Benchmarking Quantum Annealers with Near-Optimal Minor-Embedded Instances

Valentin Gilbert
Université Paris-Saclay
CEA, List, F-91120
Palaiseau, France
valentin.gilbert@cea.fr

Julien Rodriguez
Université de Montpellier
LIRMM, CNRS
Montpellier, France
julien.rodriguez@lirmm.fr

Stéphane Louise
Université Paris-Saclay
CEA, List, F-91120
Palaiseau, France
stephane.louise@cea.fr

***Abstract*—Benchmarking Quantum Process Units (QPU) at an application level usually requires considering the whole programming stack of the quantum computer. One critical task is the minor-embedding (resp. transpilation) step, which involves space-time overheads for annealing-based (resp. gate-based) quantum computers. This paper establishes a new protocol to generate graph instances with their associated near-optimal minor-embedding mappings to D-Wave Quantum Annealers (QA). This set of favorable mappings is used to generate a wide diversity of optimization problem instances. We use this method to benchmark QA on large instances of unconstrained and constrained optimization problems and compare the performance of the QPU with efficient classical solvers. The benchmark aims to evaluate and quantify the key characteristics of instances that could benefit from the use of a quantum computer. In this context, existing QA seem best suited for unconstrained problems on instances with densities less than 10%. For constrained problems, the penalty terms used to encode the hard constraints restrict the performance of QA and suggest that these QPU will be less efficient on these problems of comparable size.**

***Index Terms*—Benchmark, Quantum Annealer, Optimization, QUBO, Ising model, maxcut, maximum independent set**

I. Introduction

The past five years have seen the rise of quantum computers with successfully implemented quantum supremacy experiments. These experiments were demonstrated with superconducting qubits [1] and with photonic qubits [2], [3]. More recently, D-Wave claimed that their QA reached quantum supremacy [4]. These demonstrations constitute strong experimental steps towards achieving a useful quantum advantage but are insufficient to demonstrate the utility of quantum computers to solve real-world and useful problems. Combinatorial optimization problems constitute a set of relevant problems to evaluate the potential quantum advantage that could be brought by Noisy Intermediate Scale Quantum (NISQ) process units. The Q-Score is an example of an application-oriented protocol designed to benchmark QPU using random max-cut instances based on Erdős-Rényi graphs [5]. Several frameworks have been recently designed to benchmark annealing-based and gate-based models in a holistic way [6]. One example is the QUARK framework, which explores real industrial use cases [7]. Various national and international initiatives are developing application-oriented frameworks, including the QED Consortium [8] and the BACQ project [9]. These frameworks are completed with attempts to quantify the coverage of instance sets to gauge the fairness of a benchmark experiment [10], [11]. Recent studies gather specific instance sets designed to benchmark quantum computers, such as HamLib [12], which proposes various encodings of Hamiltonians for optimization problems and condensed matter models. The MQTBench library [13] establishes a database of quantum circuits that can be used for gate-based QPU. One can also rely on previously developed reference libraries of problems used for classical computers, such as MQLib [14] and QPLib [15]. The reader can refer to a recent survey done on this subject [16].

The methods described above are based on the same execution scheme: an input instance is defined and a preprocessing routine is used to map the problem on the QPU. This task is called the compilation or transpilation for gate-based quantum computers. It converts the original gate set to a physically realizable gate set and adds SWAP gates to compensate for the limited interconnects of the quantum chip. For QA such as D-Wave Systems [17], the preprocessing step consists of finding a transformation that maps each problem's variable to a set of physical variables that can be straightly encoded to the physical qubits of the quantum annealer. This task is called minor-embedding [18]. Transpilation, as well as minor-embedding tasks, are hard for classical computers and directly impact the quality of the solutions provided by the QPU.

A second approach uses sets of *crafted instances* to benchmark the QPU [16]. These instances are designed to reduce the impact of the preprocessing step and constitute friendly use cases for the QPU but remain hard for classical optimizers. Table I shows recent experiments led on D-Wave QA using this type of instance. Each experiment shows that QA can excel at solving specific sets of instances compared to classical solvers. These instances have two common features: they have many variables and are almost always subgraphs of the qubit layout, which avoids using extra qubits to embed the problem on the quantum chip. *Crafted instances* have been used to compare IBM gate-based QPU to D-Wave QA, leading to the superiority of QA for solving optimization problems [19].

This paper provides a new method to create large sets of *crafted instances* that are near-optimally mapped on a given QPU. Our study mainly focuses on QA, the most mature quantum technology able to approximate large optimization

TABLE I
RELATED WORK USING CRAFTED INSTANCE SETS TO DEMONSTRATE THE PERFORMANCE OF D-WAVE QUANTUM ANNEALERS.

Source	Instance type	Instance topology	#Variables	#Qubits	Tested QPU	Minor-Embedding	Classical solvers
[20] (2021)	BFM*, FBFM*, CBFM*	Chimera graph	2032	2032	*2000Q*	QPU chip sub-graph	Global & local search
[21] (2022)	CBFM*-P	Pegasus graph	5387	5387	*Advantage_System4.1*	QPU chip sub-graph	Global & local search
[22] (2023)	Unweighted Max-cut	3-regular graph	4-320	4-320	*Advantage_System4.1*	QPU chip sub-graph	Exact solver
[4] (2024)	2D & 3D Ising spin glass	square, cubic, diamond, biclique	16-567	16-576	*Advantage_System4.1* and *Advantage2*	QPU chip sub-graph 2 qubits/var (biclique)	Exact solver

*BFM, FBFM and CBFM respectively relates to Biased Ferromagnet, Frustrated Biased Ferromagnet and Corrupted Biased Ferromagnet Ising models.

problems. However, our method can be extended to benchmark gate-based QPU. The creation of the instance set is designed in a reverse fashion: At first, a near-optimal mapping of a complete graph (source graph) on the quantum chip (target graph) is found. The mapping function is then iteratively altered to increase the size of the source graph while decreasing its density. During this process, the target graph is not altered. The set of source graphs is then used to generate instances of optimization problems with various densities to benchmark the performance of QA. Our analysis provides valuable insights into the characteristics of instances that could potentially lead to a quantum advantage. Our finding is that despite near-optimal embeddings, the first generation of D-Wave Advantage QPUs rapidly struggles to find good solutions for instances with densities greater than 0.1.

