, which consist of superconducting qubits. Although there are several promising alternatives [62–64], these technologies are comparatively nascent in their development. Meanwhile D-Wave is the largest and most commonly used commercial QA platform currently and has been used extensively for both industry and research purposes. For a detailed, step-by-step guide on using a quantum annealer, interested readers are encouraged to consult the numerous introductory articles available on this topic [31,61,65]. In summary, the primary steps involved in utilizing an annealer include problem formulation, minor embedding, and sampling (i.e., the anneal itself). Problem formulation and how the classical simulations were run are described next, with the influence of embedding and sampling discussed as part of the results.

B. Adaptive mesh refinement

In order to formulate load balancing for AMR as an Ising problem suitable for annealers, data were gathered using COMPREAL [66], a fully compressible, finite difference flow solver for the Navier-Stokes equations. The flow solver interfaces with a widely used general software framework for AMR applications called BOXLIB [16,67]. BOXLIB was designed as a platform upon which to build massively parallel software and has demonstrated good scalability on up to 100 000 cores [68]. Research codes based on BOXLIB are numerous and varied, but include the realms of astrophysics [69,70], computational cosmology [69], subsurface flow [71], and combustion [72] among others.

The particular test case simulated involves a developing spherical blast wave as illustrated in Fig. 2. The sharpest gradients are in a very thin zone encompassing the shock edge which is moving outwards. As such the mesh is regularly updated to track its position. Data are defined on a nested hierarchy of a logically rectangular collection of cells called grids (or patches). This is divided into levels where each level refers

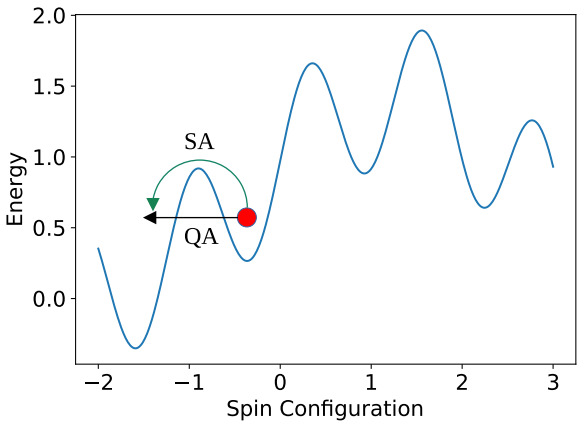


FIG. 1. When in a local minimum, QA can use quantum tunneling to directly escape. SA instead relies on thermal fluctuations to escape over the energy barrier.

to the union of all grids that share the same mesh spacing. Aside from the coarsest level, finer levels are disjoint and do not cover the entire simulation domain, thus facilitating allocation of resources to desired regions. Each grid is made up of a number of cells and this is not necessarily the same across all grids. The cells have high intraconnectivity within the same grid and lower interconnectivity to neighboring grids. Therefore, from a work distribution perspective, it is desirable to allocate whole grids to individual processors while trying to maintain roughly the same total cell count when possible.

To quantify the computational burden associated with each nested grid, the total number of cells within that patch is

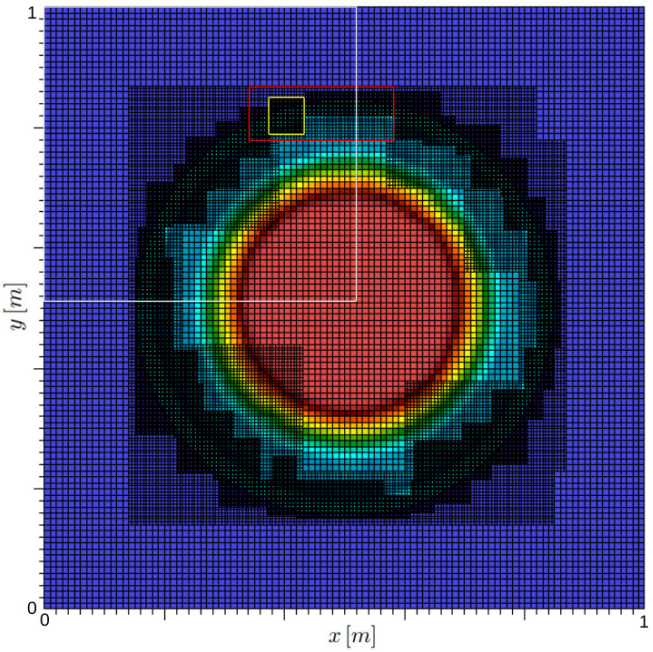


FIG. 2. A shockwave expanding outwards from a region initially at high pressure (red) to the surrounding lower pressure (blue) environment simulated using COMPREAL. The nested grid hierarchy is illustrated using individual grids from the finest, intermediate, and coarsest levels outlined in yellow, red, and white respectively.

counted. There are also alternative measures of "burden," for example actually recording the time it takes each patch to complete its allocated tasks might be more representative for multiphysics simulations where different cells solve different equations. This would affect the value of the weights but not the Ising formulation itself so the method remains completely general.

The classical simulation thus provides a series of numbers representing the notional cost of each grid, which needs to be equitably distributed across a given number of processors. This process will need to be repeated throughout the classical simulation at certain intervals. It is undesirable to split these grids due to their high intraconnectivity and so the problem essentially reduces to one of number partitioning. Given a set of N numbers, where N is the number of patches, $S = \{n_1, \ldots, n_N\}$, the task is to divide this set into two disjoint subsets such that the sum in both elements is the same, or at least minimizes the mismatch. This is framed as the following Ising model [22]:

$$H = A\left(\sum_{i=1}^{N} n_i s_i\right)^2, \tag{5}$$

where n_i are the numbers in the set (determined by counting the cells in each grid of Fig. 2), s_i is the Ising spin variable, and A is a general scaling constant which is set to unity henceforth. Despite the absence of linear biases in the model, the coupling terms will lead to the formation of a fully connected graph when mapped to the quantum processor. It is worthwhile noting that number partitioning phrased as a decision problem regarding if the two subsets are equal or not is classified as NP-complete [73]. However, since this model is based on real data from simulations, achieving a perfectly balanced split is essentially almost impossible. Therefore, the objective shifts to minimizing the discrepancy between subsets, a task that is NP-hard. This process can be applied recursively to achieve divisions for more than two processors. Currently BOXLIB uses a simple round robin (RR) strategy for distributing grids between subsets.

C. Smoothed particle hydrodynamics

This paper is not concerned with the technical details behind discretizing a set of governing equations using the SPH operator, as this is application specific and the interested reader can refer to the relevant literature [18,20]. Instead, the key takeaway here is the idea that this family of methods requires storage of particle data. Moreover these particles are irregularly positioned in space and move at each time step. Although this is a key strength of SPH as it allows particle aggregation in certain regions of interest and a sparse distribution in other areas, leveraging this benefit in practice is only viable with efficient memory and work allocation.

Compact support of the kernel ensures that particles are only influenced by neighbors that fall within the smoothing length. So it would be logical to split the domain into sections when assigning work to processors in order to limit interprocessor communication. However, as the particles are disordered it is important to remember some regions will be more densely populated than others, thus there is also the second avenue of intraprocessor communication to consider when creating a work distribution.

