The Quantum Computing Company™

The D-Wave Advantage System: An Overview

TECHNICAL REPORT

Catherine McGeoch and Pau Farré

2020-09-25

Overview

This report presents a high-level overview of the D-Wave Advantage™ quantum computer, with a comparison to its predecessor, the D-Wave 2000Q™ system. The Advantage quantum processing unit (QPU) can hold application inputs that are almost three times larger, on average, than those that fit on the D-Wave 2000Q QPU. Beyond the capability to read bigger problems, the Advantage QPU can deliver better performance on application-relevant inputs.

CONTACT

Corporate Headquarters
3033 Beta Ave
Burnaby, BC V5G 4M9
Canada
Tel. 604-630-1428

US Office
2650 E Bayshore Rd
Palo Alto, CA 94303

Email: info@dwavesys.com

www.dwavesys.com

14-1049A-A
D-Wave Technical Report Series

Notice and Disclaimer

Summary

This report presents a high-level overview of D-Wave's Advantage™ quantum computer, launched in September 2020. Technological advances in the design of the quantum processing unit (QPU) at its core make Advantage by far the largest and most powerful quantum computer in existence today. The report compares Advantage to its predecessor, the D-Wave 2000Q, in terms of their physical properties and performance on application-relevant inputs. Key findings are summarized below.

- Advantage contains at least 5,000 qubits, about 2.5 times more than found in a D-Wave 2000. The number of couplers per qubit has increased from 6 to 15, for a total of at least 35,000 couplers, representing about a six-fold increase over the earlier system.
- More qubits and couplers means that larger application problems can be solved directly on the Advantage QPU. An Advantage QPU can hold inputs between 1.9 and 4.6 times larger than similarly-structured inputs that fit on the QPU inside the D-Wave 2000Q, about 3 times larger on average.
- More couplers per qubit means that application problems can be mapped more compactly onto the Advantage QPU. Compactness is measured by *chain length*: chains on Advantage QPUs are typically less than half as long as chains on 2000Q QPUs.
- Beyond the capability to read and solve larger inputs, shorter chains mean that the Advantage system can sometimes find better-quality solutions than the D-Wave 2000Q; it also means that Advantage can find same-quality solutions faster. Three case studies demonstrate equal or superior performance from Advantage compared to the D-Wave 2000Q:
  - On clique problems, the Advantage system found optimal solutions on inputs that were 40 percent larger than the largest size that could fit on the 2000Q system. In cases where both found optimal solutions, Advantage found them faster, up to 30 times faster in terms of pure anneal time, or about 6 times faster in wall-clock time. Overall, the Advantage system found better solutions on 46 percent of inputs, over 7 times more often than the 2000Q system, which won on just 6 percent of inputs (the remaining 48 percent of cases were ties).
  - On inputs for NAE3SAT problem, performance of the two QPUs was quite similar, with a slight edge for Advantage in some cases. In one test, the Advantage QPU found better solutions on 65.5 percent of inputs, compared to just 3.9 percent of inputs for the 2000Q QPU (about 16 times more often). In another test of approximation performance, it found solutions that were 2.8 percent of optimal, a 14 percent improvement over solutions from the 2000Q QPU.
  - On inputs for 3D lattice problems, the Advantage QPU found optimal solutions up to 10 times faster than the 2000Q QPU, considering pure anneal times.
- The report also shows how software tools from the Ocean developers toolkit can be used to improve raw QPU outputs. This type of hybrid strategy combining the best features of quantum and classical computation can often produce better results than either approach alone.

Contents

1 Introduction

This report presents a high-level overview of D-Wave's Advantage™ quantum computer, launched in September 2020. The quantum processing unit (QPU) at Advantage's core represents the culmination of design innovations and technological advances brought about by D-Wave engineers and scientists. Advantage is by far the largest and most powerful quantum computer in existence today.

Section 2 gives an overview of new design features in the Advantage QPU, in comparison to that of its predecessor, the D-Wave 2000Q QPU. Section 3 presents a brief performance analysis comparing the Advantage and 2000Q systems. Section 4 contains some final remarks. The key results in this report are summarized below.

- An Advantage QPU contains at least 5,000 qubits, about 2.5 times more than found in a 2000Q QPU. The number of couplers per qubit has increased from 6 to 15, for a total of at least 35,000 couplers, about a six-fold increase over the total number found in 2000Q QPUs.
- More qubits and couplers mean that larger application problems can be solved directly on the Advantage QPU. Section 2 shows that the largest inputs that fit on the Advantage QPU are 2.9 times larger on average than similarly-structured inputs that fit on the 2000Q QPU. More generally, the Advantage QPU can typically hold inputs that are between 2 and 4 times bigger than similarly structured inputs that fit on a 2000Q QPU.
- More couplers per qubit means that application problems can be mapped more compactly onto the Advantage QPU. Compactness is measured by *chain length*: we observe that chains on Advantage QPUs are typically half as long as chains on 2000Q QPUs.
- Beyond the capability to read and solve larger inputs, shorter chains mean that the Advantage QPU can return better-quality solutions when both QPUs are given the same computational resources. It also means that the Advantage QPU can return same-quality solutions faster than the 2000Q QPU. Section 3 describes three case studies that demonstrate superior performance from the Advantage system:
  - In a study using clique problems, the Advantage QPU found optimal solutions on inputs that were 40 percent larger than those solved to optimality on the 2000Q system (from $n = 50$ to $n = 70$). In cases where both QPUs found optimal solutions, the Advantage QPU was faster, up to 30 times faster in terms of pure anneal time and about 6 times faster in terms of wall clock time (which includes system and I/O overhead). On average, the Advantage system found better solutions on 46 in percent of inputs, about 7.7 times more often than the 2000Q system, which won on just 6 percent of inputs (the remaining 48 percent of cases were ties, e.g. when both found optimal solutions).
  - In a study using inputs for the Not-all-equal 3-Satisfiability (NAE3SAT) problem, the performance profiles for the two QPUs were quite similar, with a slight edge for Advantage in some cases. In one comparison of solution quality, the Advantage QPU found better solutions in 65.5 percent of inputs, compared to

just 3.9 percent of cases for the 2000Q QPU (about 16 times more often). In another test of approximation performance, the Advantage QPU found solutions within 2.8 percent of optimal, compared to 2000Q solutions within 3.2 percent of optimal, which represents a 14 percent improvement.
  - On inputs for 3D lattice problems (described more fully in [1]), the Advantage QPU found optimal solutions about 10 times faster than the 2000Q QPU, at the largest problem size that fits on both, considering pure anneal times.

