Quantum Information Processing (2026) 25:240
https://doi.org/10.1007/s11128-026-05263-5

# Accuracy and performance evaluation of quantum, classical and hybrid solvers for the Max-Cut problem

**Jaka Vodeb**[1,2] · **Vid Eržen**[2] · **Timotej Hrga**[3,4] · **Janez Povh**[4]

Received: 8 May 2025 / Accepted: 15 June 2026 / Published online: 3 July 2026


## Abstract

This paper benchmarks quantum, classical, and hybrid solvers on NP-hard Max-Cut and QUBO problems, emphasizing solution quality relative to known global optima. We evaluate D-Wave's fast annealing QPU and Hybrid solver against classical simulated annealing (SA) and Toshiba's simulated bifurcation machine (SBM) using 139 Max-Cut instances (100 to 10,000 nodes). For small instances ($\leq$ 250 nodes) with known global optima, Hybrid and SA consistently achieve optimal solutions, outperforming the QPU. For larger instances, SBM and slower SA yield superior solutions, while Hybrid and faster SA perform less effectively. Computation time varies across solvers.

**Keywords** Max-Cut · QUBO · Quantum annealing · Simulated annealing · Global optimum

**Mathematics Subject Classification** 68Q12 · 90C27 · 90C26 · 90C59

✉ Jaka Vodeb
jaka.vodeb@ijs.si

Timotej Hrga
timotej.hrga@fmf.uni-lj.si

Janez Povh
janez.povh@rudolfovo.eu

1 Department of Complex Matter, Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia

2 Department of Physics, Faculty for Mathematics and Physics, University of Ljubljana, Jadranska 19, 1000 Ljubljana, Slovenia

3 Department of Mathematics, Faculty for Mathematics and Physics, University of Ljubljana, Jadranska 19, 1000 Ljubljana, Slovenia

4 Rudolfovo - Science and Technology Centre Novo Mesto, Podbreznik 15, 8000 Novo Mesto, Slovenia

# 1 Introduction

## 1.1 Motivation

We are well into the second quantum revolution, where the knowledge gained from the advent of quantum theory is now being used to build quantum devices. One of the biggest potential benefits from such an undertaking is a new and better way of computing, which has been termed “quantum computing". There are two main avenues being developed currently: (i) digital quantum computing, which provides universality in terms of which algorithms can be deployed on such devices, and (ii) quantum annealing (QA), which has only one specific type of algorithm available, but is therefore much more developed in terms of device size and performance. If we only take into account the number of quantum bits or qubits available on the two types of devices, which is by no means a complete performance metric, the biggest device using digital quantum computing is IBM’s Heron with 133 qubits, while the D-Wave Advantage2 hosts over 4000 qubits.

The main driver behind building quantum computers is achieving quantum supremacy, where a quantum computer employing a quantum algorithm significantly outperforms any classical algorithms. This was done by Google in 2019 [1] for a random circuit sampling problem, which holds no practical value. The more realistic goal is a quantum advantage, where a quantum computer outperforms all known classical methods in solving a given problem.

According to a recent review [2], quantum annealers have so far never been able to demonstrate a quantum advantage in practical use cases, which provides motivation for this work. Here, we perform an extensive benchmark test for the Max-Cut and QUBO problems using three standard libraries with problem instances. The result is an assessment of the performance of the Hybrid solver provided by the company D-Wave, as well as their recent more coherent version of quantum processing unit (QPU) involving the fast annealing feature, and their comparison with the performance of (i) the state-of-the-art simulated annealing algorithm for the Max-Cut from [3], and (ii) recent simulated bifurcation machine from [4]. All these solvers are only approximate, meaning they provide the best solutions they have found without guaranteeing a global optimum. Therefore, we also evaluate their results by comparing them with the best-known values: for problem sizes below 500, these are global optima calculated using exact solvers BiqBin [5] and MADAM [6], while for larger problems, we use the best-known values available in the literature.

## 1.2 State of the art

The computational advantage of quantum annealing lies in the ability to traverse the problem’s energy landscape via quantum tunneling compared to various classical update schemes used by combinatorial optimization solvers. The landmark study that unveiled the presence of tunneling effects in real-world quantum annealers [7] was accompanied by an impressive study [8] by Google, where quantum annealing surpassed simulated annealing by a factor of $\sim 10^8$, demonstrating quantum advantage.

However, the advantage is limited to specific problem instances, which were designed specifically to boost the performance of quantum annealers in comparison with classical solvers [8–14]. This is not the case in real-world industrial applications, which include problem instances that are even harder to solve than the notorious spin-glass problem [15] with the requirement for embedding an arbitrary problem graph not native to the annealer's quantum processing unit, representing an additional overhead in computational performance [16]. In principle, quantum annealing provides a theoretical guarantee of finding the ground state due to the quantum adiabatic theorem. However, real devices deviate from this idealized case due to random fields exerted on qubits and random fluctuations of qubit couplers [17], as well as coupling to an external environment at non-zero temperature [18]. The effects of both prevent the system from finding the problem solution and even with a perfect device, the annealing time required to find a solution to a generic problem typically scales exponentially with the system size.

Even with a perfect device, the promise of a performance boost from QA comes from the fact that the timescale is set by the quantum instead of classical dynamics. If the smallest energy gap between the ground and first excited state closes polynomially with system size, the time to solution for QA also scales polynomially with system size, which is an exponential advantage over SA. It has been shown that QA outperforms SA [19]. However, QA also typically exhibits an exponential closing of the gap, which then provides only an advantage in the prefactor, which can still be significant [20]. It is currently still a matter of debate whether QA exhibits any advantage in relevant industry problems [2].

Recently, a breakthrough was made by the company D-Wave through faster control of qubits on the quantum processing unit, enabling access to a time scale where qubits are isolated from the environment, thereby avoiding environmental effects [21–23]. This novel feature remains untested in the context of solving optimization problems.

Recent advancements in quantum annealing hardware have sparked a surge of research and experimentation, driving the quest for enhanced optimization capabilities. Among the forefront contenders in this domain, D-Wave Systems' Advantage quantum annealer stands out, demonstrating notable improvements over its predecessor, the D-Wave 2000Q [12, 24–27]. Rigorous analyses have unveiled Advantage's prowess in solving a diverse array of optimization problems, such as the stable set problem, constraint satisfaction problems, exact cover problems, maximum cardinality matching problems, random clique problems, 3D lattice problems, and the garden optimization problem, showcasing superior performance metrics such as increased success rates and the ability to tackle larger problem sizes. In particular, Advantage has been shown to triumph over D-Wave 2000Q in scenarios involving complex problems with up to 120 logical qubits, a feat previously beyond the reach of its predecessor [27].

In addition to the D-Wave platforms, a plethora of alternative optimization hardware has undergone rigorous scrutiny. Fujitsu's digital annealer unit (DAU), virtual MemComputing machines, Toshiba's simulated bifurcation machine (SBM) [4], and various software-based solvers have all been subjected to comprehensive scaling analyses and prefactor evaluations. These investigations show that quantum annealing exhibits the worst time-to-solution scaling in comparison to other platforms [28]. In particular, quantum annealing hardware is not well-suited for finding degenerate ground-state configurations associated with a particular instance [29, 30]. A comparison with measurement-feedback coherent Ising machines (CIMs) based on optical parametric oscillators on two problem classes, the Sherrington-Kirkpatrick (SK) model and Max-Cut showed that the D-Wave quantum annealer outperforms CIMs on Max-Cut on cubic graphs [31]. On denser problems, however, there is a several order of magnitude time-to-solution difference for instances with over 50 vertices [31]. Therefore, CIMs are not a clear replacement for quantum annealers, and they also suffer from underperformance in comparison to classical combinatorial optimization solvers [32].

The intrinsic heuristic nature of quantum annealers prevents knowing whether the best solution produced by the device is also the global minimum, which can be used to assess the accuracy of the annealer or, in other words, how close the annealer comes to actually solving the proposed problem. Exact solvers have already been used to construct exhaustive libraries of problems and the best-found solutions. These instances are readily available for benchmarking quantum annealers and have not been used so far to the best of our knowledge. A different approach would be to construct a problem specifically tailored to the quantum chip topology where the solution is known by design [33] or using a different method in the branch and bound algorithm, such as tensor networks [34], but this is not the focus of our work.

In general, better quality qubits, higher connectivity, and more tunability is needed to answer the question if quantum annealing will ever truly outperform specialized silicon technology combined with efficient heuristics for optimization and sampling applications [35], thereby making the recent hardware update enabling faster annealing times an important topic of benchmarking. On the other hand, developing software methods that make the best use of the quantum technology available today [36] is crucial in pushing the boundaries of quantum annealing hardware capabilities, given the great potential it holds in the field of machine learning [37, 38].