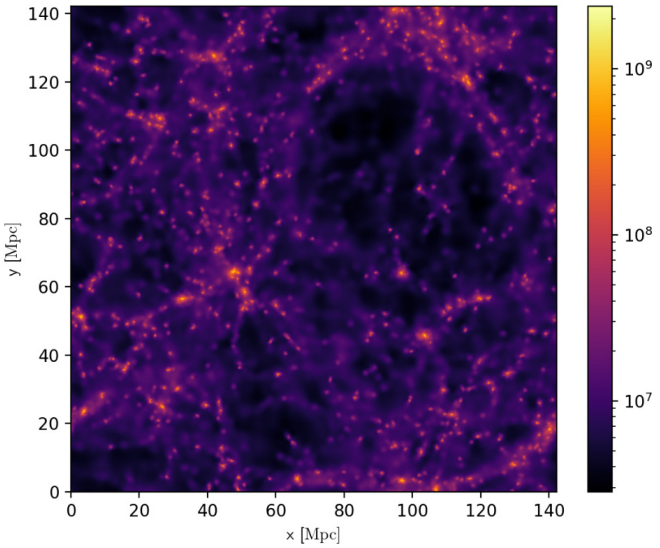


FIG. 3. A snapshot at time 13.7 Gyr of the projected mass for a small cosmological volume with dark matter simulated using SWIFT. Note that due to local gravity wells some regions will have a higher density of particles than others.

Classical data to be partitioned, by using QA in this paper, were obtained via SWIFT [74], a popular astrophysics code that has demonstrated good scalability on 100 000 cores. SWIFT relies on task based parallelism [75] in order to optimize shared-memory performance within each individual node as its backbone. This can then be readily generalized to create a work allocation across multiple, distributed memory nodes using graph partitioning algorithms. In summary, the domain is initially divided into a set of *cells*, with each cell containing a collection of particles such that if two particles interact they are either in the same cell or at most neighboring cells.

In terms of load balancing, this domain decomposition approach can be efficiently modeled as a graph, where each node symbolizes an individual cell containing some number of particles. The edges of this graph represent shared tasks reliant on particles located in separate cells that require communication between these cells. Moreover, tasks involving cells from different partitions must be processed by both respective processors, whereas tasks within the same partition are evaluated exclusively by the corresponding processor. An optimal load balancing strategy aims to minimize both the sum of node weights for each subset and the sum of edge weights bridging these subsets. The former goal is to reduce the waiting time caused by the slowest processor at each step, while the latter focuses on decreasing the necessary bandwidth for interprocessor communication. This dual objective ensures both efficient processing and minimal communication overhead.

Values for the weights were obtained by simulating a small cosmological volume with dark matter as found in the SWIFT example suite and illustrated in Fig. 3. The simulation involves 64^3 particles and the weights were extracted by timing the intracell/intercell tasks respectively for node/edge weights. The domain decomposition parameters were

modified by design to reduce the number of cells in order to accommodate the problem onto current annealers. Since the cells are cubes, in three dimensions this boils down to a minimum node/cell count of 27 since each cell has 6 face adjacent, 12 edge adjacent, and 8 corner adjacent neighbors. With all 26 neighbors residing within the one cell range defined earlier for shared tasks, this results in a fully connected graph.

The Ising model [22] for this load balancing problem places a spin variable $(-1/+1)$ on each node in order to determine whether that node will eventually belong in the "+" or "−" subset. The energy functional consists of two components:

$$H = \gamma H_1 + H_2, \tag{6}$$

where a Lagrange parameter (γ) has been introduced to allow changing γ in order to explore any conflicting effects. The model includes a penalty term when the node weight in set + is not equal to that in set −:

$$H_1 = \left(\sum_{n=1^N} w_i s_i \right)^2, \tag{7}$$

where the sum is across all nodes, each with weight w_i. Additionally there is a penalty term for each time an edge that connects nodes in different subsets is cut:

$$H_2 = \sum_{(uv)\in E} e_i \frac{1 - s_u s_v}{2}, \tag{8}$$

where the sum is across all edge connected nodes with e_i referring to the weight of the edge. This is a NP-hard problem [73] of significant value in ensuring many particle based codes retain a meaningful advantage on future HPC systems. Currently SWIFT relies on the graph partitioning software METIS [76].

III. RESULTS

A. Grid based application

We first demonstrate the potential of quantum annealing to recursively partition work from a grid based simulation across a range of processors. This is accompanied by insight into trends regarding solution quality and compared with classical strategies such as RR, steepest descent (SD), and SA. The interested reader is referred to Supplemental Material [77] for the details. While the aforementioned results explore the potential of quantum annealing under default D-Wave parameters and a single invocation of the D-Wave API, we focus here instead on providing a statistical analysis of the effects of varying these parameters. By systematically examining choices such as the number of anneals and embedding configurations, this analysis aims to quantify the robustness and consistency of quantum annealing outcomes under different parameter settings.

Although D-Wave functions in the minorminer library [78,79] effectively optimize the embedding of the logical problem onto the physical qubits, it is worthwhile noting that this algorithm is heuristic. A single call to the D-Wave API means that only one embedding configuration is utilized. Under ideal conditions, this alone should not significantly impact the results. However, for the ensuing statistical analysis, five distinct embedding configurations were precomputed and stored in order to map the problem to different physical qubits. When studying the effect of a parameter, for example number of anneals, each configuration underwent five separate runs using the precomputed embeddings prior to averaging. This aims to minimize potential biases in the hardware that might undesirably affect the outcomes.

Often only the lowest energy solution is considered from the range of possible configurations that arise from QA. However, this begs the important question of how likely this is to actually be observed in practice. Due to the inherent nature of load balancing requiring repeated redistribution after some number of time steps, it is not necessarily a stringent requirement to have the *best* possible solution each time. In fact, it may suffice to have something close enough.

Figure 4(a) attempts to illustrate this by plotting all the samples from a single run with 100 anneals for a small problem size of 50 grids during a single partition. The energy output from a QA is arbitrarily scalable, therefore the actual numerical value is of little consequence other than the trend of lower energies representing better solutions. This is in accordance with the corresponding solution disparity as defined in Eq. (9):

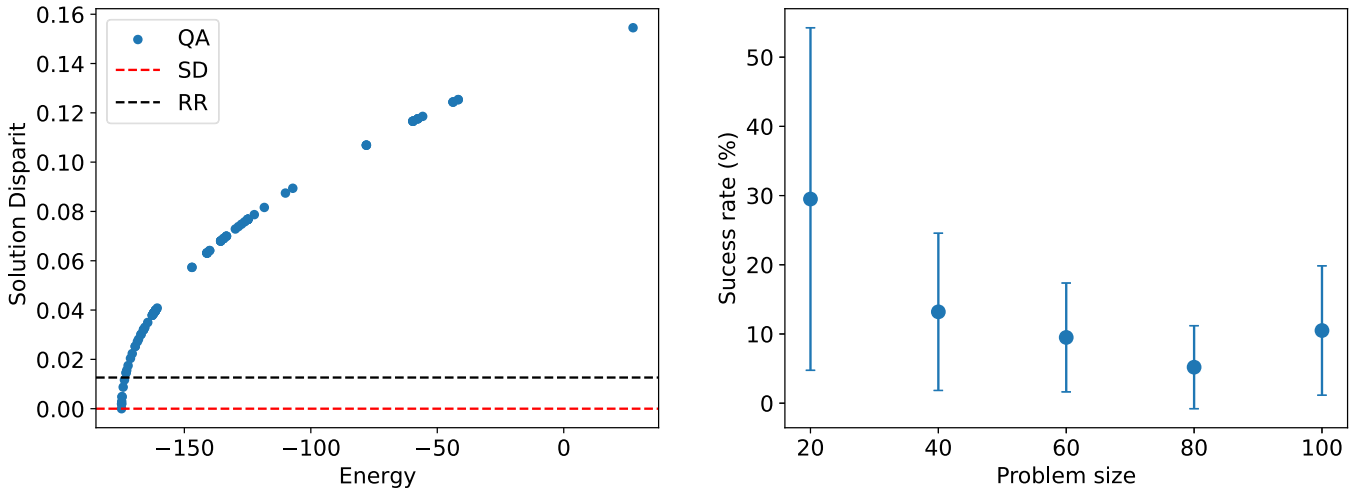
$$\text{Solution disparity} = \frac{|w_1 - w_2|}{0.5 \times (w_1 + w_2)}. \tag{9}$$