- Section 3 also shows how software utilities from the Ocean developer's toolkit can improve performance of quantum annealing processors. This *hybrid* approach combining the best features of quantum and classical computation methods often yields better results than either method used alone.

Learn more about D-Wave products and services. The 2000Q and Advantage quantum computers, the Leap web portal, and the Ocean software developer's toolkit, which contains open-source repositories of tools, tutorials, and documentation, are available to the public (in North America, Europe, Japan, Australia and India) for limited small-scale use at no cost. Larger blocks of QPU and system time are also available for purchase from D-Wave or third-party providers.

The D-Wave hybrid solver service (HSS) provides users with solutions to inputs for combinatorial optimization problems that are too large to fit onto current-generation QPUs, on a subscription bases. As of September 2020, the suite of hybrid solvers has been upgraded to incorporate an Advantage system as their back-end quantum query server, and to accept inputs containing up to 20,000 fully connected nodes, and up to 200 million total input weights. A companion D-Wave technical report [2] describes HSS properties and performance.

Visit *dwavesystems.com* to learn more about the Advantage QPU, the Leap and Ocean software stack, and HSS.

2 New Advantage Features

In a quantum annealing system, the *hardware graph topology* describes the pattern of physical connections for qubits and the couplers between them. The most important and obvious difference between 2000Q and Advantage QPUs is the upgrade from Chimera to the Pegasus topology, as shown in Figure 1.

The figure compares a C6 Chimera graph — a 6-by-6 grid of unit cells – with a P4 Pegasus graph, which contains 27 unit cells on a diagonal grid, plus partial cells around the perimeter (note that unit cells on the Pegasus graph contain four extra couplers). Both graphs contain about the same number of qubits: the C6 has 288 and the P4 has 264. However, Chimera has just 6 couplers per qubit while Pegasus has 15 couplers per qubit. This creates the visibly more complex connection structure in Pegasus.

In addition to greater size and connectivity, the Pegasus graph has been modified in other ways to improve performance. For example, Pegasus contains triangles, which means that more general (nonbipartite) graph structures can be represented directly by the physical hardware. See [3] for discussion of these and other properties.

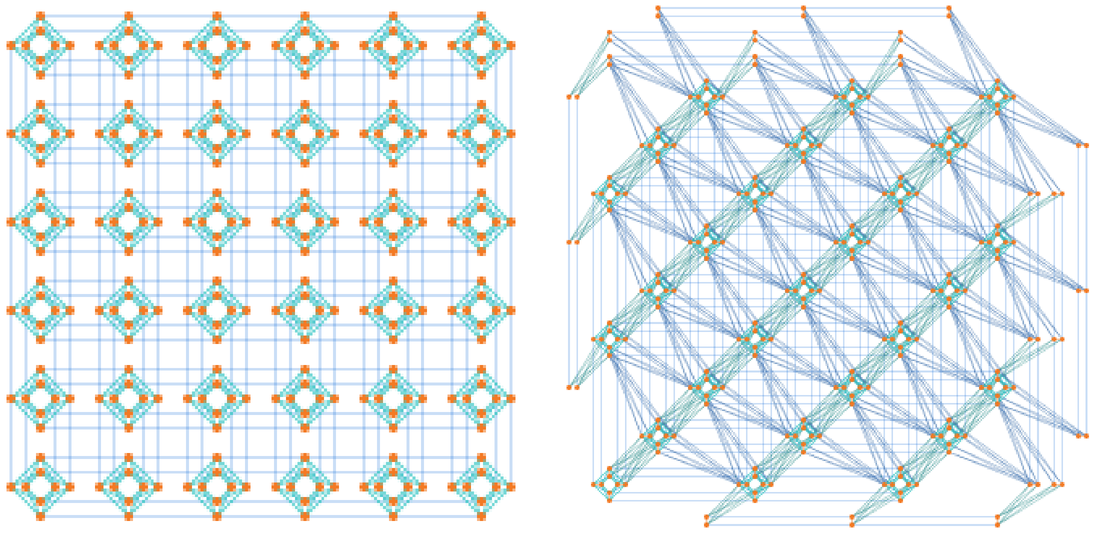


Figure 1: A C6 Chimera graph (left) with 36 unit cells containing 288 qubits. A P4 Pegasus graph (right) with 27 unit cells and several partial cells, containing 264 qubits. The comparatively rich connectivity structure of the P4 is clearly seen.

	2000Q	Advantage
Graph topology	Chimera	Pegasus
Graph size	C16	P16
Number of qubits	> 2000	> 5000
Number of couplers	> 6000	$> 35,000$
Couplers per qubit	6	15

Table 1: Typical characteristics of Chimera- and Advantage-generation QPUs.

Note that the physical hardware graphs inside the 2000Q and Advantage QPUs are much larger than shown in the figure: the former contains a C16 and the latter contains a P16.

Table 1 presents a comparison of the two quantum processor models in terms of typical component counts. The exact number of active qubits and couplers can vary across individual QPU chips, because a small percentage of components may fail to meet technical specifications. The table shows the minimum number of active qubit and couplers in any QPU chip that is made available to the public.

2.1 More Compact Embeddings

An application input $\mathcal{I}$ for some problem $\mathcal{P}$ typically goes through two transformation steps before being sent to a D-Wave QPU. The first step is to reformulate $\mathcal{I}$ as an input for the quadratic unconstrained binary optimization problem (QUBO) (or for the equivalent Ising Model problem (IM)). This reformulation step uses standard cookbook techniques from NP-completeness theory and is not further discussed here; see [4] for more.