The remainder of the paper is organized as follows: Section II introduces the basic notions and notations on quantum annealing and minor-embedding. Section III presents the method used to create the *crafted instances* and the technical settings of the experiment. Section IV benchmarks the QPU against classical solvers and discusses the results.

II. NOTATIONS AND DEFINITIONS

Quantum annealers, such as D-Wave systems, constitute a noisy version of the universal adiabatic quantum computer [23]. The evolution of these systems is based on the interpolation of a mixing Hamiltonian H_{M} whose ground state is easy to prepare and a problem Hamiltonian H_{P} whose ground state encodes the solution to the problem. The problem Hamiltonian can be fully specified with a source graph $G_{\mathrm{s}} = (V_{\mathrm{s}}, E_{\mathrm{s}})$:

$$H_{\mathrm{P}} = \sum_{v \in V_{\mathrm{s}}} h_v \sigma_v^z + \sum_{(u,v) \in E_{\mathrm{s}}} J_{uv} \sigma_u^z \sigma_v^z \tag{1}$$

where σ_i^z denotes the Pauli Z operator on variable i. The ground state of H_{p} encodes the solution to the problem:

$$\min C(\mathbf{s}) = \sum_{v \in V_{\mathrm{s}}} h_v s_v + \sum_{(u,v) \in E_{\mathrm{s}}} J_{uv} s_u s_v \tag{2}$$

where $s_i \in \{-1, +1\}$. A Quadratic Unconstrained Binary Optimization (QUBO) cost function with variables $x_i \in \{0, 1\}$ is translated to an Ising cost function (2) using a variable substitution $x_i = \frac{1+s_i}{2}$. When a graph G_{s} cannot be straightly mapped to the qubits of the quantum chip due to connectivity limitations, one has to find a mapping function that maps the source graph G_{s} to a target graph $G_{\mathrm{t}} = (V_{\mathrm{t}}, E_{\mathrm{t}})$ where G_{t} is defined as a subgraph of the quantum chip's graph $G_{\mathrm{t\_QPU}}$. This problem is well-defined in the theory of graph minors [24] and is formally stated as:

Find a function $\phi : V_{\mathrm{s}} \to \mathcal{P}(V_{\mathrm{t}})$ such that :

1) *each vertex $v \in V_{\mathrm{s}}$ is mapped onto a connected subgraph $\phi(v)$ of G_{t}. G_{t} is a subgraph of $G_{\mathrm{t\_QPU}}$.*
2) *each connected subgraph must be vertex disjoint $\phi(v) \cap \phi(v') = \emptyset$, with $v \neq v'$.*
3) *each edge $(u, v) \in E_{\mathrm{s}}$ is mapped onto at least one edge in E_{t} : $\forall (u, v) \in E_{\mathrm{s}}, \exists u' \in \phi(u), \exists v' \in \phi(v),$ such that $(u', v') \in E_{\mathrm{t}}$.*

Let n_ϕ be the number of nodes used in G_{t} to embed the graph G_{s}. A coupling strength is set to each edge of each connected subgraph $\phi(v)$ to maintain ferromagnetic couplings between the physical qubits, which act as a single logical variable. In this paper, we refer to $v \in V_s$ as logical qubits or logical nodes (equivalent to logical variables). We refer to $v \in V_t$ as physical qubits or physical nodes. Each logical node is mapped to a set of physical nodes $\phi(v)$. The density of a graph G_{s} is defined as $d = \frac{2|E_{\mathrm{s}}|}{|V_{\mathrm{s}}|(|V_{\mathrm{s}}|-1)}$.

Two different types of algorithms are mainly used to find the function ϕ. The first set of algorithms takes as input both G_{s} and $G_{\mathrm{t\_QPU}}$. One implementation is the CMR heuristic [25], which has the advantage of working with any graph $G_{\mathrm{t\_QPU}}$. The second set of algorithms takes as input a complete graph G_{s} and is designed for specific $G_{\mathrm{t\_QPU}}$. These algorithms usually take advantage of the structure of $G_{\mathrm{t\_QPU}}$ and produce mappings that are near-optimal for complete graphs. For example, the CME heuristic [26] finds high-quality mappings of complete graphs for D-Wave quantum chips, leveraging the regular structure of the chip and considering inoperable qubits.