### 1.3 Our contribution

The studies referenced above, which examine the performance of quantum annealers on combinatorial optimization problems, focus primarily on questions concerning the largest instance sizes that a given solver can address, the computation time required, and the quality of solutions compared to those obtained with other solvers. Notably, most combinatorial optimization problems explored in the literature are NP-hard, indicating that if we want to solve any instance of these problems, i.e., find a global optimum, we will face substantial theoretical and practical complexity. If near-optimal solutions are sufficient, practical heuristic algorithms can often provide them efficiently.

This naturally raises the question of how close the solutions produced by various solvers are to the global optimum. Therefore, the main objectives of this paper are: (i) to assess the proximity of solutions obtained for the Max-Cut and QUBO problems by the D-Wave fast annealing quantum processing unit (QPU) and the D-Wave Hybrid solver to the global optimum, and (ii) to benchmark the performance of these solvers against the state-of-the-art simulated annealing algorithm for the Max-Cut from [3] and the recently introduced Simulated Bifurcation Machine (SBM) from [4].

The main contributions of this paper are:

- We prepared three benchmark datasets for the Max-Cut and QUBO problems, encompassing 139 instances with sizes ranging from 100 to 10,000. For the instances with sizes below 251 (this includes the first dataset and half of the second dataset), global optima are provided by using exact solvers BiqBin [5] and MADAM [6], while for larger instances, we include the best-known solutions from the literature. These datasets are publicly available as open data in our GitHub repository [39] and are formatted for seamless use in further benchmarking studies.
- We (approximately) solved the instances from the benchmark datasets by the D-Wave fast annealing quantum processing unit (QPU) and D-Wave hybrid solver (Hybrid).
- We (approximately) solved the same instances by the state-of-the-art simulated annealing solver for Max-Cut from [3]. We used two variants, SA1 and SA2, of this algorithm that utilize different annealing schedules, i.e., at which initial temperature we start and how big step sizes we make till cooldown. The variant SA1 is more aggressive with the step size, while SA2 spends more time in the search space.
- We conduct an in-depth comparison of solvers in terms of the proximity of their solutions to the global optima and best-known solutions. On the first dataset, which contains small instances, we compared QPU, Hybrid, SA1, and SA2 solvers in terms of proximity to optimum solutions. On the second data set, we compared only the Hybrid solver against the simulated annealing solvers, since the instances exceeded the capacity of the QPU. We compared the proximity of computed solutions to optimum or best-known solutions. On the third dataset, which contains the largest instances, we compared the Hybrid solver against the Toshiba Simulated Bifurcation Machine and the simulated annealing solvers SA1 and SA2. We considered the quality of solutions and the wall-clock times required for solution computation.

Our analysis shows that on the first two datasets, the Hybrid solver and both variants of Simulated Annealing solvers compute global optima whenever they are known or compute solutions with objective values equal to the best-known values from the literature, while QPUs solutions are usually quite far from the optima or best-known solutions.

On the third dataset, for which the global optimum is not known, we observe that the SBM and simulated annealing solver SA2 are very competitive in terms of the quality of solutions, while the Hybrid solver and the SA1 solver compute solutions that are further away from the optimum or the best-known solutions. The corresponding wall-clock times, which are actually hard to compare due to different underlying hardware, indicate that Hybrid, SBM are also fast compared to others, while for simulated annealing, we must find a balance between the quality and the computing time since more computing time gives much better solutions, as is the case with SA2.

## 2 Max-cut and QUBO problems

In this paper, we will consider only undirected weighted graphs without loops and multi-edges, i.e., simple weighted graphs. Each graph $G$ is defined by a set of vertices $V = \{v_1, v_2, \ldots, v_n\}$, set of edges $E = \{\{v_i, v_j\} \mid v_i \neq v_j \text{ and } v_i \text{ is connected with } v_j\}$ and the matrix of edge weights $A$, where $a_{ij}$ is the weight on the edge between $v_i$ and $v_j$, if $v_i$ and $v_j$ are connected, otherwise we have $a_{ij} = 0$. If the graph is unweighted, we simply have $a_{ij} \in \{0, 1\}$. We usually represent a simple (weighted or unweighted) graph $G$ by the matrix $A$, which is called the adjacency matrix of $G$.

Given an edge-weighted undirected graph, the Max-Cut problem asks to find the size of the largest cut in a given graph. A cut is a set of edges that separates the vertices $V$ into two disjoint sets $V_1$ and $V_2$, such that $V_1 \subseteq V$ and $V_2 = V \backslash V_1$, and the cost of a cut is defined as the sum of all weights $a_{ij}$ of edges connecting vertices in $V_1$ with vertices in $V_2$. The Max-Cut problem is a fundamental and well-known example of NP-hard combinatorial optimization problems [40], which are notoriously hard to solve. It has a surprisingly large number of important practical applications, for example, in VLSI design [41] and in the theory of spin glasses in physics [42, 43].

The cost of a cut can be computed by assigning the binary values to express which subset the vertex $v_i$ belongs to $x_i \in \{-1, 1\}$, i.e., every partition of $V$ can be represented by a vector $\mathbf{x} \in \{-1, 1\}^n$ where $x_i = -1$ if and only if the $i$-th vertex $v_i$ belongs to $V_1$. Let $\mathrm{cut}(V_1, V_2)$ denote the cost of a cut. With this notation, we have

$$\mathrm{cut}(V_1, V_2) = \sum_{i \in V_1, j \in V_2} a_{ij} = \sum_{i<j} a_{ij} \frac{1 - x_i x_j}{2} = \frac{1}{4}\mathbf{x}^T L \mathbf{x},$$

where $L$ is the Laplacian matrix of graph $G$, defined by $L = \mathrm{Diag}(A\mathbf{e}) - A$. We denoted by $\mathbf{e}$ the vector of all ones and by $\mathrm{Diag}(Ae)$ the diagonal matrix, having on the main diagonal the vector $A\mathbf{e}$.

Therefore, the Max-Cut problem can be formulated as a binary quadratic program:

$$\begin{aligned} \max \quad & \frac{1}{4}\mathbf{x}^T L \mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in \{\pm 1\}^n. \end{aligned} \tag{MC}$$

Many heuristics have been proposed for Max-Cut. See Dunning et al. [44] for a recent overview. There have also been various articles on exact solution approaches, see e.g. Jünger et al. [45] for an up-to-date overview.

Goemans and Williamson [46] devised a polynomial-time randomized rounding algorithm based on a semidefinite relaxation that computes a solution whose objective value is within a factor $\alpha \approx 0.87856$ of the optimum whenever the weights of the underlying graph are nonnegative. Khot et al. [47] proved that, if the Unique Games Conjecture is true, then there is no polynomial-time algorithm that always computes a solution whose objective value is within a constant factor of optimum, where this factor is larger than $\alpha$.

The Max-Cut problem can be considered a special instance of the classical Ising problem of ferromagnetism in statistical mechanics, where one tries to minimize the classical Hamiltonian of system of spins:

$$H = \frac{1}{2}\sum_{i,j} J_{ij}\sigma_i\sigma_j + \sum_i h_i\sigma_i \tag{Hamiltonian}$$

where $\sigma_i = \pm 1$ represent the values of the Ising spins, $J_{ij}$ are the entries of the spin-spin coupling matrix, and the values $h_i$ represent the influence of the external magnetic field on the spins. The minimization is performed over all possible spin configurations $\sigma$. It is easy to prove that the Ising problem and the Max-Cut problem are actually equivalent. The Max-Cut problem is an instance of the Ising problem without an external magnetic field ($h_i = 0, \forall i$), and the Ising problem becomes the Max-Cut problem if we augment the matrix $J$ by adding the vector $\frac{1}{2}h$ to the first row and first column and increase the dimension of the problem by one.