The solution disparity is thus the difference between assigned work loads for a single partition normalized by the work resulting from a perfect split. In other words, a lower solution disparity corresponds with better solution quality.

Included in Fig. 4(a) for comparison are the solution disparities obtained from the deterministic round robin and steepest descent methods. Even with such a small number of anneals, the best (i.e., lowest energy) quantum solution has the same degree of imbalance as SD, with both methods arriving at the optimum solution for this configuration. There is also a significant proportion of the annealing samples that perform worse than SD but better than RR. These are the blue markers in between the two dotted, horizontal lines and suggest some degree of resilience in the method despite being heuristic. The spread of points with a higher disparity than the black dotted line is clearly inferior and holds little value.

Figure 4(b) illustrates the variation in the percentage of QA solutions that represent an improvement over RR across different problem sizes, based on results from 100 anneals. It is evident that the frequency of suboptimal solutions escalates with the increase in problem size, indicating a relative underperformance compared to simulated annealing, which achieves a success rate of approximately 90% across this range of problem sizes. Despite this, the quality of the most optimal solution obtained from both QA and SA remains the same.

This observation suggests that while QA has the potential to match the performance of more sophisticated classical algorithms, it does so for a limited fraction of the generated solutions. Given that QA is inherently a probabilistic method, it naturally involves generating a large number of samples to increase the likelihood of obtaining a high-quality solution. The advantage of QA lies in its ability to rapidly produce

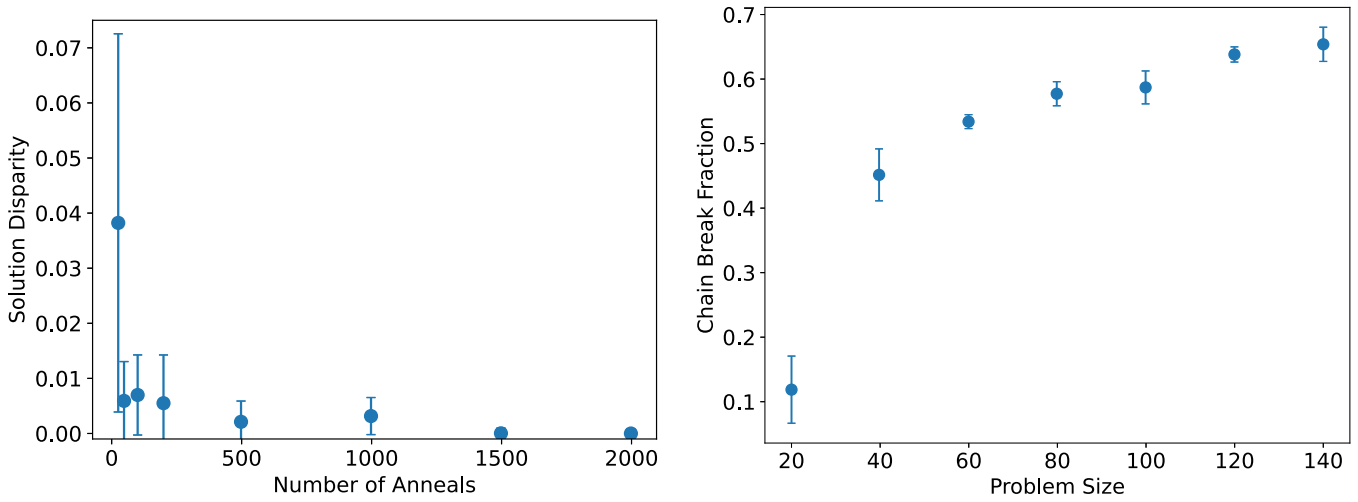


(a) (b)

FIG. 4. (a) Energy and solution disparity for all samples from a single call to the D-Wave API with 100 anneals for a problem size of 40 (QA), as well as solution disparity output from steepest descent (SD) and round robin (RR). A smaller solution disparity equates to a better solution quality. (b) Proportion of the 100 anneals that count as Success (i.e., better solution quality than RR) as a function of problem size.

samples, with each annealing process taking just microseconds. As a result, even a relatively low success rate can, in practice, almost guarantee the identification of at least one successful outcome, which is often the primary goal. This efficiency in sample generation underscores the practical viability of QA. Furthermore, it should be noted that at this point no *a priori* optimization for QA of user defined parameters such as number of anneals and chain strength have been provided to the annealer. Consequently, not only is the number of anneals very small here, but there is also a considerable number of chain breaks for problem sizes of 40 and more which can significantly degrade performance.

In order to evaluate the impact of parameter choices, the more challenging case with 100 grids is partitioned while changing the number of anneals. Figure 5(a) illustrates how the solution quality varies with anneals for a fixed problem size. Each configuration was repeated five times with a precomputed embedding and the average lowest energy solution is shown here with error bars reflecting its standard deviation. As expected, increasing the number of anneals increases the likelihood of finding a near optimum solution. Although the mean quality seems to plateau relatively early in terms of magnitude, increasing anneals has significant impact on reducing error margins and thus the reliability of obtaining said solution. Considering the relatively small number of anneals needed to drastically improve solution quality as well as the cheap computation cost of each anneal, it is evident that this is likely not a limiting factor.



(a) (b)

FIG. 5. (a) Impact of number of anneals on solution disparity for a fixed problem size of 100 grids. A lower solution disparity equates to a better quality solution, while negative values are not allowed by definition as per Eq. (9). (b) Mean chain break fraction as a function of increasing problem size (i.e., the number of grids being partitioned).