III. METHOD

A. Assessing the Quality of an Embedding

This work only considers the number of physical qubits as the embedding quality criteria. The quality of an embedding can be further refined but remains complex to assess [27]. The minimum number of physical nodes required to embed a source graph G_{s} can be lower bounded. Even if not reachable in practice, this lower bound gives an idea of the quality of the embedding found and the distance to an optimal embedding. Let $n_{\phi(v)^*}$ be a lower bound on the optimal number of nodes required to embed a node $v \in V_{\mathrm{s}}$ on the target graph. We assume $G_{\mathrm{t\_QPU}}$ has a regular topology and define c_{phys} as the number of edges per node, which is constant for D-Wave topologies, *e.g.*, $c_{\mathrm{phys}} = 15$ for Pegasus topology. If $\phi(v)^*$ maps $v \in V_{\mathrm{s}}$ to a path (this structure maximizes the potential

connectivity of $\phi(v)^*$), the node v requires at least $n_{\phi(v)^*}$ nodes to be embedded on the target graph G_t:

$$n_{\phi(v)^*} = \left\{ \begin{array}{l} 1 \text{ if } deg(v) \leq c_\mathrm{phys} \\ 2 \text{ if } c_\mathrm{phys} < deg(v) \leq (2c_\mathrm{phys} - 2) \\ \left\lceil \frac{deg(v) - (2c_\mathrm{phys} - 2)}{c_\mathrm{phys} - 2} \right\rceil + 2 \text{ otherwise} \end{array} \right\}. \quad (3)$$

The number of nodes used in the target graph to embed G_s is then $n_\phi \geq \sum_{v \in V_\mathrm{s}} n_{\phi(v)^*}$. This bound is then used to define an overhead ratio r_o that computes the overhead of physical qubits used by the mapping function:

$$r_\mathrm{o} = \frac{n_\phi}{\sum_{v \in V_\mathrm{s}} n_{\phi(v)^*}}. \quad (4)$$

The closer the ratio r_o is to 1, the better. We use this metric to show that our instance generation process creates near-optimally embedded instances.

B. Near-optimal Mapping Generation

Fig. 1 shows the workflow creating graph instances near-optimally embedded on the D-Wave quantum chip. The first step consists of finding the mapping function of a complete graph G_s to a target graph G_t that is a subgraph of $G_{\mathrm{t\_QPU}}$. The CME algorithm finds such mapping (see Fig. 1a). Each logical node $v \in V_\mathrm{s}$ is mapped to a connected subgraph $\phi_\mathrm{CME}(v)$ on the D-Wave chip, which topology is represented by the graph $G_{\mathrm{t\_QPU}}$. One special feature is that the CME algorithm generates subgraphs $\phi_\mathrm{CME}(v)$ that are paths to maximize the potential connectivity of each logical qubit $\phi_\mathrm{CME}(v)$. It is then possible to increase the size of the logical graph G_s by selecting and splitting a random path $\phi_\mathrm{CME}(v)$ in two equal parts (see Fig. 1b and c). The new source graph $G'_\mathrm{s} = (V'_\mathrm{s}, E'_\mathrm{s})$ has a lower density than G_s, but an additional node owing to the split. This last step is repeated to create arbitrarily dense source graphs G'_s, preserving the near-optimal embedding clause in the new mapping function $\phi_\mathrm{nopt}(v)$. As the number of iterations increases, the Graph Edit Distance (GED) between G'_s and G_t shrinks, meaning that the topology of G'_s becomes closer to the topology of G_t. It is important to notice that the set of nodes V_t of G_t is defined by the CME method at the beginning and remains fixed during the steps that generate the different instances G'_s. Hence, all the instances G'_s use the same physical graph G_t. Only the mapping function ϕ_nopt varies between each graph G'_s. The generated graphs G'_s that are very sparse are considered favorable to the quantum computer as they have many logical variables, each mapped to very few or only one physical qubit. On the contrary, graphs G'_s with high density have each vertex v mapped to long chains of physical qubits $\phi_\mathrm{nopt}(v)$. It suggests that this type of instance is less favorable to the quantum annealer and will be more easily solved by classical computers.

We evaluate the efficiency of our method using a complete graph of size $n = 100$. The CME method finds the mapping function of the complete graph G_s to D-Wave *Advantage6.4*, using 982 physical qubits. The method described in the previous section is used to generate source graphs G'_s of diverse densities $d \in \{0.9, 0.8, ..., 0.1\}$ with their associated near-optimal mapping ϕ_nopt. For each source graph G'_s, the CMR method is run 100 times with a time limit set to 5 minutes to find a second mapping $\phi_{\mathrm{100\_CMR}}$ that uses the least number of nodes in $G_{\mathrm{t\_QPU}}$. Fig. 2a compares the ratio r_o for each mapping (ϕ_nopt and $\phi_{\mathrm{100\_CMR}}$). It clearly shows that ϕ_nopt mappings use a number of nodes close to the optimal bound defined in (3). The mappings $\phi_{\mathrm{100\_CMR}}$ use more nodes that ϕ_nopt for every instances. The worst case behavior of $\phi_{\mathrm{100\_CMR}}$ is on sparse instances, whereas ϕ_nopt produces embeddings that can be considered near-optimal. We then use the graphs G'_s to create max-cut instances and compute the ratio of cut sizes obtained using each mapping ϕ_nopt and $\phi_{\mathrm{100\_CMR}}$ with D-Wave *Advantage6.4*. Fig. 2b shows that the best-cut size is increased when using the mapping ϕ_nopt instead of the mapping $\phi_{\mathrm{100\_CMR}}$. It confirms that in addition to using fewer qubits, the mappings found by our method increase the performance of the D-Wave *Advantage6.4*.