We also introduce another quadratic optimization problem, closely related to the Max-Cut problem: The Quadratic Unconstrained Binary Optimization (QUBO) problem. It can be formulated as follows: for a given symmetric $n \times n$ matrix $Q$ we want to solve

$$\begin{aligned} \min \quad & \mathbf{x}^T Q\mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in \{0, 1\}^n. \end{aligned} \tag{QUBO}$$

It is not difficult to show that solving (QUBO) is equivalent to solving (MC). QUBO and Max-Cut are equivalent problems, as there exists a linear transformation between them. In particular, any QUBO problem on $n$ variables can be transformed into an equivalent Max-Cut instance on $n+1$, and any Max-Cut instance on a graph $(V, E)$ can be formulated as a QUBO instance with $n = |V| - 1$, see e.g. [48]. For completeness, we show in detail how to do the transformation. Define

$$A = -\begin{bmatrix} 0 & (Qe)^T \\ Qe & Q \end{bmatrix}$$

and consider $A$ to be the adjacency matrix of a graph with vertex set $V = \{0, 1, \ldots, n\}$. Then the Laplacian is given by

$$L = \mathrm{Diag}(Ae) - A = \begin{bmatrix} 0 & (Qe)^T \\ Qe & Q \end{bmatrix} - \begin{bmatrix} e^T Qe & 0 \\ 0 & \mathrm{Diag}(2Qe) \end{bmatrix}.$$

Let $x \in \{-1, 1\}^{n+1}$ denote the adjacency vector of a cut in this graph with value $\frac{1}{4}x^T Lx$. Without loss of generality, we can assume that $x_0 = 1$. By applying the linear transformation and defining $y_i = \frac{1}{2}(x_i + 1)$ for $i = 1, \ldots, n$, we define a vector $y$ in $\{0, 1\}^n$ that is feasible for QUBO problem. The objective value for (MC) and (QUBO) are then given as follows:

$$x^T L x = \begin{bmatrix} 1 \ 2y^T - e^T \end{bmatrix} \left( \begin{bmatrix} 0 & (Qe)^T \\ Qe & Q \end{bmatrix} - \begin{bmatrix} e^T Qe & 0 \\ 0 & \mathrm{Diag}(2Qe) \end{bmatrix} \right) \begin{bmatrix} 1 \\ 2y - e \end{bmatrix}$$
$$= 4y^T Qy - 4e^T Qe.$$

Therefore,

$$\frac{1}{4} x^T L x = y^T Q y - e^T Q e,$$

where the second term is constant, hence minimizing $y^T Qy$ over all $0/1$ vectors $y$ of length $n$ (i.e., solving (QUBO)) is equivalent to minimizing $\frac{1}{4} x^T L x$ over all $\pm 1$ vectors $x$ of length $n + 1$ with $x_0 = 1$, which is clearly equivalent to maximizing $\frac{1}{4} x^T (-L) x$ over all $\pm 1$ vectors $x$ of length $n + 1$, i.e., solving an instance of (MC). For the latter conclusion, we used the fact that if $x$ is an optimal solution for (MC), then $-x$ is also optimal, so we can leave out the constraint $x_0 = 1$. As a minimal illustration, consider the two-variable QUBO

$$\min_{y \in \{0,1\}^2} y^T Q y, \qquad Q = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}.$$

The objective is

$$y^T Q y = y_1 + 3y_2 + 4y_1 y_2.$$

Using the transformation $s_i = 2y_i - 1$, this can be written, up to an additive constant, as an Ising energy

$$y^T Q y = 3 + \frac{3}{2} s_1 + \frac{5}{2} s_2 + s_1 s_2.$$

Introducing an auxiliary spin $s_0 = 1$, the linear terms can be represented as couplings to the auxiliary vertex:

$$H(s_0, s_1, s_2) = \frac{3}{2} s_0 s_1 + \frac{5}{2} s_0 s_2 + s_1 s_2.$$

Equivalently, this defines a weighted three-node Max-Cut instance with edge weights

$$w_{01} = 3, \qquad w_{02} = 5, \qquad w_{12} = 2.$$

The corresponding cut value is

$$C(s) = 3 \frac{1 - s_0 s_1}{2} + 5 \frac{1 - s_0 s_2}{2} + 2 \frac{1 - s_1 s_2}{2} = 8 - y^T Q y.$$

Thus maximizing the cut value is equivalent to minimizing the original QUBO objective, up to the additive constant. The reason for the prominence of QUBO formulation in quantum computing is that many current quantum hardware platforms have a natural connection to QUBO. In Sect. 4.1 we describe three datasets that are used for benchmarking different solvers. Two of them have QUBO origins, i.e. they were originally formulated as QUBO and later reformulated as Max-Cut. In this paper, we always use the Max-Cut formulations.

## 3 Description of used solvers for the Max-Cut problem

The instances of the Max-Cut problem that we consider in this paper are well-known benchmark instances, extensively studied in the literature. For many of them, the optimum solution is known, mostly computed by some variant of complete enumeration algorithms, like the Branch-and-bound or Branch-and-cut algorithms. For all instances with $n \leq 251$ we computed the optimum value by our solvers BiqBin [5] and MADAM [6]. We refer in the tables below only to BiqBin solver. For instances with $250 < n \leq 500$, our solvers gave us only good solutions (often equal to the best-known solutions), without a certificate that these are optimum solutions.

Besides the exact solvers, we also applied to the benchmark instances four approximate solvers. The D-Wave fast annealing quantum processing unit solver (QPU) and the out-of-the-box Hybrid solver (Hybrid) are two approximate solvers provided by D-Wave Systems, which are accessible as cloud solutions. The QPU solves a problem instance in an analogous way to simulated annealing, where temperature fluctuations are replaced with quantum fluctuations. The classical Hamiltonian (Hamiltonian), representing the Max-Cut, is encoded into the extended Hamiltonian

$$H = -\frac{1}{2}A(s)\sum_{i}\sigma_i^x + \frac{1}{2}B(s)\left(\sum_{i<j}J_{i,j}\sigma_i^z\sigma_j^z + \sum_{i}h_i\sigma_i^z\right), \tag{D-Wave Hamiltonian}$$

where $\sigma_i^z$ and $\sigma_i^x$ are Pauli operators, which describe the dynamics of the qubit representing $x_i$. The parameter $s \in [0, 1]$ is used to drive the values of scalar functions $A(s)$ and $B(s)$ within a time frame $t_a$. After the Hamiltonian is encoded on the quantum annealer, $s$ is set to 0, where $A(0)$ is non-zero and $B(0) = 0$. Then within time $t_a$ the value of $s$ goes to 1, where $A(1) = 0$ and $B(1)$ is non-zero. This procedure is analogous to the simulated annealing (SA) method, where the temperature is first set to infinity (or a high enough value above the phase transition in practice) and is then reduced to 0.

According to the publicly available description of the Hybrid[1], it is comprised of an iterated loop consisting of several classical optimization methods as well as the QPU. The classical optimization methods include an interruptable simulated annealing sampler, tabu sampler and a parallel tempering sampler. These methods are run simultaneously in an unspecified way within the Python method `LeapHybridSampler()`, which we employed for this work. However, the providers claim that this is a method optimized for a wide variety of optimization problems, not for any specific applications, and encourage users to develop their own workflows in order to optimize the

[1] https://cloud.dwavesys.com/leap/example-details/285687969/

method for their own specific purposes. We stuck to the out-of-the-box method due to the variety of our input problem instances.

The sub-method of the Hybrid, which employs the QPU, is comprised of an energy decomposer that reduces the input graph to smaller graphs, solves them, and then puts them back together. The public does not have access to the details of the exact methods used for the decomposition and reconstruction of the input graph. When the QPU returns a solution to the whole input graph, the whole iteration loop terminates. With the QPU we could solve only instances with $n \leq 151$, while Hybrid could solve all instances.

Additionally, we also used classical simulated annealing (SA) which was independently developed by Kirkpatrick et al. [49] and Černý [50]. Simulated annealing works by taking a random walk in the space of feasible solutions. It considers a random local modification of the current solution. If that modification leads to a better solution, it is always accepted. If not, its new modification is made with some probability that decreases exponentially with the amount of objective degradation. By accepting points that increase the objective value, the algorithm avoids being trapped in local minima, and is able to explore globally far more possible solutions.

The implementation of simulated annealing for solving Max-Cut problems that we used for computations is from [3]. The reason for this is that in their version, each iteration is computationally efficient, and this allows the code to explore a substantially large number of feasible solutions. Furthermore, the parameters that determine the initial temperature and decrement heavily influence the search time and have an impact on the runtime of the algorithm. We used two variants that employ different annealing schedules. For the SA1 version, the initial temperature was set to 10,000 with the decrement step of 2e$-$4 in order to get the runtimes that are comparable with D-Wave Hybrid. In [3], the author reports that by changing the initial temperature to 40,000 and the decrement to heat to 5e$-$6, the SA finds a feasible solution to instance G35 with objective value 7685, which is better than those found by any other heuristic. Motivated by this and to get better feasible solutions, we set the starting value of the temperature to 40,000 with the decrement step of 2e$-$6 for version SA2.

## 4 Numerical results

In this section, we introduce three benchmark datasets and present numerical results obtained from all solvers capable of providing solutions. For instances for $n \leq 151$, we employed the exact solvers BiqBin and MADAM, the fast annealing quantum solver QPU with annealing time 10 $\mu s$, the hybrid solver, and two simulated annealing solvers, SA1 and SA2. For instances with $n = 251$, the QPU solver exceeded its capacity, and for instances with $n > 251$, even the exact solvers could no longer handle the computations.