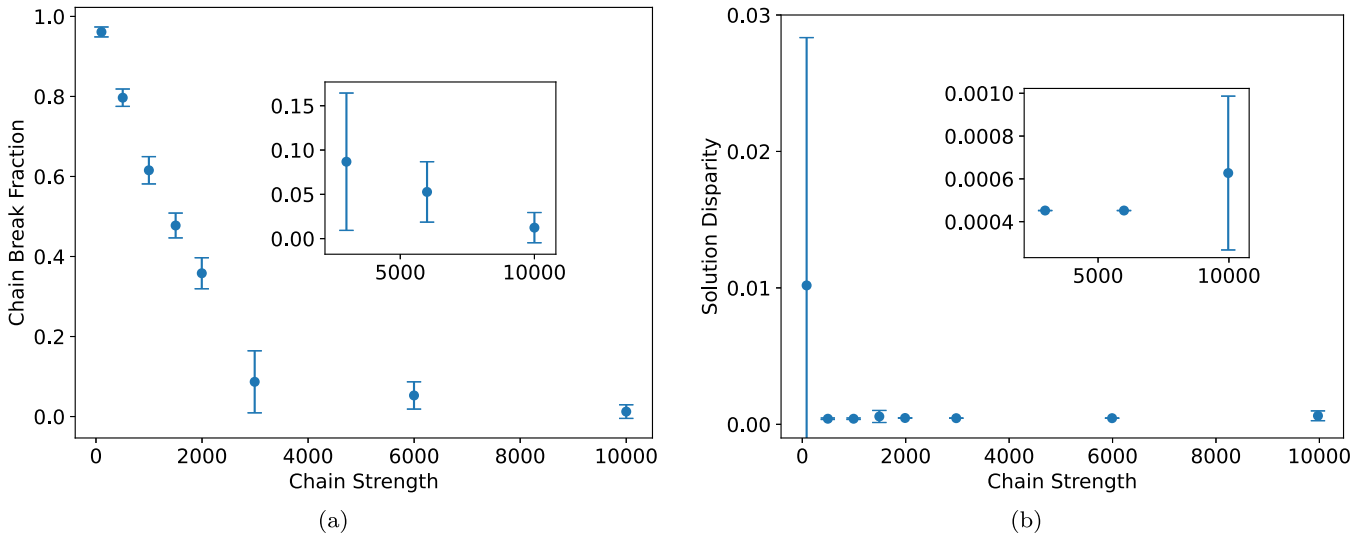


FIG. 6. (a) Mean chain break fraction as a function of chain strength for a fixed problem size of 100 grids. (b) Mean solution disparity (i.e., lower disparity equates to a better solution) as a function of chain strength for the same 100 grid problem. Insets include a focus view on the rightmost three points.

Thus, the main roadblock to achieving scalability is unlikely to be the number of required anneals, but is perhaps the susceptibility to chain breaks. The Ising model here forms a fully connected graph while annealing hardware has limited physical couplings. This results in minor embedding forming chains of physical qubits to represent the same logical qubit. However, it is evident in Fig. 5(b) that the default chain strength, calculated using uniform torque compensation, quickly becomes inadequate at larger problem sizes. The significant increase in chain break fraction (CBF) as the number of patches/grids requiring partitioning increases results in a decision problem that is by default resolved via majority voting. At values as high as those seen in Fig. 5(b) this has severe implications for the utility and robustness of the method for realistic applications where the number of patches/grids needing to be partitioned will be well over 100 if not in the thousands or more.

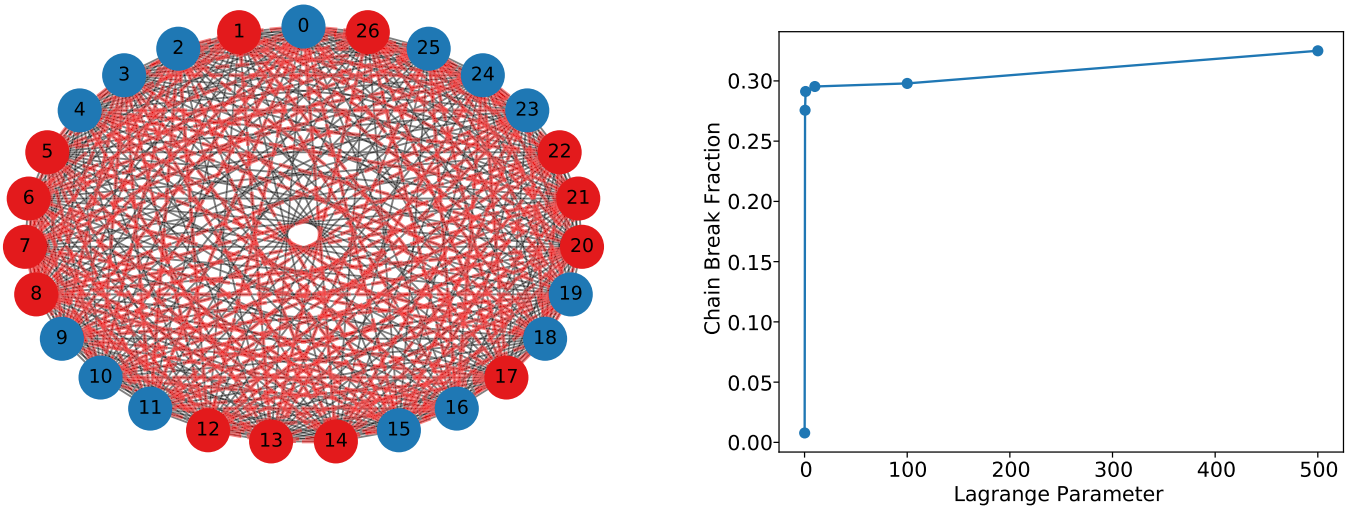
One solution to this is overriding the default chain strength scheme and manually setting stronger chains. This approach was explored in Fig. 6(a) for a fixed problem size involving 100 grids. The chain strength in the graph is defined by the multiplier applied to the maximum value in the data set. In other words, a chain strength of 1000 here means a value 1000 times larger than the maximum number in the set. It is evident that the value required to maintain an acceptable fraction of breaks is several orders of magnitude higher than one might have anticipated from the size of numbers being partitioned. It is unclear if it would have been possible to predict this *a priori* and how this trend would scale at realistic problem sizes.

It is also well known that blindly increasing the chain strength is not the perfect solution. Having qubits in very rigid chains can make them inflexible and less efficient at exploring the solution space. It can also wash out the other energy scales so that they are less than the device temperature and therefore lead to effectively random solutions. Furthermore, the energy penalty from the chain term can start rivaling the energy contribution from the actual Ising model if it grows too large and this can introduce a bias in the solution. Thus there is a balancing act to be performed, where clearly the default scheme is not adequate for the problem being considered here, yet too strong a chain is also undesirable. This is demonstrated in Fig. 6(b) where increasing the chain strength brings a large improvement in solution quality. However, with very strong chains the solution does begin to degrade again. The silver lining is that there seems to be a very large region of the state space that results in good solutions.

Taken together, the analysis here suggests that QA exhibits notable improvements when compared to basic classical algorithms like RR. Furthermore, QA demonstrates the capability to achieve solutions of comparable quality to those generated by more sophisticated algorithms, such as SA and SD. However, to attain such high-quality solutions consistently, QA may necessitate preliminary adjustments or tuning, highlighting the importance of optimizing the quantum annealing process for specific problem sets. It is important to acknowledge that the scope of problem sizes investigated in this paper is constrained by the limitations of current quantum computing hardware. As hardware capabilities improve, it is anticipated that these limitations will decrease, thereby expanding the range of problem sizes that can be efficiently addressed. The complexity of the energy landscape in this section is also limited, whereby advanced classical methods are able to arrive at near optimum solutions without being excessively hindered by local minima traps. We anticipate QA will be more advantageous over classical approaches for more complex energy landscapes with deep local minima traps. This highlights a crucial aspect of quantum annealing: identifying and formulating problem sets that truly leverage the unique strengths of quantum algorithms to solve problems that are intractable for classical methods.