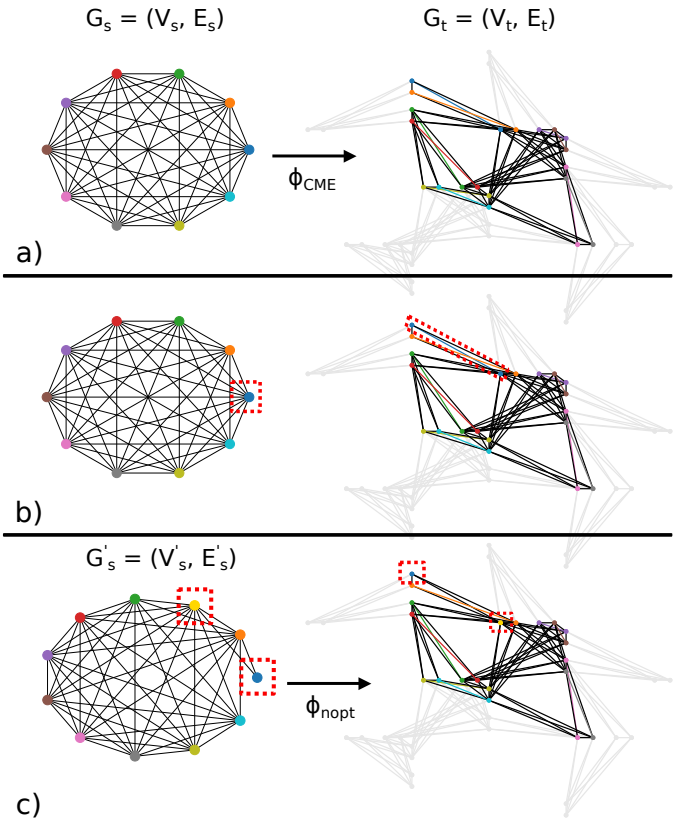

Fig. 1. Instance generation with iterative chain split. a) The algorithm starts with a complete graph embedding using CME method on the graph $G_{\mathrm{t\_QPU}}$. Each node in the source graph G_s is mapped to a chain in the target graph G_t. b) A random logical node $v \in V_\mathrm{s}$ is selected. c) The corresponding chain $\phi_\mathrm{CME}(v)$ is split into two parts to create a new logical node in the source graph. It changes the mapping function to ϕ_nopt. Steps b. and c. are repeated on G'_s until the desired density in the logical graph is reached.

C. Instance Set

In the rest of the paper, we generate instances from a complete graph of size $n = 174$. It is the largest clique that

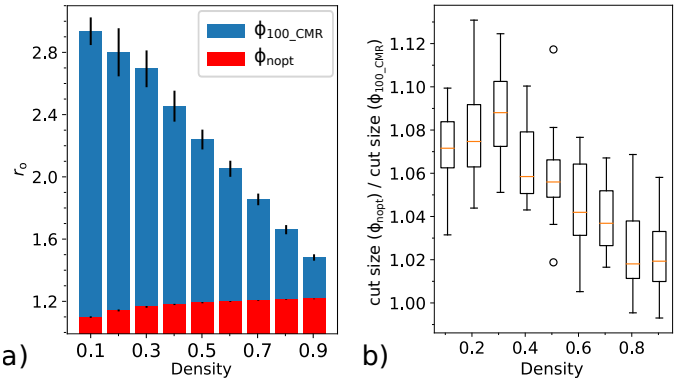

Fig. 2. Performance comparison of mapping functions found by our method ϕ_{nopt} and the best among 100 tries of CMR method ϕ_{100_CMR}. 30 instances are generated for each density. a) Comparison using the overhead ratio (see (4)). Error bars show the standard deviations b) Ratio of the best cut size obtained for each mapping for each instance.

can be embedded with the CME method on *Advantage6.4*, using 2918 physical qubits, representing approximately 52% of the physical working qubits of the chip. Using the method described above, we generate graphs of various densities G'_{s} with their associated near-optimal mapping ϕ_{nopt} on *Advantage6.4* (see Table II for instances properties). The source graphs G'_{s} are then used to create instances of three different optimization problems with 30 instances for each density. The two first problems are unweighted and weighted max-cut problems defined as:

$$\min C(\mathbf{s}) = \sum_{(u,v)\in E'_{\text{s}}} J_{uv} s_u s_v \tag{5}$$

where $s_u, s_v \in \{-1, +1\}$. $J_{uv} = 1$ for unweighted max-cut and $J_{uv} \in \{-\frac{128}{128}, ..., -\frac{1}{128}, \frac{1}{128}, ..., \frac{128}{128}\}$ for weighted max-cut. Weights are used to challenge the precision and long-range coherence of the QPU. The precision is the same as in [4]. The third problem is a weighted Maximum Independent Set (MIS) defined by the cost function:

$$\min C(\mathbf{x}) = -\sum_{v\in V'_{\text{s}}} \omega_v x_v + \sum_{(u,v)\in E'_{\text{s}}} \omega_{uv} x_u x_v \tag{6}$$

where $x_u, x_v \in \{0, 1\}$, $\omega_v \in \{\frac{1}{128}, ..., \frac{128}{128}\}$ and $\omega_{uv} = 2$. ω_{uv} acts as the penalty term used to enforce the hard constraints for D-Wave and Tabu Search (TS).