### 4.1 Datasets

We performed computations on three Max-Cut datasets, where two of them were originally formulated as QUBO:

- The instances of Billionnet and Elloumi from [51, 52] with $n = 100, 120, 150$, denoted by `be100.1,…,be100.10, be120.3.1,…,be120.3.10, be120.8.1,…,be120.8.10, be150.3.1,…,be150.3.10`, and `be150.8.1,…,be150.8.10`. These instances were originally formulated as QUBO, but we used the Max-Cut reformulations. The source files for our computations were `be100.1.sparse.mc,…,be150.8.10.sparse.mc`, computed from the data files `be100.1.sparse,…,be150.8.10.sparse` with a code `qp2mc`, written by prof. Angelika Wiegele. The files `*.sparse` were taken from the BiqMac site[2]. The new files `*.sparse.mc` are available on our GitHub repository [39].
We created matrices $\frac{1}{4}L$ from these files and named them as `be100.1_L.txt, …, be150.8.10_L.txt`. From these matrices, we extracted Python dictionaries needed as input files for the D-Wave solvers. From each matrix $\frac{1}{4}L$, we extracted the lower triangular matrix (including the diagonal) and multiplied it with $(-1)$ since the D-Wave solvers solve minimization problems. The resulting matrices are available at the GitHub repository [39] as Python dictionaries with names `be100.1_py.txt,…,be150.8.10_py.txt`. Additionally, for each instance we also provided an optimum cut vector `be100.1_opt_cut.txt,…,be150.8.10_opt_cut.txt` and the corresponding optimum values `be100.1_opt_value.txt,…, be150.8.10_opt_value.txt`. Note that the numbers in the names of the instances tell the size of the original QUBO formulation, but after the reformulation into the Max-Cut problem, the size of the instance is increased by 1, so, e.g., the Max-Cut formulation of instance `be100.1` has $n = 101$.
- Instances from the Beasley collection [51, 52] with $n = 250$ and $n = 500$, denoted by `bqp250.1,…,bqp250.10` and `bqp-500.1,…,bqp500.10`. These instances were also originally created as QUBO problems, but we use the Max-Cut reformulation. Like with the `be` instances, the input files were the files `bqp250-1.sparse.mc,…, bqp500-10.sparse.mc`. Again, we created from them the matrices $\frac{1}{4}L$ and the Python dictionaries with lower negative triangular part of $\frac{1}{4}L$ (including diagonals).
As with `be` instances, the dimension of the Max-Cut formulations of `bqp` instances is 1 greater than that of the QUBO formulation. As with the `be` instances, we provide in the GitHub repository [39] for each instance the input files, an optimal cut vector and the optimum value when $n = 251$, and a cut vector giving the best-known cut, including the value of the best-known cut when $n = 501$.
- The G-dataset from Yinyu Ye[3], as used in [4]. The set of Max-Cut instances was generated by Helmberg and Rendl [53] using G. Rinaldi's graph gener-

[2] https://biqmac.aau.at/biqmaclib.html

[3] http://web.stanford.edu/ yyye/yyye/Gset/

ator `rudy`. It consists of 69 matrices labelled as G1, G2,..., and G72 (G68, G69, and G71 are missing for a reason not known to us). Here, our workflow was slightly different. We took as input the edge list format of graph adjacency matrices `G1.txt,...,G72.txt` from Yinyu Ye's web page and created out of them the matrices $L$ (not $\frac{1}{4}L$) and the corresponding Python dictionaries with negative lower triangular parts, without the diagonal. Results were stored with names `G1_L.txt,...,G72_L.txt` and `G1_py.txt,...,G72_py.txt`. In the GitHub repository [39], we provide for each instance the original data file, the $L$ matrix, the Python dictionary, a cut vector yielding the best-known cut value, and the best-known cut value.

### 4.2 Numerical results on benchmark datasets

Our numerical results were obtained using a variety of computing infrastructures. All exact values, computed by BiqBin or MADAM, were computed on the ROME partition of the HPC system at the University of Ljubljana, Faculty of Mechanical Engineering. We used a serial version of the solvers, which ran on single computing nodes, each consisting of 2 AMD EPYC 7402 24-core processor, 128 GB DDR4-3200 RAM, and 1 TB NVMe SSD. The tests with classical simulated annealing were performed on a machine with 2.8 GHz Dual-Core Intel Core i7 with 8GB of RAM. The results obtained using the simulated bifurcation machine solver were performed on an NVIDIA Tesla V100 [4].

On the `be` instances, which are small enough, we executed the exact solver BiqBin, the fast annealing quantum processing unit solver (QPU) with annealing time set to 10 $\mu s$, the out-of-the-box Hybrid solver (Hybrid), and the simulated annealing solvers SA1 and SA2. The results are given in Table 1. The first column of each row gives the name of the underlying graph, the second column contains the size of the problem, and the third column, labelled BiqBin, gives the optimum solutions of the Max-Cut problem obtained by BiqBin or MADAM solvers (both were used for these computations). In the next four columns, we report solutions provided by QPU, Hybrid, SA1, and SA2 solvers. We can see that Hybrid, SA1, and SA2 returned the optimum values for all test instances.

Table 2 contains numerical results for larger instances. For instances with $n = 251$, the optimal values for Max-Cut calculated by BiqBin and the values provided by Hybrid, SA1, and SA2 are given. For $n = 501$, the BiqBin column contains the best values calculated by BiqBin, without a certificate that it has found the optimum. Like in Table 1, for $n = 251$, Hybrid, SA1, and SA2 all find the optimal solutions. For $n = 501$, all four solvers find equivalent solutions, i.e. the cuts they found have the same values.

In Tables 3 and 4 we report the numerical results for the G data set. These instances are too large to use BiqBin and QPU, so we applied only the Hybrid solver and both variants of simulated annealing solvers, SA1 and SA2. Additionally, we report results from Toshiba Simulated Bifurcation Machine solver (SBM), taken from the Supplementary Material of [4]. In addition to the computed value of the Max-Cut problem, we also give the computational wall-clock times in seconds. The reported times correspond to wall-clock times measured in the native execution environments of each solver. Because these solvers run on fundamentally different hardware architectures and software stacks—a local CPU for SA, a single GPU for SBM, and a proprietary cloud-based hybrid quantum-classical service for Hybrid—these times should not be interpreted as intrinsic algorithmic complexities or as direct cross-platform efficiency comparisons. They are included only to document the computational context in which the reported solution values were obtained. Since the optimal solutions for the G dataset are not known, we highlight only the best-computed value among the solvers considered here, which we refer to as the best-known value within this comparison. In terms of solution quality, SBM and SA2 give the strongest results on this dataset. SBM achieves the largest number of best values, while SA2 also reaches the best-known value for a substantial subset of instances. The Hybrid solver and SA1 generally return lower-quality solutions on the G dataset. Thus, the G dataset distinguishes the solvers primarily by solution quality, while the wall-clock times should be read only as deployment-specific contextual information.

A summary of all results is given in Table 5, from which it is clear that datasets `be` and `bqp` were not demanding enough to differentiate between the Hybrid, SA1, and SA2 solvers - all of them produced solutions of the same quality for all these problems. Dataset `G`, on the other hand, was able to distinguish the Hybrid solver, SBM and both SA algorithms according to the quality of solutions. SBM produced the greatest number of best solutions and the greatest number of solutions that were strictly better compared to the other solvers. The solver SA2 was close to SBM, even better in some cases, while the Hybrid and SA1 solvers performed noticeably worse. SA1 exhibited the worst performance among all. It is interesting to note that SA2, which is a traditional simulated annealing algorithm, adapted to Max-Cut, outperforms the very advanced black-box Hybrid solver on the `G` dataset.

## 5 Discussion and conclusions

In this paper, we presented a comprehensive analysis of exact, quantum, and classical approximate solvers for the NP-hard Max-Cut problem, focusing on the solution quality. The main objectives of this study were to evaluate the performance of the D-Wave fast annealing quantum processing unit (QPU) and Hybrid solvers in generating near-optimal solutions and to compare these results both with the best-known solutions from the literature and with solutions obtained using classical heuristic algorithms, in particular Simulated Annealing (algorithms SA1 and SA2) and the Toshiba Simulated Bifurcation Machine (SBM), for which we did not have the code available, but we referred to the results from the literature.

We utilized three benchmark datasets comprising 139 instances ranging from 100 to 10,000 nodes. For smaller instances (up to 251 nodes), we provided the global optima computed by the solvers BiqBin and MADAM. For larger instances, we used the best-known solutions from the literature. We have made these datasets publicly available for further numerical analysis.