The resulting QUBO input is represented by a so-called *logical graph* G containing n nodes and m edges; the input is specified by the pair (h, J) consisting of a set of n node weights $h = \{h_i\}$ and m edge weights $J = \{J_{ij}\}$. In order to be solved directly on a given QPU, the graph G must be mapped onto the *physical hardware graph* of qubits and couplers, e.g. Chimera or Pegasus.

This mapping is normally performed by software utilities available in the Ocean developers toolkit, using a technique called *minor embedding* (or informally, *embedding*). Figure 3 shows an example graph G before and after minor-embedding onto a P4 Pegasus graph. The P4 embedding contains color-coded *chains*: each chain of qubits and couplers corresponds to a single node of the same color in the logical graph. For example, the blue ovals highlight a pink node that is mapped to a chain of two qubits during minor-embedding. The logical edges of G are shown in black on the P4; qubits and couplers that are not used in the embedding are shown in light grey.

The greater connectivity of Pegasus compared to Chimera means that the same logical graph can be minor-embedded more compactly onto the physical hardware. This fact has two important consequences:

- Minor embeddings on the Advantage QPU use fewer qubits per node, so the same number of qubits can hold larger logical graphs. This boosts the input capacity of the Advantage QPU beyond that obtained by simply increasing the qubit count.
- Advantage embeddings generally require shorter chains than 2000Q embeddings. Shorter chains are stronger chains, which means the Advantage QPU can return better-quality solutions than the 2000Q QPU.

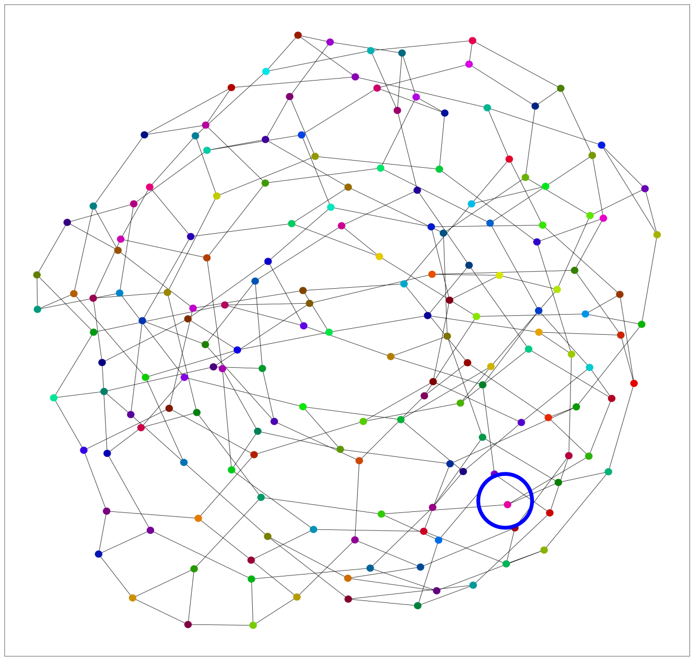
These two points are illustrated and quantified in the following sections.

2.2 Bigger Inputs, Shorter Chains

The *degree* of a graph is the maximum number of edges per node. The logical graph of Figure 3 has degree $d = 3$, which makes it fairly sparse and relatively easy to minor-embed into the Pegasus graph. In contrast, a fully-connected graph on n nodes, known as a *clique*, has degree $d = n - 1$. Cliques are among the hardest graphs to embed on hardware topologies having limited connectivity, and tend to produce the longest chains of any logical graph type.

The graph at the top of Figure 3 compares clique embeddings obtained using an Advantage P16 (orange) and a 2000Q C16 (blue). Each curve shows the largest clique size that can be embedded using a given chain length. With chains of length 17, the P16 can hold cliques of size up to $n = 124$, while the C16 can only hold cliques of size up to $n = 64$. A graph of size $n = 64$ requires chains of length 17 on the C16 but only needs chains of length 7 on the P16. That is, compared to the 2000Q QPU, the Advantage QPU can embed cliques that are about twice as big, using chains that are about half as long.

The middle table compares maximum embeddable graph sizes for other representative graph types, ordered roughly by increasing degree. The first two columns show the graph family and the typical number of edges per node. For comparison to embedded inputs, the top row shows maximum sizes for native inputs, represented by the physical hardware graph, which require no chains.



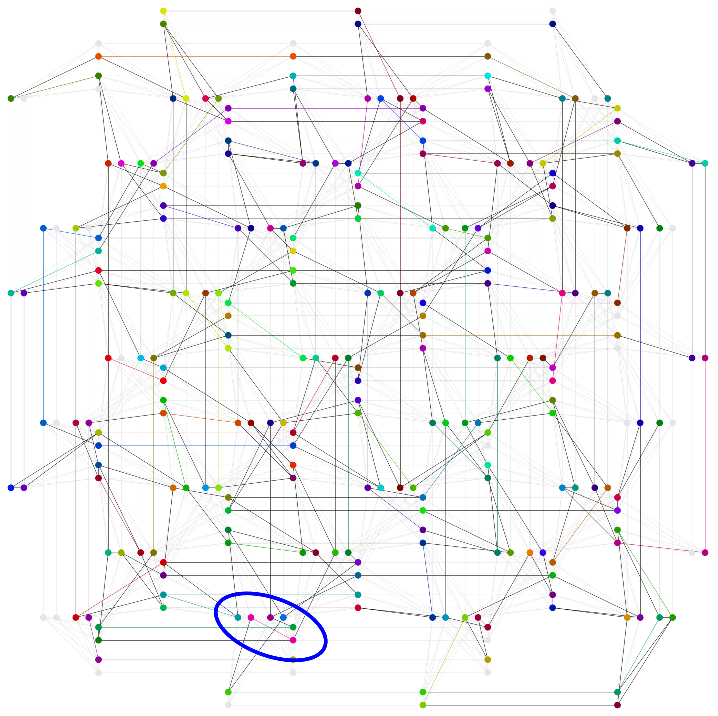
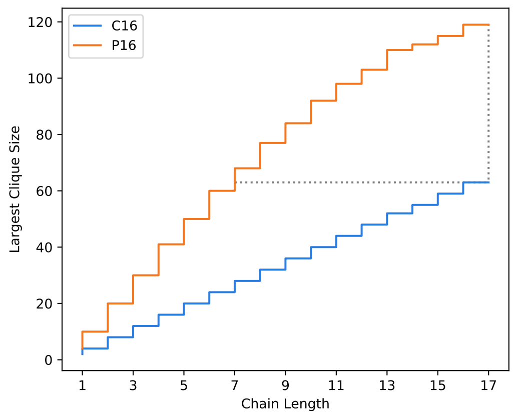