D. Solver Settings

1) Advantage6.4 settings: The performance of the quantum annealer is evaluated using an access time limit of 1 second. Each instance is mapped on the QPU using the mapping ϕ_{nopt}. Fig. 3 shows the repartition of the processing time considering different annealing times $\{1, 10, 100, 1000\}\mu$s with respectively $\{5000, 4780, 3310, 820\}$ shots. Fig. 4 shows the performance of D-Wave considering different values of annealing time for each problem. The optimal annealing time is instance-dependent and cannot be estimated before running the annealing process in a noisy context. For each problem, we select a fixed annealing time that produces the best results on average for low (0.02), medium (0.5) and high (0.9) densities. Therefore, we choose 1μs and 5000 shots for unweighted max-cut, 1000μs and 820 shots for weighted max-cut and 1μs and 5000 shots for weighted MIS problems. The D-Wave *uniform_torque_compensation* method is used to set the ferromagnetic couplings between the physical qubits $\phi_{\text{nopt}}(v)$. The weight associated with a single qubit h_v is uniformly spread over the physical qubits $\phi_{\text{nopt}}(v)$. The same method is used for edges. A majority vote is used to unembed the problem. We do not use annealing offset or spin reversal methods. MIS instances are post-processed to avoid constraint violations. We iteratively and randomly select edges that violate the constraint and only keep the node with the higher weight in the independent set. The benchmark is done using the best solution found among all shots for each instance.

2) Tabu Search settings: The TS algorithm [28] is run directly on the instance associated with the source graph G'_{s}. The processing time limit is set to 1s and the tenure length is set to $|V'_{\text{s}}|/4$. The TS uses a single processor core *AMD EPYC 7702P*. MIS instances are post-processed with the same method used for D-Wave.

3) Gurobi settings: Gurobi [29] is run with a time limit of $60s$. It establishes a reference solution with which the TS

TABLE II
PROPERTIES OF G'_{s} GRAPHS USED TO BUILD OPTIMIZATION PROBLEM INSTANCES. THESE GRAPHS ARE MAPPED ON 2918 QUBITS.

Density	Avg $\lvert V'_{\text{s}}\rvert$	r_{o}	Density	Avg $\lvert V'_{\text{s}}\rvert$	r_{o}
0.02	1318	1.06	0.1	565	1.14
0.03	1062	1.08	0.2	395	1.16
0.04	912	1.10	0.3	321	1.17
0.05	810	1.11	0.4	277	1.18
0.06	737	1.12	0.5	248	1.19
0.07	680	1.12	0.6	226	1.19
0.08	635	1.13	0.7	209	1.19
0.09	597	1.13	0.8	195	1.20
			0.9	184	1.20

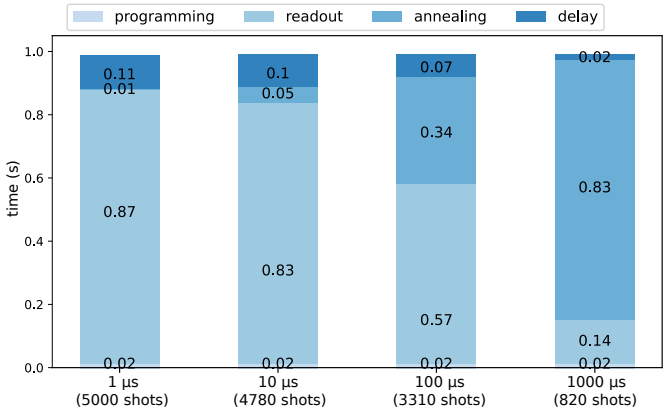

Fig. 3. QPU access time repartition for different annealing times per shot with a total running time limit of $1s$. Each color represents the fraction of time used for programming the QPU, annealing the n shots, reading the results of the n shots, and delays between the n shots.

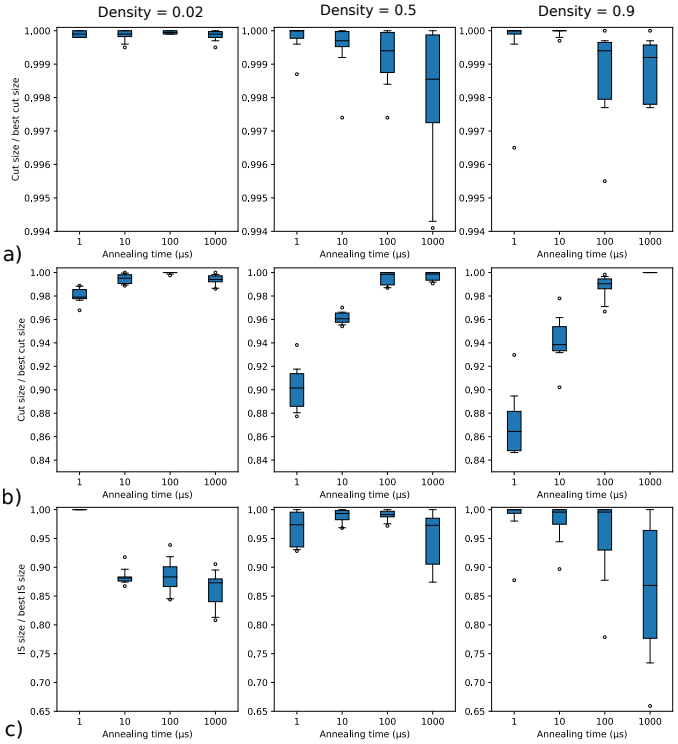

Fig. 4. Annealing time scan with a quantum processing time limit of 1s for three types of problems with various densities. a) Unweighted maxcut instances b) Weighted maxcut instances c) Weighted MIS.

and D-Wave are compared. The solver is set with default parameters. The solver is run on the instance associated with the source graph G'_{s}. For the MIS problem, the constraints are directly encoded with inequalities instead of penalties. Gurobi is parallelized on 20 cores of a processor *AMD EPYC 7702P*.