On the first dataset, containing instances with a maximum size of 151 nodes, we were able to test all algorithms except SBM. Our results showed that the Hybrid

**Table 1** A comparison between the optimal values obtained with the BiqBin exact solver, and the best objective values obtained with D-Wave's fast annealing QPU and the D-Wave Hybrid solver, and the Simulated Annealing heuristics SA1 and SA2.

| Instance | $n$ | BiqBin | QPU | Hybrid | SA1 | SA2 |
|---|---|---|---|---|---|---|
| be100.1 | 101 | – 19412 | – 19258 | – 19412 | – 19412 | – 19412 |
| be100.2 | 101 | – 17290 | – 17174 | – 17290 | – 17290 | – 17290 |
| be100.3 | 101 | – 17565 | – 17389 | – 17565 | – 17565 | – 17565 |
| be100.4 | 101 | – 19125 | – 18988 | – 19125 | – 19125 | – 19125 |
| be100.5 | 101 | – 15868 | – 15661 | – 15868 | – 15868 | – 15868 |
| be100.6 | 101 | – 17368 | – 17368 | – 17368 | – 17368 | – 17368 |
| be100.7 | 101 | – 18629 | – 18368 | – 18629 | – 18629 | – 18629 |
| be100.8 | 101 | – 18649 | – 18641 | – 18649 | – 18649 | – 18649 |
| be100.9 | 101 | – 13294 | – 12807 | – 13294 | – 13294 | – 13294 |
| be100.10 | 101 | – 15352 | – 15154 | – 15352 | – 15352 | – 15352 |
| be120.3.1 | 121 | – 13067 | – 12342 | – 13067 | – 13067 | – 13067 |
| be120.3.2 | 121 | – 13046 | – 12726 | – 13046 | – 13046 | – 13046 |
| be120.3.3 | 121 | – 12418 | – 12156 | – 12418 | – 12418 | – 12418 |
| be120.3.4 | 121 | – 13867 | – 13381 | – 13867 | – 13867 | – 13867 |
| be120.3.5 | 121 | – 11403 | – 11237 | – 11403 | – 11403 | – 11403 |
| be120.3.6 | 121 | – 12915 | – 12582 | – 12915 | – 12915 | – 12915 |
| be120.3.7 | 121 | – 14068 | – 13946 | – 14068 | – 14068 | – 14068 |
| be120.3.8 | 121 | – 14701 | – 14016 | – 14701 | – 14701 | – 14701 |
| be120.3.9 | 121 | – 10458 | – 10230 | – 10458 | – 10458 | – 10458 |
| be120.3.10 | 121 | – 12201 | – 11375 | – 12201 | – 12201 | – 12201 |
| be120.8.1 | 121 | – 18691 | – 18372 | – 18691 | – 18691 | – 18691 |
| be120.8.2 | 121 | – 18827 | – 18161 | – 18827 | – 18827 | – 18827 |
| be120.8.3 | 121 | – 19302 | – 18561 | – 19302 | – 19302 | – 19302 |
| be120.8.4 | 121 | – 20765 | – 19536 | – 20765 | – 20765 | – 20765 |
| be120.8.5 | 121 | – 20417 | – 20224 | – 20417 | – 20417 | – 20417 |
| be120.8.6 | 121 | – 18482 | – 18132 | – 18482 | – 18482 | – 18482 |
| be120.8.7 | 121 | – 22194 | – 21431 | – 22194 | – 22194 | – 22194 |
| be120.8.8 | 121 | – 19534 | – 17832 | – 19534 | – 19534 | – 19534 |
| be120.8.9 | 121 | – 18195 | – 17511 | – 18195 | – 18195 | – 18195 |

solver and both simulated annealing variants consistently achieved the global optimum, whereas the QPU produced solutions that were far from optimal.

For the second dataset, the exact solver computed the global optimum for instances up to 251 nodes and matched the best-known values from the literature for instances of size 501. On this dataset, we tested the exact solver, the Hybrid solver, and both simulated annealing variants. The instances were too large for the QPU to handle. We observed similar results as on the first dataset, except the QPU could no longer be tested on these larger instances. The Hybrid solver and both simulated annealing

**Table 1** continued

| Instance | $n$ | BiqBin | QPU | Hybrid | SA1 | SA2 |
|---|---|---|---|---|---|---|
| be120.8.10 | 121 | – 19049 | – 18604 | – 19049 | – 19049 | – 19049 |
| be150.3.1 | 151 | – 18889 | – 17804 | – 18889 | – 18889 | – 18889 |
| be150.3.2 | 151 | – 17816 | – 17103 | – 17816 | – 17816 | – 17816 |
| be150.3.3 | 151 | – 17314 | – 16609 | – 17314 | – 17314 | – 17314 |
| be150.3.4 | 151 | – 19884 | – 18231 | – 19884 | – 19884 | – 19884 |
| be150.3.5 | 151 | – 16817 | – 15904 | – 16817 | – 16817 | – 16817 |
| be150.3.6 | 151 | – 16780 | – 16346 | – 16780 | – 16780 | – 16780 |
| be150.3.7 | 151 | – 18001 | – 16156 | – 18001 | – 18001 | – 18001 |
| be150.3.8 | 151 | – 18303 | – 17054 | – 18303 | – 18303 | – 18303 |
| be150.3.9 | 151 | – 12838 | – 11728 | – 12838 | – 12838 | – 12838 |
| be150.3.10 | 151 | – 17963 | – 16117 | – 17963 | – 17963 | – 17963 |
| be150.8.1 | 151 | – 27089 | – 24231 | – 27089 | – 27089 | – 27089 |
| be150.8.2 | 151 | – 26779 | – 25912 | – 26779 | – 26779 | – 26779 |
| be150.8.3 | 151 | – 29438 | – 26448 | – 29438 | – 29438 | – 29438 |
| be150.8.4 | 151 | – 26911 | – 24586 | – 26911 | – 26911 | – 26911 |
| be150.8.5 | 151 | – 28017 | – 26874 | – 28017 | – 28017 | – 28017 |
| be150.8.6 | 151 | – 29221 | – 27481 | – 29221 | – 29221 | – 29221 |
| be150.8.7 | 151 | – 31209 | – 28275 | – 31209 | – 31209 | – 31209 |
| be150.8.8 | 151 | – 29730 | – 27397 | – 29730 | – 29730 | – 29730 |
| be150.8.9 | 151 | – 25388 | – 22949 | – 25388 | – 25388 | – 25388 |
| be150.8.10 | 151 | – 28374 | – 27251 | – 28374 | – 28374 | – 28374 |

variants achieved solutions with objective values that matched global optima where known and best-known values otherwise.

The third dataset, containing instances between 800 and 10,000 nodes, was especially informative because the global optima are not known. For these instances, we compared the Hybrid solver, both simulated annealing variants, and published SBM results from [4]. The comparison therefore concerns solution quality relative to the best-known values, not certified optimality. On this dataset, SBM produced the largest number of best-known values among the solvers considered, while SA2 was also highly competitive. The Hybrid solver was able to process the full range of instance sizes considered here, but its solution quality on the G dataset was generally below that of SBM and SA2.

Our results indicate that the D-Wave fast annealing quantum processing unit (QPU) serves as an intriguing solver, capable of producing multiple good solutions within a short time frame. However, these solutions are far from optimal, and the QPU's limitations in handling larger problem instances are significant; we could only solve instances up to a size of 151. We found that decreasing the annealing time into the coherent regime ($< 20\ ns$) substantially decreases the solution quality, meaning that the coherent regime has to be extended in future quantum hardware to provide optimal solutions with purely quantum annealing. In contrast, the D-Wave Hybrid solver can

**Table 2** A comparison between the best objective values obtained with the BiqBin solver, the D-Wave Hybrid solver, and the Simulated Annealing heuristics SA1 and SA2.