Figure 2: Minor embedding of a general QUBO graph (top) onto a P4 Pegasus graph (bottom). In the physical embedding, chain edges are colored to match their nodes, and logical edges are black. The blue circle and oval highlight a pink node that is mapped to a chain of two qubits during minor-embedding.



Maximum Graph Sizes				
Logical Graph	Degree	C16	P16	Ratio
		n	n	P16/C16
Native	Chi=6, Peg=15	2030	5436	2.7
3D lattice (w/defects)	6	512	2354	4.6
NAE3SAT $r = 2.1$	13.2	100	286	2.9
NAE3SAT $r = 3$	18	90	242	2.7
Clique	Chi=63, Peg=123	64	119	1.9

Chain Lengths at Matched Graph Sizes				
Logical Graph	Graph	C16 Chain	P16 Chain	Ratio
	Size	Length	Length	P16/C16
3D lattice (defects)	(all n)	4	2	.50
NAE3SAT $r = 2.1$	$n = 70$	12	5	.42
NAE3SAT $r = 3.0$	$n = 70$	15	7	.47
Clique	$n = 60$	16	7	.44

Figure 3: The graph at top shows the largest graph size for each chain length, when cliques are minor-embedded onto 2000Q C16 (blue) and Advantage P16 (orange) hardware graphs. The dotted gray lines show the maximum difference in graph sizes for equal chain lengths, and in chain lengths for equal graph sizes. The Advantage QPU can hold cliques that are about twice as large, using chains that are about half as long as those embedded on the 2000Q QPU. The middle table shows maximum embeddable graph sizes for a variety of other input types: mean of ratios in the table is 2.9. The bottom table shows typical chain lengths for other input graph types, matched by size; the mean ratio of chain lengths in the table is .46.

Embedded three-dimensional lattices (described in Section 3.2) contain defects (missing nodes) due to imperfect qubit yield: the largest such lattice that can fit on a C16 is $8 \times 8 \times 8$ and the largest that fits on a P16 is $15 \times 15 \times 12$. Not-all-equal 3-Satisfiability problems (also described in Section 3.2) are random boolean expressions containing v variables and c triplet clauses, with $\rho = 2.1$ and $\rho = 3$ variables per clause; the degrees shown in the table are means for random inputs of each type. Cliques are fully-connected graphs with degree equal to $n-1$ for problem size n. The mean ratio of embeddable graph sizes in this table is 2.9.

The bottom table compares typical chain lengths for representative graph types, when input sizes are matched to fit on both hardware graphs. The mean ratio of chain lengths in this table is .46.

These results are consistent with others not shown in the table: the maximum-size input that can be embedded onto an Advantage QPU is typically between 2 and 4 times bigger than similarly-structured inputs embeddable on a 2000Q QPU, around 3 times larger on average. Embedded chains are on average less than half as long on the Advantage QPU as on the 2000Q QPU.

Note that these numbers were obtain from empirical tests using specific QPU hardware graphs with imperfect yield. The clique embeddings were obtained using the polynomial-time `find_clique` utility, which finds clique embeddings containing chains of uniform length. The 3D lattice embeddings use a custom embedder that also returns uniform chain lengths. Minor-embeddings for random NAE3SAT inputs were found using the *minorminer* heuristic available in the Ocean toolkit, which finds embeddings with chains of uneven length. The tables show average degree and maximum chain lengths for NAE3SAT embeddings. See [5, 6] to learn more about D-Wave embedding tools.

2.3 Shorter Chains are Stronger Chains

In every minor-embedding, the weights on couplers that connect pairs of qubits within the same chain must be set to some large negative value J_{chain}, so that (ideally) chained qubits will have matching values in the output. A *broken chain* contains qubit values that disagree; a *broken solution* contains at least one broken chain. Broken chains are problematic because the qubits disagree as to what value should be assigned to their corresponding node in the original logical problem. Broken solutions may either be discarded from the solution sample or repaired using a post-processing utility available in Ocean, as described in Section 3.1.

The value of J_{chain} is called the *chain strength*. The optimal choice of chain strength for any given input lies in a sweet spot between two hazards, as outlined below.

Suppose the weights (h, J) in the logical graph all lie within some range $[-x \ldots + x]$. If the magnitude of J_{chain} is set relatively small with respect to x (for example, $\text{abs}(J_{chain}) \leq x$), then the objective function for the embedded input (h, J, J_{chain}) may introduce new optimal-but-broken solutions that are invalid in the logical problem space.