4) Random solver settings: A random solver is used on MIS instances. It iteratively and randomly selects a node in the set V'_{s} and removes from V'_{s} all the neighbors of this node. This method generates the same number of solutions generated by D-Wave, i.e., 5000 solutions. The best MIS is then selected among these shots. We did not use this method for the max-cut problem as random solution cuts are far from competitive with D-Wave and TS solutions.

IV. Results and Discussion

We compare the performance of D-Wave *Advantage6.4* against classical solvers to approximate solutions of optimization problems introduced in the previous section. Table II shows the average number of logical variables for each density processed in Fig. 5. We recall that the density of the graph is inversely proportional to the number of variables in the graph G'_s. The results of D-Wave and the TS are expressed as a ratio of a reference solution computed with Gurobi with a runtime limit set to 60s. In general, D-Wave performs well with respect to TS on sparse instances under 0.1 density. Under this density, the logical problem has more than 565 variables, which limits the performance of the TS due to the large size of the solution space. We observed that the TS did not have sufficient processing time for large instances, leading to almost random solutions for these instances. It explains why the box plots are stretched for TS for small densities. The second reason that can explain the nice performance of the QA for small density instances is that the GED between G'_{s} and G_{t} shrinks as G'_{s} density decreases (i.e., G'_{s} topology becomes close to G'_{t} topology). It denotes instances favorable to the quantum computer, with only a few variables mapped to several physical qubits. For very sparse instances ($d < 0.03$), the *Advantage6.4* outperforms Gurobi on few instances of unweighted and weighted max-cut, showing that the QPU may be useful against classical methods for very sparse but large instances with a GED close to G_{t}. Denser instances become easier for classical solvers as the search space reduces exponentially with density but stays approximately the same for the QPU as the number of physical qubits remains the same due to the embedding. In addition, long chains of physical qubits increase the rate of chain breaks. Increasing the strength of ferromagnetic couplings can prevent this issue, but it also adds extra energy to the system and can induce the rescaling of Hamiltonian weights due to the finite programming range of each coupler, which challenges the precision of the QPU. TS and QA do not seem adapted to approximate solutions to constrained problems such as MIS. In this benchmark, these methods cannot compete with the random solver used to generate maximal independent sets. We observed that for almost all instances, the TS and QA produced solutions that violate the independent set constraint (encoded as soft constraints in the cost function with penalty terms). Instead, Gurobi is efficient for this problem as the constraints encoded with inequalities ease the branch and bound resolution.

Several points have to be clarified and considered carefully to avoid misleading conclusions. The aim of this benchmark is not to study the scaling of the QPU but rather to determine when the quantum computer performs best at solving instances that use the same physical requirements (in our case, the number of physical qubits). The settings of this experiment only involved tuning the annealing time, which has a huge impact when working on time-limited tasks. We did not want to include other complex parameter optimization such as the advanced tuning of the chain strength [30]–[32], annealing schedules [33], [34] or advanced calibration methods called shimming [35]. The main idea of our study is to provide an objective view of D-Wave's average performance without employing complex preprocessing methods. Nevertheless, the study of the trade-offs brought by these methods represents a relevant perspective for future work. The parameters of the classical heuristics can be further optimized, and more specific algorithms can be used (see [36] for sparse maxcut instances).

The minor-embedding task is a step that is usually excluded from quantum benchmark studies, which almost only consider the pure annealing time without considering delays, readouts and programming time. A huge amount of classical preprocessing time can be spent to optimize the mapping function, making comparing existing studies difficult. Instead, our approach permits the evaluation of the QPU on near-