B. Particle based application

The graph representing load balancing for SPH is incredibly dense as illustrated in Fig. 7(a) due to its fully connected



(a) (b)

FIG. 7. (a) The 27 node, fully connected load balancing graph for SWIFT. Blue and red circles represent nodes in different subsets, and red/black lines represent cut/uncut edges following a partition. (b) Change in chain break fraction as a function of Lagrange parameter. The Lagrange parameter determines the relative importance between penalty terms in the Ising model. A larger value places more significance on minimizing the difference in node weights, while a smaller value signifies greater value in minimizing cut edge weights.

nature. An optimized partition should aim to minimize the mismatch between subsets of total node weight, while also minimizing the sum of cut edge weights. It is not a clear *a priori* which of the two objectives is more important as this is likely to be somewhat dependant on the HPC architecture, in particular whether intraprocessor or interprocessor bandwidth is the more limiting factor for the CPU stack. For example, in the case of the latter, it would be more strategic to further minimize cut edge weight where possible even at the cost of a slightly higher node imbalance and vice versa for the former. Therefore in order to remain general, the task will be considered a multiobjective optimization problem here.

The Lagrange parameter, γ, determines the relative importance between these two objectives as shown in Eq. (6). A high value implies more significance attributed to minimizing the difference of node weights between the two subsets. In the limit of very large γ the problem essentially reduces back to number partitioning since the edges will have negligible relative influence. An interesting finding was that changing this parameter had an unexpected impact on the chain break fraction. This is shown in Fig. 7(b) where a small value results in lower chain breaks than was observed in the grid based, number partitioning example even for a graph of the same size. Indeed, as the Lagrange parameter is increased, the problem tends back towards number partitioning and chain breaks become more frequent. This is despite no change to the underlying graph structure, which remains a 27 node clique. Moreover, numerical values are autoscaled in the process of mapping onto a quantum processor and so differences in the values of weights alone should not be significantly impacting chain breaks. This implies that despite not changing the nature of the underlying embedding, the load balancing problem here is intrinsically more resilient to chain breaks than for its grid based counterpart.

Consider for now a neutral value for the Lagrange parameter of unity. A single QA was conducted with 1000 anneals and the lowest energy solution compared to a partition obtained using METIS, a state of the art classical graph partitioning software. This is illustrated in Fig. 8, which displays the cut edge matrices as heat maps. Each entry in the matrix represents the edge connecting nodes with the corresponding axis indices, while the color intensity is indicative of the weight of said edge. A color value of null (i.e., dark purple) indicates that the edge was not cut. Both QA and METIS make an exact total of 182 cuts each and share the same magnitude in terms of single largest cut. However, QA consistently makes better choices in exploring the energy landscape and manages to sever less expensive edges for a combined saving of close to 33%. The combined weight of cut edges from this QA run was only 66% the size of its METIS counterpart, indicating a drastic reduction in the amount of required communication between processors. Furthermore, this came with a better node balance as well where the degree of imbalance using QA was only 34% the size of imbalance allocated by METIS.

Repeating the QA runs five times and averaging, the same trend still stands as indicated in Table I, suggesting a resilience of the method to probabilistic fluctuations. Note that node and edge weights have been normalized by the same factor for both methods as they operate on the same data set. Furthermore the performance ratio entries in the table are simply the fractional result of dividing the corresponding METIS entry by the QA counterpart. As both objectives are

TABLE I. Average performance statistics for quantum annealing compared with METIS. The ideal solution has as small a value as possible for both solution imbalance and cut edge weights.

	Solution disparity	Cut edge weights
Quantum annealing	0.057	3.69
METIS	0.189	5.20
Performance ratio	3.32	1.41

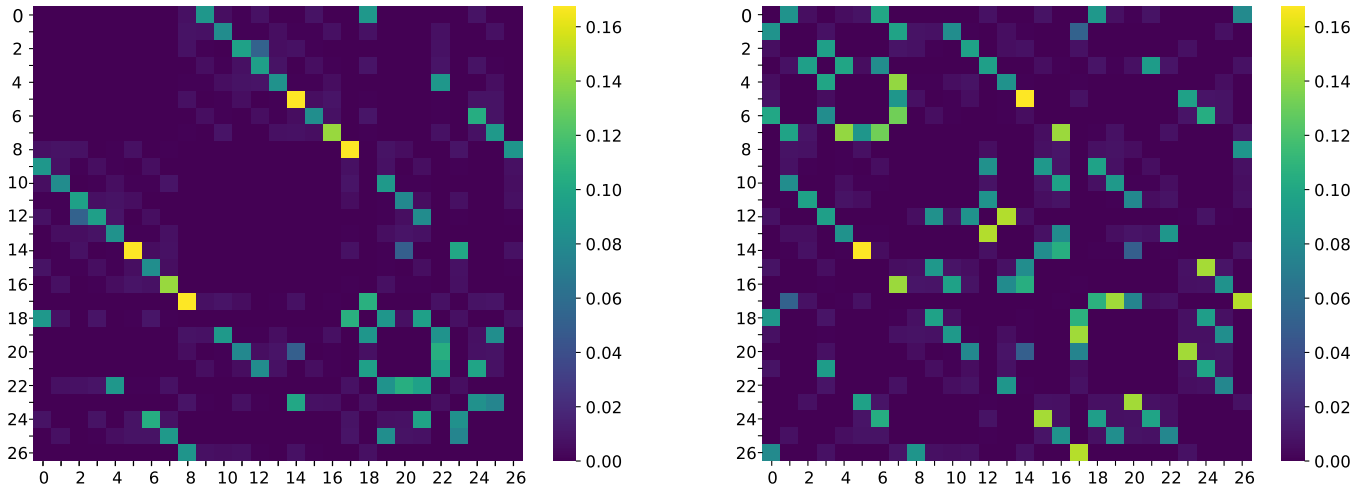


FIG. 8. Heat maps for the partitions from QA and METIS respectively. Each entry in the matrix represents an edge between two nodes with the corresponding indices along the outer edges. The color intensity indicates the normalized weight of a cut edge. A higher proportion of low-weight entries (i.e., light purple) implies a better partition as opposed to more highly weighted cut edges (i.e., green/yellow).

tailored to be minimized, a larger than unity performance ratio indicates some quantum advantage. Moreover QA performs better across both objectives simultaneously.

In addition to this, the outcomes of quantum annealing can be further fine tuned to meet the specific needs of the HPC cluster by adjusting the Lagrange parameter. To maintain a broad applicability, this paper extensively explored the parameter's state space through 100 iterations with 1000 anneals each and evenly spaced values for the Lagrange parameter between 0 and 50. The Pareto dominant solutions in terms of the two objective functions obtained from this process are presented in Fig. 9. The Pareto front here represents a set of

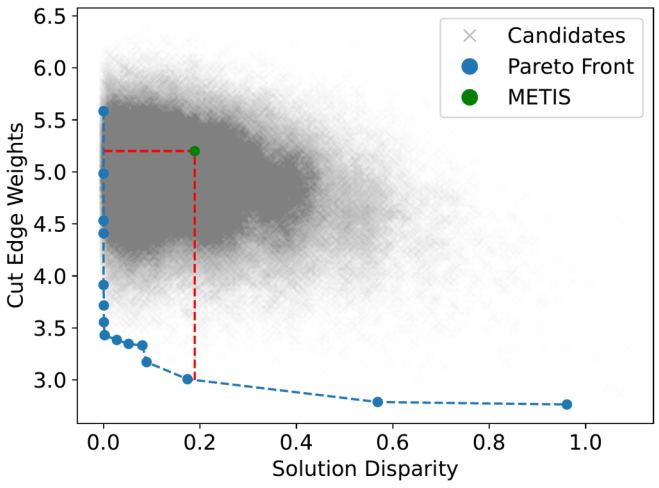


FIG. 9. Approximate Pareto front as mapped out by QA for optimizing (i.e., minimizing) cut edge weight and solution disparity. Included are all the other solutions obtained by QA as well as the output from METIS. A significant proportion (close to 41%) of QA samples that are suboptimal to the Pareto front are still Pareto dominant when compared to METIS. These are the points to the left and below the METIS solution as indicated by the area encapsulated by red dashed lines and the Pareto front.