On the other hand, when the embedded problem (h, J, J_{chain}) is presented to the QPU, it is rescaled to a fixed energy range according to its largest-magnitude weight. For example, suppose $J_{chain} = -8x$. Scaling to an energy range of $[-1 \ldots + 1]$, would mean that J_{chain} maps to -1 and the remaining weights (i.e. all of (h, J)) map to $[-1/8 \ldots + 1/8]$. D-Wave

Chain Length per Problem Size				
Problem size n:	30	40	50	60
2000Q C16	9	11	14	17
Advantage P16	4	6	7	9

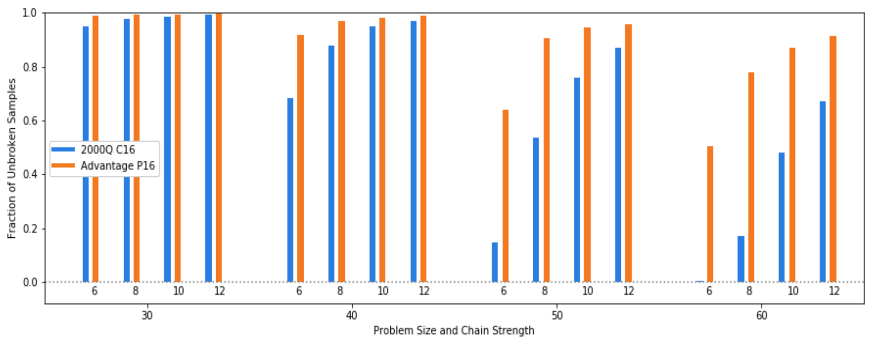


Figure 4: The table (top) compares chain lengths for a 2000Q and an Advantage QPU, in clique embeddings of various sizes n. The graph (bottom) shows the percentage of intact samples returned by the 2000Q (blue) and the Advantage (orange), when different chain strengths $J_{chain} = [2,4,6,8]$ are assigned. When matched by input size and chain strength, the Advantage QPU consistently returns a higher proportion of unbroken solutions per sample. On larger problems, Advantage can achieve equivalent proportions of unbroken samples using lower chain strengths.

QPUs offer an *extended_J_range* option, which uses the energy range $[-2\ldots+1]$: in this case J_{chain} maps to -2 and the problem weights map to $[-1/4\ldots+1/4]$ to maintain the original ratios. Either way, setting J_{chain} too large carries the danger of compressing the problem weights beyond the precision limits of the QPU control system. Too much compression makes it difficult for the quantum hardware to distinguish one small weight from another, which may degrade solution quality.

The higher connectivity of the Pegasus graph means that minor-embedded inputs can fit more compactly onto the physical graph, with shorter chains needed to represent the original nodes. Since shorter chains are less likely to break, the Advantage QPU can tolerate lower chain strengths, achieving better problem fidelity while maintaining comparable percentages of broken solutions.

Figure 4 illustrates this point using random clique inputs of size $n = [30, 40, 50, 60]$, with $h_i = 0$ and random edge weights $J_{ij} \in \{\pm 1\}$.

The table at the top of the figure shows chain lengths for each QPU at these problem sizes, taken from Figure 3. The graph below shows the effect of chain strength on the proportion P of unbroken solutions found in output samples from the 2000Q and Advantage QPUs.

For each n we generated 10 random inputs and embedded them with the `find_clique_embedding` utility, with four different chain strengths $J_{chain} = 6, 8, 10, 12$. The anneal time was set to 200μs and the `extended_j_range` feature was enabled. The vertical bars show the median values of P observed over all inputs, in samples of 1000 outputs, using a C16-based 2000Q QPU (blue) and a P16-based Advantage QPU (orange).

In every case, when QPUs are matched by problem size and chain strength, the value of P is higher for Advantage than for the 2000Q QPU. At $n = 30$, this difference is small:

both processors can return nearly perfect samples with$P \geq .95$ for any J_{chain}. However, at $n = 60$, the value of P depends significantly on the value of J_{chain}.

The Figure shows that not only does Advantage outperform 2000Q when chain strengths are matched, Advantage can achieve higher values of P using lower values of J_{chain}. For example, at $n = 60$, chain strength 6 suffices to ensure that Advantage meets the threshold $P \geq 0.5$, whereas chain strength 10 is needed for the 2000Q to achieve the same result. At all values of n, chain strength S on the Advantage QPU achieves equal or higher values of P than chain strength $S + 2$ on the 2000Q system.

The next section shows how this ability to support stronger chains with smaller chain strengths translates into better-quality solutions from the Advantage QPU.

3 Performance Comparison

This section considers performance of the 2000Q and Advantage quantum systems using three classes of logical inputs, all of which require minor embedding.

Our tests take a user-centric approach to performance analysis by making use of tools and utilities available in the Ocean developer's toolkit, as described in Section 3.1. Section 3.2 evaluates performance of the Advantage and 2000Q quantum systems on the clique problems from Section 2, as well as two additional problem categories known as NAE3SAT and 3DLattice.

3.1 Hardware and Software Better Together

The open-source Ocean developer's toolkit [7, 8] is part of a broad effort by D-Wave developers and engineers to make quantum-based problem-solving accessible to the computing public, by leveling out the learning curve associated with quantum computation. Furthermore, growing experience with Ocean tools suggests that from a performance point of view, hybrid methods combining the best features of quantum and classical computation can often produce better results than either approach used alone.

For example, a fast and simple post-processing utility known as *Majority Voting* (MV) can boost the quality of raw QPU results by repairing broken chains in output solutions. This utility, which is invoked by default in some QPU-based Ocean solvers, assigns a value to each logical node based on a "vote" of the qubits in its corresponding chain (breaking ties at random). Thus, the MV postprocessor converts all broken solutions into intact solutions, typically in just a few milliseconds of additional computation time. (Ocean provides other chain-repair strategies not considered here; see [9] for details.)

Chain repair is especially valuable in use cases that require quick response from the quantum system, in which case exploratory work to find optimal chain strengths is not an option. To demonstrate this point, we generate ten random clique inputs at each problem size $n = [5, 10, \ldots, 120]$, and additionally at the largest sizes embeddable on each QPU, $n = 64$ and $n = 124$. We set $J_{chain} = 8$ for all n, which is too low for problem sizes $n > 50$, as indicated in Figure 7.