| Instance | $n$ | BiqBin | Hybrid | SA1 | SA2 |
|---|---|---|---|---|---|
| bqp250-1 | 251 | – 45607 | – 45607 | – 45607 | – 45607 |
| bqp250-2 | 251 | – 44810 | – 44810 | – 44810 | – 44810 |
| bqp250-3 | 251 | – 49037 | – 49037 | – 49037 | – 49037 |
| bqp250-4 | 251 | – 41274 | – 41274 | – 41274 | – 41274 |
| bqp250-5 | 251 | – 47961 | – 47961 | – 47961 | – 47961 |
| bqp250-6 | 251 | – 41014 | – 41014 | – 41014 | – 41014 |
| bqp250-7 | 251 | – 46757 | – 46757 | – 46757 | – 46757 |
| bqp250-8 | 251 | – 35726 | – 35726 | – 35726 | – 35726 |
| bqp250-9 | 251 | – 48916 | – 48916 | – 48916 | – 48916 |
| bqp250-10 | 251 | – 40442 | – 40442 | – 40442 | – 40442 |
| bqp500-1 | 501 | – 116586 | – 116586 | – 116586 | – 116586 |
| bqp500-2 | 501 | – 128339 | – 128339 | – 128339 | – 128339 |
| bqp500-3 | 501 | – 130812 | – 130812 | -130812 | -130812 |
| bqp500-4 | 501 | – 130097 | – 130097 | -130097 | -130097 |
| bqp500-5 | 501 | – 125487 | – 125487 | -125487 | -125487 |
| bqp500-6 | 501 | – 121772 | – 121772 | -121772 | -121772 |
| bqp500-7 | 501 | – 122201 | – 122201 | -122201 | -122201 |
| bqp500-8 | 501 | – 123559 | – 123559 | -123559 | -123559 |
| bqp500-9 | 501 | – 120798 | – 120798 | -120798 | -120798 |
| bqp500-10 | 501 | – 130619 | – 130619 | -130619 | -130619 |

handle all instances considered in this work, including the largest G-set instances with up to 10,000 nodes. It matches the certified optimum on the smaller instances for which such certificates are available and matches the best-known values on the bqp instances with $n = 501$. On the larger G-set instances, where optimality is not certified, it provides feasible solutions but is generally outperformed in solution quality by SBM and SA2. For scientific benchmarking, a remaining limitation is that the Hybrid solver operates as a proprietary black-box cloud service, leaving us without detailed insight into its internal workflow or into the extent to which quantum resources are used. The SBM also seems to be a highly competitive solver. Unfortunately, we had no access to this solver and were unable to conduct testing with it and had to rely entirely on publication [4], limiting our ability to fully evaluate this solver.

Notably, classical simulated annealing algorithms, which are transparent and can leverage specific features of the Max-Cut problem, compete strongly with the Hybrid solver in terms of both solution quality and time. The key advantage here is that we fully understand the steps within these classical algorithms.

Our study, therefore, advances the understanding of quantum-assisted optimization for combinatorial challenges and offers valuable insights for selecting appropriate solvers based on application-specific constraints and desired performance.

**Table 3** A comparison between the best solutions obtained with the Simulated Annealing heuristic (SA), Toshiba Simulated Bifurcation Machine solver (SBM), and the D-Wave Hybrid solver.

| Instance | $n$ | Hybrid | $t_{Hybrid}(s)$ | SBM | $t_{\mathrm{SBM}}(s)$ | SA1 | $t_{SA1}(s)$ | SA2 | $t_{SA2}(s)$ |
|---|---|---|---|---|---|---|---|---|---|
| G1 | 800 | **11624** | 3.00 | **11624** | 0.033 | 11615 | 2.66 | **11624** | 672.70 |
| G2 | 800 | **11620** | 3.00 | **11620** | 0.24 | 11604 | 2.64 | **11620** | 672.01 |
| G3 | 800 | **11622** | 2.99 | **11622** | 0.046 | 11610 | 2.82 | **11622** | 675.80 |
| G4 | 800 | **11646** | 3.00 | **11646** | 0.034 | 11626 | 2.75 | **11646** | 671.84 |
| G5 | 800 | **11631** | 3.00 | **11631** | 0.059 | 11618 | 2.64 | **11631** | 674.53 |
| G6 | 800 | **2178** | 3.00 | **2178** | 0.006 | **2178** | 1.64 | **2178** | 556.42 |
| G7 | 800 | **2006** | 3.00 | **2006** | 0.007 | **2006** | 1.62 | **2006** | 569.27 |
| G8 | 800 | **2005** | 3.00 | **2005** | 0.012 | 1999 | 1.62 | **2005** | 572.57 |
| G9 | 800 | 2053 | 3.00 | **2054** | 0.036 | 2046 | 1.57 | **2054** | 567.68 |
| G10 | 800 | **2000** | 3.00 | **2000** | 0.048 | 1998 | 2.16 | 1999 | 572.10 |
| G11 | 800 | 562 | 3.00 | **564** | 0.003 | 558 | 1.49 | **564** | 571.73 |
| G12 | 800 | 554 | 3.00 | **556** | 0.005 | 552 | 1.48 | **556** | 573.89 |
| G13 | 800 | 580 | 2.99 | **582** | 0.012 | 576 | 1.43 | **582** | 571.59 |

**Table 3** continued

| Instance | $n$ | Hybrid | $t_{Hybrid}(s)$ | SBM | $t_{\mathrm{SBM}}(s)$ | SA1 | $t_{SA1}(s)$ | SA2 | $t_{SA2}(s)$ |
|---|---|---|---|---|---|---|---|---|---|
| G14 | 800 | 3058 | 3.00 | **3064** | 72 | 3059 | 1.86 | 3062 | 611.48 |
| G15 | 800 | 3045 | 3.00 | **3050** | 0.34 | 3035 | 1.87 | 3049 | 615.67 |
| G16 | 800 | 3047 | 2.99 | **3052** | 0.35 | **3052** | 1.86 | **3052** | 603.63 |
| G17 | 800 | 3043 | 3.00 | **3047** | 1.6 | 3042 | 1.80 | 3045 | 606.08 |
| G18 | 800 | 988 | 2.98 | **992** | 0.38 | 986 | 1.54 | 988 | 575.92 |
| G19 | 800 | **906** | 2.99 | **906** | 0.018 | **906** | 1.52 | 903 | 579.26 |
| G20 | 800 | **941** | 3.00 | **941** | 0.009 | **941** | 1.55 | **941** | 577.41 |
| G21 | 800 | 928 | 3.01 | **931** | 0.26 | 927 | 1.54 | 927 | 578.40 |
| G22 | 2000 | 13351 | 5.22 | **13359** | 0.43 | 13099 | 2.71 | **13359** | 707.93 |
| G23 | 2000 | 13327 | 5.22 | 13342 | 0.089 | 13077 | 2.70 | **13344** | 708.00 |
| G24 | 2000 | 13330 | 5.22 | **13337** | 0.46 | 13081 | 2.73 | **13337** | 706.96 |
| G25 | 2000 | 13334 | 5.22 | **13340** | 2.3 | 13088 | 2.72 | 13333 | 707.71 |
| G26 | 2000 | 13322 | 5.22 | **13328** | 0.48 | 13077 | 2.73 | 13323 | 707.69 |
| G27 | 2000 | 3333 | 5.22 | **3341** | 0.050 | **3341** | 1.77 | **3341** | 585.01 |
| G28 | 2000 | 3291 | 5.22 | **3298** | 0.087 | 3287 | 1.77 | **3298** | 583.88 |
| G29 | 2000 | 3395 | 5.22 | **3405** | 0.22 | 3391 | 1.82 | **3405** | 572.47 |
| G30 | 2000 | 3409 | 5.21 | **3413** | 0.44 | 3412 | 1.79 | 3409 | 608.00 |
| G31 | 2000 | 3305 | 5.23 | **3310** | 1.2 | 3306 | 1.77 | 3309 | 577.13 |
| G32 | 2000 | 1398 | 5.22 | **1410** | 3.6 | 1400 | 1.57 | 1408 | 569.81 |
| G33 | 2000 | 1372 | 5.22 | **1382** | 58 | 1374 | 1.56 | 1380 | 584.98 |
| G34 | 2000 | 1376 | 5.22 | **1384** | 2.1 | 1376 | 1.58 | 1382 | 595.74 |

Bold represents the best found values across all of the solvers

**Table 4** A comparison between the best solutions obtained with the Simulated Annealing heuristic (SA), Toshiba Simulated Bifurcation Machine solver (SBM), and the D-Wave Hybrid solver.

| Instance | $n$ | Hybrid | $t_{Hybrid}(s)$ | SBM | $t_{\text{SBM}}(s)$ | SA1 | $t_{SA1}(s)$ | SA2 | $t_{SA2}(s)$ |
|---|---|---|---|---|---|---|---|---|---|
| G35 | 2000 | 7659 | 5.22 | **7686** | 8319 | 7472 | 2.41 | **7686** | 683.91 |
| G36 | 2000 | 7655 | 5.21 | **7680** | 62647 | 7458 | 2.51 | 7675 | 668.88 |
| G37 | 2000 | 7665 | 5.22 | **7691** | 27343 | 7464 | 2.41 | 7676 | 663.61 |
| G38 | 2000 | 7661 | 5.22 | **7688** | 99 | 7451 | 2.45 | 7680 | 689.77 |
| G39 | 2000 | 2390 | 5.22 | **2408** | 56 | 2392 | 1.72 | 2400 | 580.16 |
| G40 | 2000 | 2387 | 5.21 | **2400** | 24 | 2386 | 1.70 | 2393 | 620.22 |
| G41 | 2000 | 2386 | 5.22 | **2405** | 11 | 2387 | 1.72 | **2405** | 592.46 |
| G42 | 2000 | 2464 | 5.22 | **2480** | 550 | 2465 | 1.73 | 2467 | 569.23 |
| G43 | 1000 | **6660** | 2.99 | **6660** | 0.006 | 6656 | 2.13 | **6660** | 603.74 |
| G44 | 1000 | **6650** | 3.00 | **6650** | 0.007 | 6637 | 2.14 | **6650** | 601.53 |
| G45 | 1000 | 6653 | 2.99 | **6654** | 0.043 | 6644 | 2.14 | **6654** | 601.96 |
| G46 | 1000 | **6649** | 2.99 | **6649** | 0.016 | 6642 | 2.14 | **6649** | 604.65 |
| G47 | 1000 | 6656 | 2.99 | **6657** | 0.045 | 6647 | 2.13 | 6655 | 604.32 |
| G48 | 3000 | **6000** | 7.50 | **6000** | 0.0008 | **6000** | 1.78 | **6000** | 558.03 |
| G49 | 3000 | **6000** | 7.50 | **6000** | 0.0008 | **6000** | 1.75 | **6000** | 559.09 |