solutions for which one objective function cannot be improved without bringing about a detriment to the second objective. The right side of the Pareto front is fairly close to horizontal, suggesting that if one approaches from the far right, vast improvements can be made in terms of solution disparity at the cost of a very minimal increase to cut edge weights. Conversely, initially starting at the left and proceeding towards the right large improvements to the cut edges are attainable at little cost to the node balance. The solution from METIS is also included for comparison and evidently inferior in that it is clearly possible to improve both objectives. Moreover the latter is true not just for a handful of lucky anneals, but for a large proportion as observed with the collection of QA points that lie off of the Pareto front but still are Pareto dominant in relation to the METIS output. This corresponds to close to 41 000 of the total 100 000 sample solutions and demonstrates the resiliency of QA for complex problems. Even outside this region, most candidate solutions are only inferior to METIS in one objective, likely as a result of enforcing either very large or very small Lagrange parameters. This will naturally prioritize one objective even at the potential detriment of the other.

IV. CONCLUSIONS AND OUTLOOK

This paper explored the utilization of quantum annealing as a strategic approach for the critical task of workload allocation to processors in parallel HPC applications using real data. Initially focusing on less complex, grid-based applications, QA demonstrated improvements over simpler classical strategies, yet it did not conclusively surpass more sophisticated classical methods. A significant challenge for QA, unlike these classical methods, is the limited connectivity between physical qubits. This limitation results in a critical barrier to scalability, particularly given the fully connected nature of the problem at hand.

However, this drawback is somewhat mitigated in the context of the second application, which focused on load

balancing in particle based codes. While the smallest problem configuration which was studied here is fully connected due to each cell's proximity within the system, this is not expected to hold for larger problems. Recall that SPH kernels are characterized by compact support; this implies that larger problem graphs, while more extensive, are not likely to be fully connected. Thus, it is likely to be more efficient to embed larger problem sizes onto quantum annealing hardware. This subtlety enhances the feasibility of applying QA, especially in the noisy intermediate scale quantum era where interqubit connectivity can sometimes be a more critical constraint than the total number of qubits. Moreover, for simpler two-dimensional/quasiplanar configurations this might not be as severe a constraint even for larger problem sizes. This is due to the compatibility between the problem graph and alignment of qubits with their couplers, which when implemented on a chip are inherently quasiplanar as well.

In addition, QA demonstrated an improvement in performance even against a state of the art classical method, despite the problem's evolution into a complex multiobjective optimization. The Lagrange parameter formulation allows the user to explore the Pareto front and obtain highly optimized solutions tailored towards individual hardware architectures. Although this necessitated running a representative simulation and partitioning it repeatedly to explore the state space here, in general more cost effective methods such as machine learning could be used to estimate a good value *a priori*. This approach draws strong parallels with existing efforts that have endeavored to do so for some of the other annealing parameters [80,81]. The Ising formulation can also be readily extended to allow concurrent partitioning into multiple subsets simultaneously rather than recursively [24] to better accommodate larger problems.

Given the observed enhancements in solution quality and the reduced connectivity demands in larger problem graphs, it is conceivable that as annealers continue to evolve, they may become viable for graph partitioning tasks such as this. This is particularly relevant for hybrid applications such as load balancing where the majority of the algorithm could still be executed on CPUs, with only a select, complex segment offloaded to the quantum processor in tandem. As such, perhaps a heterogeneous architecture integrating both quantum and classical computing resources could be a strategic path forward, particularly so in light of recent endeavors to strive towards such integrated systems [82,83]. Moreover, one of the most time consuming components of the QA algorithm is currently the communication time between classical and quantum hardware and associated latency costs. This would be crucially reduced to almost negligible levels by colocating the respective chips in the same data center. The time for each anneal itself remains competitive at around 20 μs compared to the roughly 15 μs needed by METIS and is likely to improve with evolving hardware. However, this will be of comparatively little consequence as colocation will allow the system to run in parallel, thus best leveraging the inherent strengths of each respective system.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Peter Draper at Durham University and Dr. Matthieu Schaller at Leiden University for insightful discussions. Similarly, the authors are grateful to Prof. Vivien Kendon at University of Strathclyde for useful comments and discussion on this paper, as well as the rest of the QEVEC/QuANDiE teams. The authors acknowledge funding through UKRI Engineering and Physical Sciences Research Council Projects No. EP/W00772X/2 and No. EP/Y004515/1. This work used the DiRAC@Durham facility managed by the Institute for Computational Cosmology on behalf of the STFC DiRAC HPC Facility [84]. The equipment was funded by BEIS capital funding via STFC capital Grants No. ST/K00042X/1, No. ST/P002293/1, No. ST/R002371/1, and No. ST/S002502/1, Durham University and STFC operations Grant No. ST/R000832/1. DiRAC is part of the National e-Infrastructure.

O.R. formulated the application configurations, executed all simulations, and wrote the initial draft. All authors contributed to the core ideas and reviewed the paper.

DATA AVAILABILITY

The data and software supporting this paper are available upon reasonable request from the corresponding author (O.R). The majority of the code, including the datasets themselves and partitioning algorithms, is publicly accessible on GitHub [85].

[1] A. Danowitz, K. Kelley, J. Mao, J. P. Stevenson, and M. Horowitz, Cpu db: Recording microprocessor history: With this open database, you can mine microprocessor trends over the past 40 years., Queue **10**, 10 (2012).

[2] H. Sutter *et al.*, The free lunch is over: A fundamental turn toward concurrency in software, Dr. Dobb's journal **30**, 202 (2005).

[3] W. Tan, P. Lin, B. Liang, and H. Deng, Influence of network bandwidth on parallel computing performance with intra-node and inter-node communication, in *2009 Second International Conference on Intelligent Networks and Intelligent Systems* (IEEE, New York, 2009), pp. 534–537.

[4] R. Rabenseifner, Hybrid parallel programming on HPC platforms, in *Proceedings of the Fifth European Workshop on OpenMP, EWOMP* (Aachen, Germany, 2003), Vol. 3, pp. 185–194.

[5] H. A. Schwarz, Ueber einige Abbildungsaufgaben, J. Reine Angew. Math. **70**, 105 (1869).

[6] K. A. Hessenius and T. H. Pulliam, A zonal approach to solution of the euler equations, in *Proceedings of the Third Joint Thermophysics, Fluids, Plasma, and Heat Transfer Conference* (1982), p. 969.

[7] W. D. Henshaw and G. Chesshire, Multigrid on composite meshes, SIAM J. Sci. Stat. Comput. **8**, 914 (1987).