For each input we request a sample of 1000 solutions, using anneal time 200μs, with the

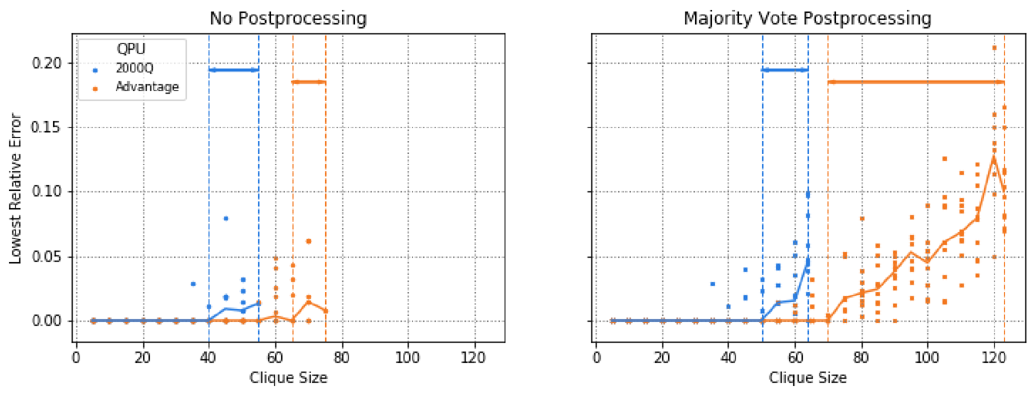


Figure 5: The graphs compare solution quality found by the 2000Q (blue) and Advantage (orange) QPUs when chain strengths are set to $J_{chain} = 8$, too low for large n. In the left panel, broken solutions are discarded; in the right panel, broken solutions are repaired using Majority Vote postprocessing. The points show the lowest relative error L, for ten trials at each n, and the lines connect data medians. A value of $L = 0$ indicates the reference solution was found in the solution sample; a value of $L > 0$ shows the scaled distance to the putative ground state. Missing points and lines indicate $L =$ NA: either the sample contains no intact solutions (left panel), or the problem is too big to fit on the QPU (right panel). For each QPU, a pair of vertical dotted lines indicates the transition points from $L = 0$ to $L > 0$, and from $L > 0$ to $L =$ NA.

`extended_j_range` option turned on. We measure solution quality with respect to a *reference energy* E_{ref}, corresponding to a putative optimal solution for each input. That is, since it is computationally infeasible to compute certifiably optimal solutions at the largest problem sizes tested, we instead employ a reliable classical heuristic running for a long time to find putative optimal solutions.

The solution quality metric is *lowest relative error*, denoted L. For each input we take 1000 solution samples from the QPU and either discard (left panel) or repair (right panel) the broken solutions. The lowest relative error is the scaled difference between the reference energy and the lowest energy found in the resulting sample:

$$L = \frac{(E_{ref} - E_{low})}{E_{ref}}.$$

This metric can be interpreted as follows:

- At small n we observe $L = 0$, indicating that the QPU can find at least one reference solution in the sample. With 1000 solutions this corresponds to achieving a threshold success probability of $\pi > 0.001$.
- At larger n we observe $L > 0$, in which case L shows the scaled distance between the sample minimum and the reference solution.
- At even larger n we observe $L =$ NA, that is, the statistic could not be computed because no viable results were returned. This happens for one of two reasons: (1) the QPU returns no solutions because the input is too large to fit; or (2) the solution sample from the QPU contained no intact solutions and all were discarded.

Figure 5 shows the results of setting the chain strength parameter too low, with and without MV postprocessing. The left panel shows solution quality when broken solutions are discarded from the sample, and the right panel shows solution quality when MV postprocessing is used to repair all broken solutions.

In both panels, the points show values of L returned by the 2000Q (blue) and Advantage (orange) QPUs for each input, and the lines connect medians over n. For each QPU, a pair of vertical dotted bars show a region bounded by two transition points in n: from $L = 0$ to $L > 0$ (left bar) and from $L > 0$ to $L = \text{NA}$ (right bar). Here are some observations.

- Comparing left bars, we see that Advantage can find optimal solutions at larger problem sizes than the 2000Q QPU: the transition point is 44% higher (from $n = 45$ to $n = 65$) when considering raw QPU outputs and 40% higher ($n = 50$ to $n = 70$) after postprocessing.
- Comparison of right bars shows that Advantage finds intact solutions at higher n. On the left panel, Advantage goes to 36% higher n than the 2000Q QPU ($n = 55$ vs. $n = 75$). On the right panel, we see that Advantage is able to realize the full potential of higher qubit counts and greater connectivity, finding solutions to problems of size up to $n = 124$, nearly twice as big as $n = 64$ for the 2000Q QPU.
- In both panels, the region marked by orange bars is strictly to the right of the region marked by blue bars. In this test the Advantage system can find optimal solutions at problem sizes for which the 2000Q fails to find any viable solutions at all.

MV postprocessing improves performance of both QPUs in this case and in general; for this reason the performance tests in the next section incorporate this Ocean feature.

3.2 Performance Analysis

The capability to support stronger chains and lower chain strengths, means that the Advantage QPU can sometimes return better-quality solutions than the 2000Q QPU when both are given the same computational resources. It also means that the Advantage QPU can sometimes find same-quality solutions faster.

This section describes three small case studies comparing performance of the two QPUs under a various performance metrics. In all three cases the Advantage QPU achieved equal or superior performance to the 2000Q QPU.

Random Cliques In this section we compare performance of an Advantage and 2000Q system using the clique inputs described in the previous section. We generate 10 random cliques at each problem size $n = [10, \ldots, 60]$. Anneal time is set to 200μs, $R = 1000$ output samples are read, and the `extended_j_range` option is turned on. The majority voting (MV) postprocessing tool was used throughout these tests.

A small exploratory study was performed to identify good values for chain strength J_{chain}, for each QPU and each input size n. The study considered integer values of J_{chain} as well as a functional form $J_{chain} = c\sqrt{n}$ found by a regression analysis on each QPU.