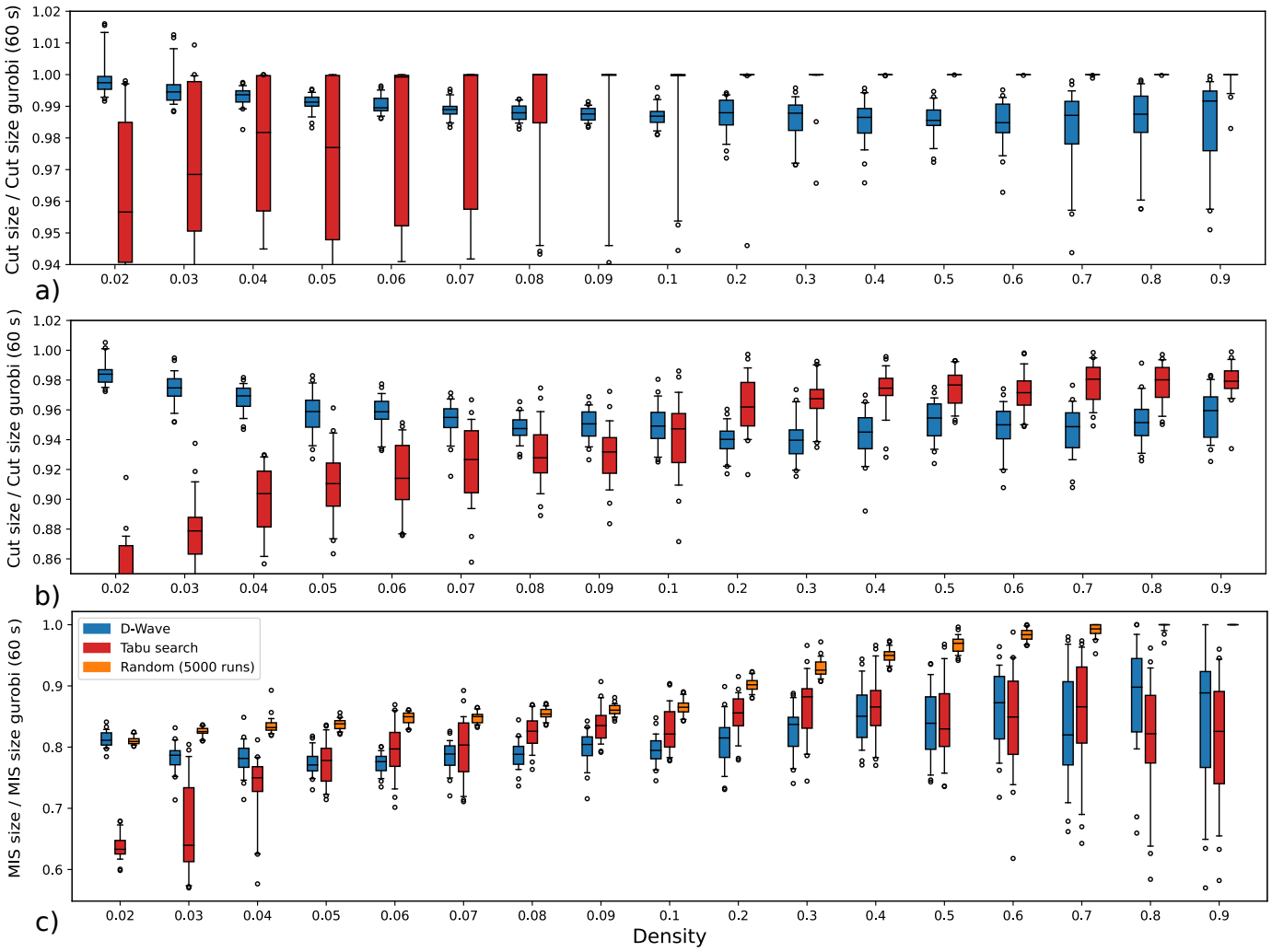

Fig. 5. Benchmark of D-Wave *Advantage6.4* and TS (1s time limit) expressed as ratios of a reference solution found by Gurobi in 60s. Each instance's size is provided in Table II. Boxes extend from Q1 (quartile) to Q3 with a line at the median. Whiskers are delimited by the farthest data point lying within 1.5x the inter-quartile range and flier points are data points beyond the whiskers. a) Unweighted max-cut b) Weighted max-cut c) MIS with a random solver.

optimally mapped instances, giving insights into the properties that leverage the power of QA.

To conclude the discussion, this benchmark methodology is not fair and representative of the performance of D-Wave systems' ability to solve real-world instances. It rather acts as a performance evaluation of D-Wave systems' ability to solve large optimization instances based on an ideal mapping of the instance on the QPU, using QA default settings. This methodology could be used to detect optimization problems that could benefit from a quantum advantage.

V. Conclusion

This article introduced a new method to create source graphs of various densities near-optimally mapped on a QA. This set of graphs was then used to generate large optimization problem instances to benchmark D-Wave QPU against classical solvers. Considering an equal processing time limit of 1s, the QPU was competitive with TS for instances with densities inferior to 0.1 on the weighted max-cut problem. It even outperformed Gurobi solver on some really sparse max-cut instances, whereas Gurobi had 60 times the processing time of D-Wave QPU. Conversely, the QPU seemed less efficient at approximating similar-sized constrained problems. It is important to remember that this set of *crafted instances* is very favorable and specifically designed for D-Wave QA, especially when the instances are very sparse. Our results can be seen as the benchmark of the default behavior of D-Wave QPU on large instances of ideal shape. For real instances, the mapping function found by embedding methods such as CMR will not be as efficient as the one designed in this paper.

A relevant perspective is to study other optimization problems using this methodology to build the instance set. It could also be used to evaluate the benefits of advanced preprocessing methods of quantum annealers. Finally, this methodology can be adapted to benchmark variational quantum algorithms on gate-based quantum computers, considering the generation of instances with optimal planted swapping networks.

References

[1] F. Arute, K. Arya, R. Babbush, *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, p. 505–510, 2019.

[2] H.-S. Zhong, H. Wang, Y.-H. Deng, *et al.*, "Quantum computational advantage using photons," *Science*, vol. 370, no. 6523, pp. 1460–1463, 2020.

[3] L. S. Madsen, F. Laudenbach, M. F. Askarani, *et al.*, "Quantum computational advantage with a programmable photonic processor," *Nature*, vol. 606, no. 7912, p. 75–81, 2022.

[4] A. D. King, A. Nocera, M. M. Rams, *et al.*, "Computational supremacy in quantum simulation," *arXiv preprint arXiv:2403.00910*, 2024.

[5] S. Martiel, T. Ayral, and C. Allouche, "Benchmarking quantum coprocessors in an application-centric, hardware-agnostic, and scalable way," *IEEE Transactions on Quantum Engineering*, vol. 2, pp. 1–11, 2021.

[6] D. Mills, S. Sivarajah, T. L. Scholten, and R. Duncan, "Application-motivated, holistic benchmarking of a full quantum computing stack," *Quantum*, vol. 5, p. 415, 2021.

[7] J. R. Finžgar, P. Ross, L. Hölscher, *et al.*, "Quark: A framework for quantum computing application benchmarking," in *2022 IEEE international conference on quantum computing and engineering (QCE)*, pp. 226–237, IEEE, 2022.

[8] T. Lubinski, S. Johri, P. Varosy, *et al.*, "Application-oriented performance benchmarks for quantum computing," *IEEE Transactions on Quantum Engineering*, 2023.