[8] R. Glowinski, Q. Dinh, and J. Periaux, Domain decomposition methods for nonlinear problems in fluid dynamics, Comput. Methods Appl. Mech. Eng. **40**, 27 (1983).

[9] D. E. Keyes, L. C. McInnes, C. Woodward, W. Gropp, E. Myra, M. Pernice, J. Bell, J. Brown, A. Clo, J. Connors *et al.*, Multiphysics simulations: Challenges and opportunities, Int. J. High Perform. Comput. Appl. **27**, 4 (2013).

[10] G. Houzeaux, J. Cajas, M. Discacciati, B. Eguzkitza, A. Gargallo-Peiró, M. Rivero, and M. Vázquez, Domain decomposition methods for domain composition purpose: Chimera, overset, gluing and sliding mesh methods, Arch. Computat. Methods. Eng. **24**, 1033 (2017).

[11] H. Tang, R. Haynes, and G. Houzeaux, A review of domain decomposition methods for simulation of fluid flows: Concepts, algorithms, and applications, Arch. Computat. Methods. Eng. **28**, 841 (2021).

[12] P. Wang, T. Abel, and R. Kaehler, Adaptive mesh fluid simulations on gpu, New Astronomy **15**, 581 (2010).

[13] J. Bennett and M. Botkin, Structural shape optimization with geometric description and adaptive mesh refinement, AIAA J. **23**, 458 (1985).

[14] L. Botti, M. Piccinelli, B. Ene-Iordache, A. Remuzzi, and L. Antiga, An adaptive mesh refinement solver for large-scale simulation of biological flows, Int. J. Numer. Method. Biomed. Eng. **26**, 86 (2010).

[15] G. Zumbusch, *Parallel Multilevel Methods: Adaptive Mesh Refinement and Loadbalancing* (Springer, New York, 2012).

[16] A. Dubey *et al.*, A survey of high level frameworks in block-structured adaptive mesh refinement packages, J. Parallel Distrib. Comput. **74**, 3217 (2014).

[17] R. A. Gingold and J. J. Monaghan, Smoothed particle hydrodynamics: theory and application to non-spherical stars, Mon. Not. R. Astron. Soc. **181**, 375 (1977).

[18] J. J. Monaghan, Smoothed particle hydrodynamics and its diverse applications, Annu. Rev. Fluid Mech. **44**, 323 (2012).

[19] M. S. Shadloo, G. Oger, and D. Le Touzé, Smoothed particle hydrodynamics method for fluid flows, towards industrial applications: Motivations, current state, and challenges, Comput. Fluids **136**, 11 (2016).

[20] S. J. Lind, B. D. Rogers, and P. K. Stansby, Review of smoothed particle hydrodynamics: Towards converged lagrangian flow modelling, Proc. R. Soc. A **476**, 20190801 (2020).

[21] T. Kadowaki and H. Nishimori, Quantum annealing in the transverse Ising model, Phys. Rev. E **58**, 5355 (1998).

[22] A. Lucas, Ising formulations of many np problems, Front. Phys. **2**, 5 (2014).

[23] B. Camino, J. Buckeridge, P. Warburton, V. Kendon, and S. Woodley, Quantum computing and materials science: A practical guide to applying quantum annealing to the configurational analysis of materials, J. Appl. Phys. **133**, 221102 (2023).

[24] H. Ushijima-Mwesigwa, C. F. Negre, and S. M. Mniszewski, Graph partitioning using quantum annealing on the d-wave system, in *Proceedings of the Second International Workshop on Post Moores Era Supercomputing*, 2017 (unpublished), pp. 22–29.

[25] H. Collins and C. Nay, IBM unveils 400 qubit-plus quantum processor and next-generation IBM quantum system two (2022), https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two.

[26] D.-W. Q. Inc., D-Wave announces 1,200+ qubit advantage2™ prototype in new, lower-noise fabrication stack, demonstrating 20x faster time-to-solution on important class of hard optimization problems, Press Release, 2024.

[27] H. G. Katzgraber, F. Hamze, and R. S. Andrist, Glassy chimeras could be blind to quantum speedup: Designing better benchmarks for quantum annealing machines, Phys. Rev. X **4**, 021008 (2014).

[28] H. G. Katzgraber, F. Hamze, Z. Zhu, A. J. Ochoa, and H. Munoz-Bauza, Seeking quantum speedup through spin glasses: The good, the bad, and the ugly, Phys. Rev. X **5**, 031026 (2015).

[29] S. Boixo, V. N. Smelyanskiy, A. Shabani, S. V. Isakov, M. Dykman, V. S. Denchev, M. H. Amin, A. Y. Smirnov, M. Mohseni, and H. Neven, Computational multiqubit tunnelling in programmable quantum annealers, Nat. Commun. **7**, 10327 (2016).

[30] V. S. Denchev, S. Boixo, S. V. Isakov, N. Ding, R. Babbush, V. Smelyanskiy, J. Martinis, and H. Neven, What is the computational value of finite-range tunneling? Phys. Rev. X **6**, 031015 (2016).

[31] S. Yarkoni, E. Raponi, T. Bäck, and S. Schmitt, Quantum annealing for industry applications: Introduction and review, Rep. Prog. Phys. **85**, 104001 (2022).

[32] H. M. Bauza and D. A. Lidar, Scaling advantage in approximate optimization with quantum annealing, arXiv:2401.07184.

[33] N. Chancellor, R. Cumming, and T. Thomas, Toward a standardized methodology for constructing quantum computing use cases, arXiv:2006.05846.

[34] J. A. Mena, O. Shaaban, V. Lopez, M. Garcia, P. Carpenter, E. Ayguad, and J. Labarta, Transparent load balancing of mpi programs using ompss-2@ cluster and dlb, in *Proceedings of the 51st International Conference on Parallel Processing (ICPP)* (2022) (unpublished).

[35] G. Zhu, J. Hughes, S. Zheng, and D. Greaves, A novel mpi-based parallel smoothed particle hydrodynamics framework with dynamic load balancing for free surface flow, Comput. Phys. Commun. **284**, 108608 (2023).

[36] A. Mohammed, A. Cavelan, F. M. Ciorba, R. M. Cabezón, and I. Banicescu, Two-level dynamic load balancing for high performance scientific applications, in *Proceedings of the 2020 SIAM Conference on Parallel Processing for Scientific Computing* (SIAM, Philadelphia, PA, 2020), pp. 69–80.

[37] K. G. Miller, R. P. Lee, A. Tableman, A. Helm, R. A. Fonseca, V. K. Decyk, and W. B. Mori, Dynamic load balancing with enhanced shared-memory parallelism for particle-in-cell codes, Comput. Phys. Commun. **259**, 107633 (2021).

[38] B. Liu, D. Zydek, H. Selvaraj, and L. Gewali, Accelerating high performance computing applications: Using cpus, gpus, hybrid cpu/gpu, and fpgas, in *2012 13th International Conference on Parallel and Distributed Computing, Applications and Technologies* (IEEE, New York, 2012), pp. 337–342.

[39] G. Chen, G. Li, S. Pei, and B. Wu, High performance computing via a gpu, in *2009 First International Conference on Information Science and Engineering* (IEEE, New York, 2009), pp. 238–241.

[40] D. Tiwari *et al.*, Understanding gpu errors on large-scale hpc systems and the implications for system design and operation, in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)* (IEEE, New York, 2015), pp. 331–342.