Tuned Chain Strength						
n :	10	20	30	40	50	60
2000Q	4	6	7.3	8.4	12	10
Advantage	4	6	6	8	8	10

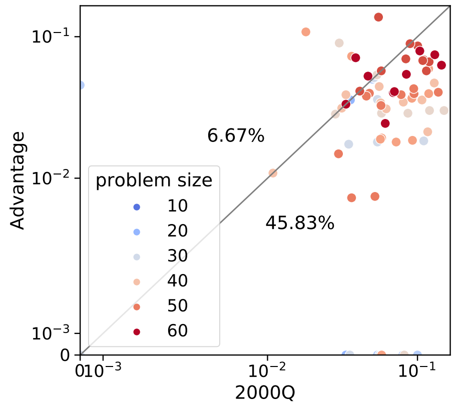


Figure 6: The table shows the best chain strength J_{chain} for each QPU and problem size, to minimize the median scaled error M. The graph shows an input-by-input comparison of M for the Advantage (y-axis) and the 2000Q (x-axis) QPUs using tuned chain strengths. Points are color-coded by probem size n. The Advantage QPU found better values of M the 2000Q QPU, on about 45.83% of inputs, especially the largest inputs, whereas the 2000Q found better results on just 6.67% of inputs. Percentages do not sum to 100 because many outcomes were ties.

Here, J_{chain} was selected to minimize the *median relative error* metric M, similar to the lowest relative error L from the previous section, except computed on the sample median instead of the sample minimum. Although L is arguably more interesting to practitioners, we use M instead because both QPUs frequently find optimal solutions on these problems, with the result that $L = 0$ in nearly all cases; for this reason L cannot be used to optimize parameters, nor to distinguish performance of the two quantum processors.

The results are shown in Figure 6. The table at top shows the best chain strength J_{chain} found in the exploratory study. The one case of non-monotonicity in the table appears to be due to experimental noise: we note that both quantum processors can tolerate some imprecision in selection of optimal J_{chain}, since values within, say, ± 2 of those shown in the table have nearly imperceptible effect on solution quality as measured by M.

The bottom graph shows an input-by-input comparison of M obtained by the Advantage (y-axis) and 2000Q (x-axis) systems. Points below the diagonal indicate cases where the Advantage system found better solutions than the 2000Q system, and points above the line indicate cases where the 2000Q outperformed Advantage. The percentages shown do not sum to 100 because many outcomes were ties, especially on small inputs where $M = 0$ for both.

In the cases that are not ties, the Advantage system significantly outperforms the 2000Q system. In this test the win percentage for the Advantage QPU was 45.83 percent, over six times higher than the win percentage of 6.67 percent for the 2000Q QPU.

Finally, Figure 7 compares performance of the 2000Q and the Advantage systems using the

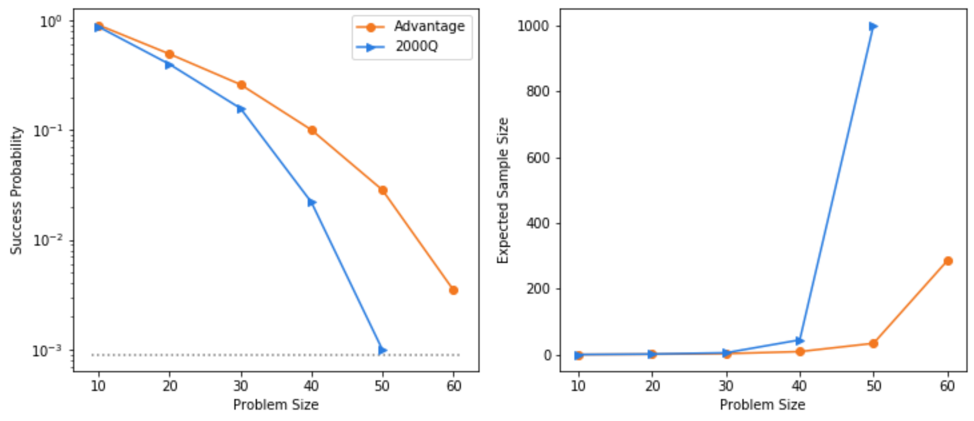


Figure 7: The graph shows median success probabilities Advantage vs. 2000Q QPUs, on random cliques. In these tests using 1000 reads, a success probability below 0.001 (i.e. below the gray dotted line), as observed for 2000Q at $n = 60$, is recorded as zero and not shown on this logarithmic scale. Throughout this range of problem sizes, the Advantage QPU achieves higher success probabilities than the 2000Q QPU. The right panel shows $1/\pi$, corresponding to the expected sample size needed to observe a reference solution in the output sample. At $n = 50$, the expected sample size is 30 times smaller for Advantage than for the 2000Q QPU.

success probability metric denoted π, which is estimated by the proportion of solutions in each sample having energies matching the reference energy.

The left panel shows median values of π for each N and each QPU, using tuned chain strengths and MV postprocessing. At $n = 60$ the 2000Q system found no reference solutions in 1000 samples: this indicates $\pi < 0.001$ (below the grey dotted line), which is not plotted. At $n = 50$ the Advantage system achieves median success probability near $\pi = .03$, which is about 30 times higher than $\pi = .001$ from the 2000Q system.

We note that when chain strength is optimally tuned, the 2000Q can find optimal solutions to these inputs for $n \leq 50$, whereas Figure 7 (right panel) indicates that the Advantage QPU can find optimal solutions up to $n = 70$, even when chain strength is not optimally tuned. This represents at least a 40 percent increase in the size of inputs that can be solved to optimality.

The right panel shows $R = 1/\pi$, corresponding to the expected number of solution samples needed to observe a reference solution. Expected sample size R corresponds roughly to the computation time needed to find reference solutions, although times at small values of R such as these tend to be dominated by chip I/O overhead. In this context, a 30-fold speedup in R maps to over a 6-fold speedup in wall clock time, from about .42 seconds to .063 seconds.