**Table 4** continued

| Instance | $n$ | Hybrid | $t_{Hybrid}(s)$ | SBM | $t_{\mathrm{SBM}}(s)$ | SA1 | $t_{SA1}(s)$ | SA2 | $t_{SA2}(s)$ |
|---|---|---|---|---|---|---|---|---|---|
| G50 | 3000 | **5880** | 7.50 | **5880** | 0.0026 | 5824 | 1.78 | **5880** | 560.66 |
| G51 | 1000 | 3843 | 2.99 | **3848** | 12 | 3836 | 2.01 | **3848** | 597.82 |
| G52 | 1000 | 3845 | 2.99 | **3851** | 6.9 | 3839 | 2.03 | 3849 | 596.27 |
| G53 | 1000 | 3845 | 3.00 | **3850** | 94 | 3827 | 2.00 | 3848 | 598.09 |
| G54 | 1000 | 3845 | 3.00 | **3852** | 2307 | 3827 | 2.00 | 3845 | 617.68 |
| G55 | 5000 | 10264 | 14.59 | **10298** | 6874 | 9367 | 2.44 | 10289 | 680.00 |
| G56 | 5000 | 3982 | 14.58 | **4016** | 6887 | 3912 | 1.98 | 4007 | 592.00 |
| G57 | 5000 | 3456 | 14.59 | **3490** | 6569 | 3436 | 1.84 | 3486 | 582.95 |
| G58 | 5000 | 19221 | 14.59 | **19283** | 21638 | 17392 | 3.23 | 19118 | 858.10 |
| G59 | 5000 | 6025 | 14.60 | **6086** | 22651 | 5892 | 2.28 | 6065 | 624.34 |
| G60 | 7000 | 14142 | 24.76 | **14186** | 21827 | 12120 | 2.72 | 14114 | 733.45 |
| G61 | 7000 | 5750 | 24.75 | **5796** | 10883 | 5445 | 2.19 | 5788 | 616.86 |
| G62 | 7000 | 4818 | 24.74 | **4862** | 20501 | 4632 | 2.06 | 4856 | 603.25 |
| G63 | 7000 | 26948 | 24.75 | **27023** | 31279 | 23751 | 3.47 | 26384 | 967.89 |
| G64 | 7000 | 8665 | 24.74 | **8739** | 31448 | 8158 | 2.51 | 8728 | 661.92 |
| G65 | 8000 | 5494 | 29.83 | **5546** | 5651 | 5164 | 2.06 | 5544 | 613.44 |
| G66 | 9000 | 6282 | 34.92 | 6342 | 27408 | 5770 | 2.13 | **6350** | 621.45 |
| G67 | 10000 | 6868 | 39.98 | 6922 | 6340 | 6136 | 2.18 | **6924** | 630.33 |
| G70 | 10000 | 9516 | 40.00 | **9578** | 31599 | 7909 | 2.48 | 9546 | 709.99 |
| G72 | 10000 | 6914 | 39.98 | 6982 | 6142 | 6148 | 2.20 | **6984** | 641.89 |

Bold represents the best found values across all of the solvers

**Table 5** For each set of instances, we report how many times each algorithm discovered the best and worst solution. The "only" column counts the number of instances where the solver is the only one that achieved the best value, whereas the "worst" column gives the number of instances the solver computed the smallest value among all solvers, and this value was strictly smaller than the best value, although there may be another solver that also computed the same smallest value.

| Set | Set size | Hybrid | | | SBM | | | SA1 | | | SA2 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | best | only | worst | best | only | worst | best | only | worst | best | only | worst |
| `be` | 50 | 50 | 0 | 0 | – | – | – | 50 | 0 | 0 | 50 | 0 | 0 |
| `bqp` | 20 | 20 | 0 | 0 | – | – | – | 20 | 0 | 0 | 20 | 0 | 0 |
| `G` | 69 | 17 | 0 | 11 | 65 | 34 | 0 | 8 | 0 | 53 | 33 | 4 | 3 |

**Acknowledgements** The research of J.P. was co-funded by the Republic of Slovenia, the Ministry of Higher Education, Science and Innovation, and the Slovenian Research and Innovation Agency of the European Union—NextGenerationEU through the DIGITOP project, by the Slovenian Research and Innovation Agency (ARIS) through the annual work program of Rudolfovo and the research project L1-60136. J.V. acknowledges the financial support from ARIS, P08333 young researcher grant, P1-0040 Nonequilibrium Quantum System Dynamics, and P1-0416 Physics of Quantum Technologies.

**Author Contributions** J.V. and J.P. conceptualized the benchmarking study, J.V., J.P., and T.H. co-wrote the manuscript, V.E. performed the calculations on the quantum annealer, T.H. performed the calculations using simulated annealing.

**Data Availability** The benchmark dataset is publicly available at https://doi.org/10.5281/zenodo.14290290.

## Declarations

**Competing interests** The authors declare no competing interests.



## References

1. Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J.C., Barends, R., Biswas, R., Boixo, S., Brandao, F.G., Buell, D.A.: Quantum supremacy using a programmable superconducting processor. Nature **574**(7779), 505–510 (2019)
2. Hauke, P., Katzgraber, H.G., Lechner, W., Nishimori, H., Oliver, W.D.: Perspectives of quantum annealing: Methods and implementations. Rep. Prog. Phys. **83**(5), 054401 (2020)
3. Myklebust, T.G.: Solving maximum cut problems by simulated annealing (2015). arXiv:1505.03068
4. Goto, H., Endo, K., Suzuki, M., Sakai, Y., Kanao, T., Hamakawa, Y., Hidaka, R., Yamasaki, M., Tatsumura, K.: High-performance combinatorial optimization based on classical mechanics. Sci. Adv. **7**(6), 7953 (2021)
5. Gusmeroli, N., Hrga, T., Lužar, B., Povh, J., Siebenhofer, M., Wiegele, A.: Biqbin: a parallel branch-and-bound solver for binary quadratic problems with linear constraints. ACM Trans. Math. Softw. (TOMS) **48**(2), 1–31 (2022)
6. Hrga, T., Povh, J.: Madam: a parallel exact solver for max-cut based on semidefinite programming and admm. Comput. Optim. Appl. **80**(2), 347–375 (2021)
7. Boixo, S., Smelyanskiy, V.N., Shabani, A., Isakov, S.V., Dykman, M., Denchev, V.S., Amin, M.H., Smirnov, A.Y., Mohseni, M., Neven, H.: Computational multiqubit tunnelling in programmable quantum annealers. Nat. Commun. **7**(1), 10327 (2016)
8. Denchev, V.S., Boixo, S., Isakov, S.V., Ding, N., Babbush, R., Smelyanskiy, V., Martinis, J., Neven, H.: What is the computational value of finite-range tunneling? Phys. Rev. X **6**(3), 031015 (2016)
9. Pang, Y., Coffrin, C., Lokhov, A.Y., Vuffray, M.: The potential of quantum annealing for rapid solution structure identification. Constraints **26**(1), 1–25 (2021)
10. Tasseff, B., Albash, T., Morrell, Z., Vuffray, M., Lokhov, A.Y., Misra, S., Coffrin, C.: On the emerging potential of quantum annealing hardware for combinatorial optimization (2022). arXiv:2210.04291
11. Cao, Y., Jiang, S., Perouli, D., Kais, S.: Solving set cover with pairs problem using quantum annealing. Sci. Rep. **6**(1), 33957 (2016)