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

[42] N. P. De Leon, K. M. Itoh, D. Kim, K. K. Mehta, T. E. Northup, H. Paik, B. Palmer, N. Samarth, S. Sangtawesin, and D. W. Steuerman, Materials challenges and opportunities for quantum computing hardware, Science **372**, eabb2823 (2021).

[43] S. Kwon, A. Tomonaga, G. Lakshmi Bhai, S. J. Devitt, and J.-S. Tsai, Gate-based superconducting quantum computing, J. Appl. Phys. **129**, 041102 (2021).

[44] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University, New York, 2010).

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

[46] W. Van Dam, M. Mosca, and U. Vazirani, How powerful is adiabatic quantum computation? in *Proceedings 42nd IEEE Symposium on Foundations of Computer Science* (IEEE, New York, 2001), pp. 279–287.

[47] D. Aharonov, W. Van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev, Adiabatic quantum computation is equivalent to standard quantum computation, SIAM Rev. **50**, 755 (2008).

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

[49] T. Kato, On the adiabatic theorem of quantum mechanics, J. Phys. Soc. Jpn. **5**, 435 (1950).

[50] S. Jansen, M.-B. Ruskai, and R. Seiler, Bounds for the adiabatic approximation with applications to quantum computation, J. Math. Phys. **48**, 102111 (2007).

[51] A. Elgart and G. A. Hagedorn, A note on the switching adiabatic theorem, J. Math. Phys. **53**, 102202 (2012).

[52] R. Stinchcombe, Ising model in a transverse field. i. basic theory, J. Phys. C **6**, 2459 (1973).

[53] A. D. King *et al.*, Coherent quantum annealing in a programmable 2,000 qubit Ising chain, Nat. Phys. **18**, 1324 (2022).

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

[55] 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).

[56] D. Bertsimas and J. Tsitsiklis, Simulated annealing, Stat. Sci. **8**, 10 (1993).

[57] R. Yaacoby, N. Schaar, L. Kellerhals, O. Raz, D. Hermelin, and R. Pugatch, Comparison between a quantum annealer and a classical approximation algorithm for computing the ground state of an ising spin glass, Phys. Rev. E **105**, 035305 (2022).

[58] E. Starchl and H. Ritsch, Unraveling the origin of higher success probabilities in quantum annealing versus semi-classical annealing, J. Phys. B **55**, 025501 (2022).

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

[60] M. Razavy, *Quantum Theory of Tunneling* (World Scientific, Singapore, 2013).

[61] E. K. Grant and T. S. Humble, Adiabatic quantum computing and quantum annealing, in *Oxford Research Encyclopedia of Physics* (Oxford University, New York, 2020).

[62] P. Scholl *et al.*, Quantum simulation of 2d antiferromagnets with hundreds of Rydberg atoms, Nature (London) **595**, 233 (2021).

[63] W. Lechner, P. Hauke, and P. Zoller, A quantum annealing architecture with all-to-all connectivity from local interactions, Sci. Adv. **1**, e1500838 (2015).

[64] S. Ebadi *et al.*, Quantum phases of matter on a 256-atom programmable quantum simulator, Nature (London) **595**, 227 (2021).

[65] D. de Falco and D. Tamascelli, An introduction to quantum annealing, RAIRO-Theor. Inf. Appl. **45**, 99 (2011).

[66] O. Rathore and S. Navarro-Martinez, Flame dynamics modelling using artificially thickened models, Flow, Turbul. Combust. **111**, 897 (2023).

[67] J. Bell, A. Almgren, V. Beckner, M. Day, M. Lijewski, A. Nonaka, and W. Zhang, Boxlib user's guide, https://github.com/BoxLib-Codes/BoxLib (2012).

[68] W. Zhang, A. Almgren, M. Day, T. Nguyen, J. Shalf, and D. Unat, Boxlib with tiling: An adaptive mesh refinement software framework, SIAM J. Sci. Comput. **38**, S156 (2016).

[69] A. S. Almgren, V. E. Beckner, J. B. Bell, M. Day, L. H. Howell, C. Joggerst, M. Lijewski, A. Nonaka, M. Singer, and M. Zingale, Castro: A new compressible astrophysical solver. I. hydrodynamics and self-gravity, The Astrophysical Journal **715**, 1221 (2010).

[70] A. Nonaka, A. Almgren, J. Bell, M. Lijewski, C. Malone, and M. Zingale, Maestro: An adaptive low mach number hydrodynamics algorithm for stellar flows, The Astrophysical Journal Supplement Series **188**, 358 (2010).

[71] G. S. Pau, A. S. Almgren, J. B. Bell, and M. J. Lijewski, A parallel second-order adaptive mesh algorithm for incompressible flow in porous media, Philos. Trans. R. Soc. A **367**, 4633 (2009).

[72] M. S. Day and J. B. Bell, Numerical simulation of laminar reacting flows with complex chemistry, Combust. Theory Model. **4**, 535 (2000).

[73] R. M. Karp, *Reducibility Among Combinatorial Problems* (Springer, New York, 2010).

[74] M. Schaller *et al.*, Swift: A modern highly-parallel gravity and smoothed particle hydrodynamics solver for astrophysical and cosmological applications, MNRAS **530**, 2378 (2024).

[75] M. Schaller, P. Gonnet, A. B. Chalk, and P. W. Draper, Swift: Using task-based parallelism, fully asynchronous communication, and graph partition-based domain decomposition for strong scaling on more than 100,000 cores, in *Proceedings of the Platform for Advanced Scientific Computing Conference*, 2016 (unpublished), pp. 1–10.

[76] G. Karypis and V. Kumar, Metis: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices, Technical Report (1997).

[77] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevResearch.7.013067 for partitioning across multiple processors and further comparison with classical methods.

[78] J. Cai, W. G. Macready, and A. Roy, A practical heuristic for finding graph minors, arXiv:1406.2741.

[79] S. Zbinden, A. Bärtschi, H. Djidjev, and S. Eidenbenz, Embedding algorithms for quantum annealers with chimera and pegasus connection topologies, in *International Conference on High Performance Computing* (Springer, New York, 2020), pp. 187–206.

[80] A. Barbosa, E. Pelofske, G. Hahn, and H. N. Djidjev, Opti-

mizing embedding-related quantum annealing parameters for reducing hardware bias, in *Parallel Architectures, Algorithms and Programming: 11th International Symposium, PAAP 2020, Shenzhen, China, December 28–30, 2020, Proceedings 11* (Springer, New York, 2021), pp. 162–173.

[81] A. Barbosa, E. Pelofske, G. Hahn, and H. N. Djidjev, Using machine learning for quantum annealing accuracy prediction, Algorithms **14**, 187 (2021).

[82] G. Cavallaro, M. Riedel, T. Lippert, and K. Michielsen, Hybrid quantum-classical workflows in modular supercomputing architectures with the JULICH unified infrastructure for quantum computing, in *IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium* (IEEE, New York, 2022), pp. 4149–4152.

[83] T. Lippert and K. Michielsen, *Publication Series of the John von Neumann Institute for Computing (NIC) NIC Series*, (Forschungszentrum Jülich GmbH Zentralbibliothek, Verlag, Jülich, 2022), Vol. 51, pp. 3–23.

[84] www.dirac.ac.uk.

[85] https://github.com/gtgr48/Load-Balancing-For-High-Performance-Computing-Using-Quantum-Annealing.git.