[9] F. Barbaresco, L. Rioux, C. Labreuche, *et al.*, "Bacq–application-oriented benchmarks for quantum computing," *arXiv preprint arXiv:2403.12205*, 2024.

[10] T. Tomesh, P. Gokhale, V. Omole, *et al.*, "Supermarq: A scalable quantum benchmark suite," in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 587–603, IEEE, 2022.

[11] V. Gilbert, S. Louise, and R. Sirdey, *TAQOS: A Benchmark Protocol for Quantum Optimization Systems*, p. 168–176. Springer Nature Switzerland, 2023.

[12] N. P. Sawaya, D. Marti-Dafcik, Y. Ho, *et al.*, "Hamlib: A library of hamiltonians for benchmarking quantum algorithms and hardware," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 2, pp. 389–390, IEEE, 2023.

[13] N. Quetschlich, L. Burgholzer, and R. Wille, "Mqt bench: Benchmarking software and design automation tools for quantum computing," *Quantum*, vol. 7, p. 1062, 2023.

[14] I. Dunning, S. Gupta, and J. Silberholz, "What works best when? a systematic evaluation of heuristics for max-cut and qubo," *INFORMS Journal on Computing*, vol. 30, no. 3, p. 608–624, 2018.

[15] F. Furini, E. Traversi, P. Belotti, *et al.*, "Qplib: a library of quadratic programming instances," *Mathematical Programming Computation*, vol. 11, no. 2, p. 237–265, 2018.

[16] A. Abbas, A. Ambainis, B. Augustino, *et al.*, "Quantum optimization: Potential, challenges, and the path forward," *arXiv preprint arXiv:2312.02279*, 2023.

[17] "D-wave system. solver properties and parameters." https://docs.dwavesys.com/docs/latest/doc_solver_ref.html [Accessed 20-04-2024].

[18] V. Choi, "Minor-embedding in adiabatic quantum computation: I. the parameter setting problem," *Quantum Information Processing*, vol. 7, pp. 193–209, 2008.

[19] E. Pelofske, A. Bärtschi, and S. Eidenbenz, "Quantum annealing vs. qaoa: 127 qubit higher-order ising problems on nisq computers," in *International Conference on High Performance Computing*, pp. 240–258, Springer, 2023.

[20] Y. Pang, C. Coffrin, A. Y. Lokhov, and M. Vuffray, "The potential of quantum annealing for rapid solution structure identification," *Constraints*, vol. 26, no. 1, pp. 1–25, 2021.

[21] B. Tasseff, T. Albash, Z. Morrell, *et al.*, "On the emerging potential of quantum annealing hardware for combinatorial optimization," *arXiv preprint arXiv:2210.04291*, 2022.

[22] T. Lubinski, C. Coffrin, C. McGeoch, *et al.*, "Optimization applications as quantum performance benchmarks," *arXiv preprint arXiv:2302.02278*, 2023.

[23] T. Albash and D. A. Lidar, "Adiabatic quantum computation," *Reviews of Modern Physics*, vol. 90, no. 1, p. 015002, 2018.

[24] N. Robertson and P. Seymour, "Graph minors .xiii. the disjoint paths problem," *Journal of Combinatorial Theory, Series B*, vol. 63, no. 1, p. 65–110, 1995.

[25] J. Cai, W. G. Macready, and A. Roy, "A practical heuristic for finding graph minors," *arXiv preprint arXiv:1406.2741*, 2014.

[26] T. Boothby, A. D. King, and A. Roy, "Fast clique minor generation in chimera qubit connectivity graphs," *Quantum Information Processing*, vol. 15, pp. 495–508, 2016.

[27] V. Gilbert and J. Rodriguez, "Discussions about high-quality embeddings on Quantum Annealers," in *EU/ME meeting*, (Troyes, France), 2023.

[28] G. Palubeckis, "Multistart tabu search strategies for the unconstrained binary quadratic optimization problem," *Annals of Operations Research*, vol. 131, no. 1–4, p. 259–282, 2004.

[29] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2023.

[30] V. Gilbert and S. Louise, "Quantum annealers chain strengths: A simple heuristic to set them all," *arXiv preprint arXiv:2404.05443*, 2024.

[31] D. Willsch, M. Willsch, C. D. Gonzalez Calaza, *et al.*, "Benchmarking advantage and d-wave 2000q quantum annealers with exact cover problems," *Quantum Information Processing*, vol. 21, no. 4, p. 141, 2022.

[32] T. V. Le, M. V. Nguyen, T. N. Nguyen, *et al.*, "Benchmarking chain strength: An optimal approach for quantum annealing," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, IEEE, 2023.

[33] J. I. Adame and P. L. McMahon, "Inhomogeneous driving in quantum annealers can result in orders-of-magnitude improvements in performance," *Quantum Science and Technology*, vol. 5, no. 3, p. 035011, 2020.

[34] M. Khezri, X. Dai, R. Yang, *et al.*, "Customized quantum annealing schedules," *Physical Review Applied*, vol. 17, no. 4, 2022.

[35] K. Chern, K. Boothby, J. Raymond, *et al.*, "Tutorial: Calibration refinement in quantum annealing," *arXiv preprint arXiv:2304.10352*, 2023.

[36] D. Rehfeldt, T. Koch, and Y. Shinano, "Faster exact solution of sparse maxcut and qubo problems," *Mathematical Programming Computation*, vol. 15, no. 3, p. 445–470, 2023.