12. King, J., Yarkoni, S., Raymond, J., Ozfidan, I., King, A.D., Nevisi, M.M., Hilton, J.P., McGeoch, C.C.: Quantum annealing amid local ruggedness and global frustration. J. Phys. Soc. Jpn. **88**(6), 061007 (2019)
13. Mandra, S., Zhu, Z., Wang, W., Perdomo-Ortiz, A., Katzgraber, H.G.: Strengths and weaknesses of weak-strong cluster problems: a detailed overview of state-of-the-art classical heuristics versus quantum approaches. Phys. Rev. A **94**(2), 022337 (2016)
14. Koshikawa, A.S., Ohzeki, M., Kadowaki, T., Tanaka, K.: Benchmark test of black-box optimization using d-wave quantum annealer. J. Phys. Soc. Jpn. **90**(6), 064001 (2021)
15. Perdomo-Ortiz, A., Feldman, A., Ozaeta, A., Isakov, S.V., Zhu, Z., O'Gorman, B., Katzgraber, H.G., Diedrich, A., Neven, H., Kleer, J.: Readiness of quantum optimization machines for industrial applications. Phys. Rev. Appl. **12**(1), 014004 (2019)
16. Könz, M.S., Lechner, W., Katzgraber, H.G., Troyer, M.: Embedding overhead scaling of optimization problems in quantum annealing. PRX Quantum **2**(4), 040322 (2021)
17. Zhu, Z., Ochoa, A.J., Schnabel, S., Hamze, F., Katzgraber, H.G.: Best-case performance of quantum annealers on native spin-glass benchmarks: how chaos can affect success probabilities. Phys. Rev. A **93**(1), 012317 (2016)
18. Albash, T., Martin-Mayor, V., Hen, I.: Temperature scaling law for quantum annealing optimizers. Phys. Rev. Lett. **119**(11), 110502 (2017)
19. Albash, T., Lidar, D.A.: Demonstration of a scaling advantage for a quantum annealer over simulated annealing. Phys. Rev. X **8**(3), 031016 (2018)
20. King, A.D., Raymond, J., Lanting, T., Isakov, S.V., Mohseni, M., Poulin-Lamarre, G., Ejtemaee, S., Bernoudy, W., Ozfidan, I., Smirnov, A.Y.: Scaling advantage over path-integral Monte Carlo in quantum simulation of geometrically frustrated magnets. Nat. Commun. **12**(1), 1113 (2021)
21. King, A.D., Suzuki, S., Raymond, J., Zucca, A., Lanting, T., Altomare, F., Berkley, A.J., Ejtemaee, S., Hoskinson, E., Huang, S.: Coherent quantum annealing in a programmable 2,000 qubit ising chain. Nat. Phys. **18**(11), 1324–1328 (2022)
22. King, A.D., Raymond, J., Lanting, T., Harris, R., Zucca, A., Altomare, F., Berkley, A.J., Boothby, K., Ejtemaee, S., Enderud, C.: Quantum critical dynamics in a 5,000-qubit programmable spin glass. Nature **617**(7959), 61–66 (2023)
23. King, A.D., Nocera, A., Rams, M.M., Dziarmaga, J., Wiersema, R., Bernoudy, W., Raymond, J., Kaushal, N., Heinsdorf, N., Harris, R.: Computational supremacy in quantum simulation (2024). arXiv:2403.00910
24. Gonzalez Calaza, C.D., Willsch, D., Michielsen, K.: Garden optimization problems for benchmarking quantum annealers. Quantum Inf. Process. **20**(9), 305 (2021)
25. Inc, D.-W.S.: The Advantage System: Performance Update. https://www.dwavesys.com/media/kjtlcemb/14-1054a-a_advantage_system_performance_update.pdf Accessed 2024-04-22
26. McLeod, C.R., Sasdelli, M.: Benchmarking d-wave quantum annealers: spectral gap scaling of maximum cardinality matching problems. In: International Conference on Computational Science, pp. 150–163. Springer (2022)
27. Willsch, D., Willsch, M., Gonzalez Calaza, C.D., Jin, F., De Raedt, H., Svensson, M., Michielsen, K.: Benchmarking advantage and d-wave 2000q quantum annealers with exact cover problems. Quantum Inf. Process. **21**(4), 141 (2022)
28. Kowalsky, M., Albash, T., Hen, I., Lidar, D.A.: 3-regular three-xorsat planted solutions benchmark of classical and quantum heuristic optimizers. Quantum Sci. Technol. **7**(2), 025008 (2022)
29. Mandra, S., Zhu, Z., Katzgraber, H.G.: Exponentially biased ground-state sampling of quantum annealing machines with transverse-field driving hamiltonians. Phys. Rev. Lett. **118**(7), 070502 (2017)
30. Zhu, Z., Ochoa, A.J., Katzgraber, H.G.: Fair sampling of ground-state configurations of binary optimization problems. Phys. Rev. E **99**(6), 063314 (2019)
31. Hamerly, R., Inagaki, T., McMahon, P.L., Venturelli, D., Marandi, A., Onodera, T., Ng, E., Langrock, C., Inaba, K., Honjo, T.: Experimental investigation of performance differences between coherent ising machines and a quantum annealer. Sci. Adv. **5**(5), 0823 (2019)
32. Sharma, A., Burns, M., Hahn, A., Huang, M.: Augmenting an electronic ising machine to effectively solve boolean satisfiability. Sci. Rep. **13**(1), 22858 (2023)
33. Pelofske, E., Hahn, G., Djidjev, H.: Increasing the hardness of posiform planting using random qubos for programmable quantum annealer benchmarking (2024). arXiv:2411.03626

34. Dziubyna, A.M., Śmierzchalski, T., Gardas, B., Rams, M.M., Mohseni, M.: Limitations of tensor-network approaches for optimization and sampling: a comparison to quantum and classical ising machines. Phys. Rev. Appl. **23**(5), 054049 (2025)
35. Katzgraber, H.G.: Viewing vanilla quantum annealing through spin glasses. Quantum Sci. Technol. **3**(3), 030505 (2018)
36. Montanez-Barrera, J., Heuvel, P., Willsch, D., Michielsen, K.: Improving performance in combinatorial optimization problems with inequality constraints: an evaluation of the unbalanced penalization method on d-wave advantage. In: 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), vol. 1, pp. 535–542. IEEE (2023)
37. Radovic, A., Williams, M., Rousseau, D., Kagan, M., Bonacorsi, D., Himmel, A., Aurisano, A., Terao, K., Wongjirad, T.: Machine learning at the energy and intensity frontiers of particle physics. Nature **560**(7716), 41–48 (2018)
38. Zlokapa, A., Mott, A., Job, J., Vlimant, J.-R., Lidar, D., Spiropulu, M.: Quantum adiabatic machine learning by zooming into a region of the energy surface. Phys. Rev. A **102**(6), 062405 (2020)
39. Eržen, V., Hrga, T., Povh, J., Vodeb, J.: Max-Cut Benchmark Dataset. https://doi.org/10.5281/zenodo.14290290
40. Karp, R.M.: Reducibility Among Combinatorial Problems. Springer, Berlin (2010)
41. Barahona, F., Grötschel, M., Jünger, M., Reinelt, G.: An application of combinatorial optimization to statistical physics and circuit layout design. Oper. Res. **36**(3), 493–513 (1988)
42. Grötschel, M., Jünger, M., Reinelt, G.: Calculating exact ground states of spin glasses: A polyhedral approach. In: Heidelberg Colloquium on Glassy Dynamics: Proceedings of a Colloquium on Spin Glasses, Optimization and Neural Networks Held at the University of Heidelberg June 9–13, 1986, pp. 325–353 (1987). Springer
43. Liers, F.: Contributions to determining exact ground-states of ising spin-glasses and to their physics, Universität zu Köln (2004). (PhD thesis)
44. Dunning, I., Gupta, S., Silberholz, J.: What works best when? A systematic evaluation of heuristics for max-cut and qubo. INFORMS J. Comput. **30**(3), 608–624 (2018)
45. Jünger, M., Lobe, E., Mutzel, P., Reinelt, G., Rendl, F., Rinaldi, G., Stollenwerk, T.: Quantum annealing versus digital computing: an experimental comparison. J. Exp. Algorithm. (JEA) **26**, 1–30 (2021)
46. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. J. ACM (JACM) **42**(6), 1115–1145 (1995)
47. Khot, S., Kindler, G., Mossel, E., O'Donnell, R.: Optimal inapproximability results for max-cut and other 2-variable csps? SIAM J. Comput. **37**(1), 319–357 (2007)
48. Barahona, F., Jünger, M., Reinelt, G.: Experiments in quadratic 0–1 programming. Math. Program. **44**(1), 127–137 (1989)
49. Kirkpatrick, S., Gelatt, C.D., Jr., Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)
50. Černỳ, V.: Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J. Optim. Theory Appl. **45**, 41–51 (1985)
51. Wiegele, A.: Biq mac library—a collection of max-cut and quadratic 0-1 programming instances of medium size. Preprint (2007)
52. Wiegele, A.: BiqMac Library. http://biqmac.aau.at/biqmaclib.html (2007)
53. Helmberg, C., Rendl, F.: A spectral bundle method for semidefinite programming. SIAM J. Optim. **10**(3), 673–696 (2000)