NAE3SAT Inputs Next we compare QPU performance on inputs for the Not-All-Equal 3-Satisfiability (NAE3SAT) problem. A random NAE3SAT input is a boolean expression with n variables $v = \{v_1, \ldots v_n\}$ organized in m clauses: each clause contains three randomly-selected variables or their negations. A clause is *satisfied* (i.e. has the value True) if the

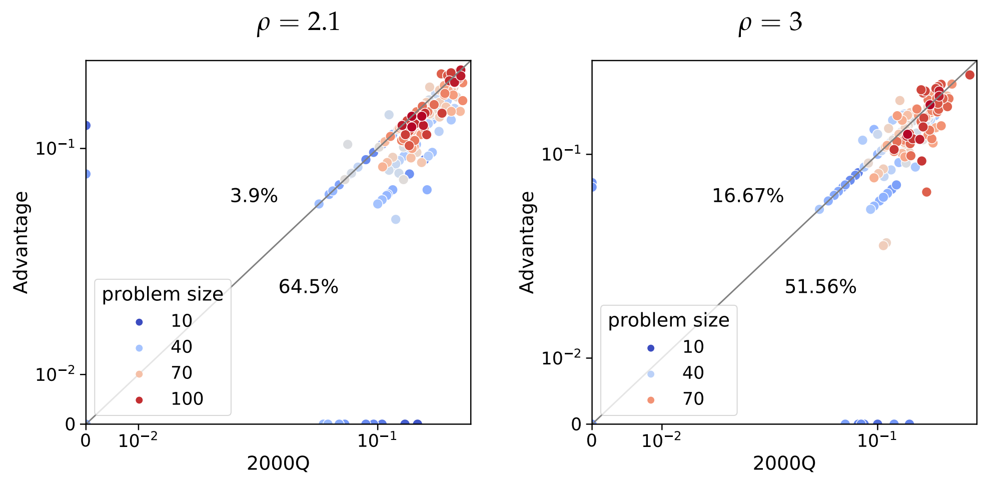


Figure 8: The left and right panels show results for $\rho = 2.1$ and $\rho = 3$, respectively. Maximum input size is $n = 100$ in the left panel and $n = 90$ in the right panel. The graphs present input-by-input comparisons of median relative error M, for the Advantage (y-axis) and 2000Q (x-axis) QPUs. In each panel, points below the diagonal line indicate cases where Advantage returned superior solutions. The percentages do not sum to 100 because many ties were observed.

values of the three elements are not all equal. Clauses are joined by disjunctions (logical ”and”), so that the whole formula is satisfied only when every clause is True.

The combinatorial hardness of random formulas constructed this way depends on the clause-to-variable ratio $\rho = m/n$. Our tests use random NAE3SAT inputs generated at $\rho = 2.1$ (interesting for testing performance at finding optimal solutions) and at $\rho = 3$ (interesting for testing performance at finding approximate solutions). The boolean expressions thusly generated are translated into QUBO logical inputs using standard techniques. The logical graphs resulting from problems generated at $\rho = 2.1$ and $\rho = 3$ have average node degrees 13.2 and 18, respectively.

These inputs were embedded using the heuristic `find_embedding` tool available in Ocean. This embedder differs from the polynomial-time `find_clique_embedding` tool used for clique inputs in several ways, for example: embeddings may contain chains of varying length; different runs will produce different embeddings. See [5, 6] for more about embedders in Ocean.

In these tests we generated five random inputs at each problem size $n \in [10, 12, \ldots, n_\rho]$ where $n_{2.1} = 100$ and $h_3 = 90$. For these types of inputs, the minimum and maximum chain lengths in a given embedding tend to differ by a factor of about two. Embeddings on the Advantage QPU tend to have mean and maximum chain lengths about half as long as embeddings on the 2000Q QPU.

Putative optimal solution energies were obtained using reliable heuristics applied to the original problem formulations. For each input we took 1000 samples from each QPU. Anneal time was set to 200μs; chain strength was set to $J_{chain} = -3$ (based on an exploratory study); and the MV postprocessing option was selected.

The results appear in the two graphs of Figure 8, which show input-by-input comparisons

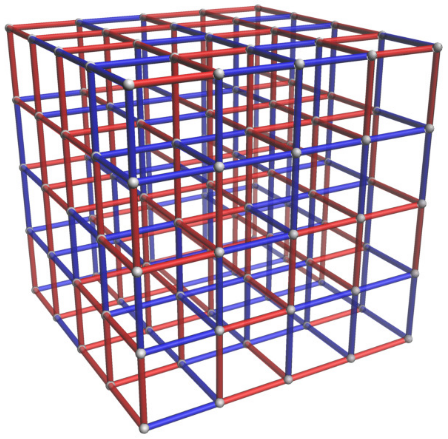


Figure 9: A $5 \times 5 \times 5$ 3D spin glass, from [1]. The edge weights are assigned $J_{ij} = +1$ (red) or $J_{ij} = -1$ (blue) uniformly at random.
.

of Advantage (orange) and 2000Q (blue) performance, for input sets $\rho = 2.1$ (left panel) and $\rho = 3$ (right panel).

Solution quality is measured by the median relative error metric M described in the previous section. Points below the diagonal line correspond to cases where the Advantage processer returned strictly better solutions than the 2000Q processor. The win percentages shown in the panels do not sum to 100 because the two processors tied in many cases.

As before, the Advantage system finds better solutions than the 2000Q systems on more inputs. For $\rho = 2.1$ the 64.5% win percentage is about 16 times higher, and for $\rho = 3$ it is about 3 times higher.

As with clique inputs, the lowest relative error metric L cannot distinguish QPU performance because most outcomes are ties. On $\rho = 2.1$ inputs, both QPUs find optimal solutions ($L = 0$) up to $n = 54$, and L values are statistically indistinguishable at higher n. At $n_{2.1} = 100$ both return $L = 0.25$, indicating that solution energies are within 97.5 percent of the reference energy.

On $\rho = 3$ inputs, the Advantage system shows a small but consistent improvement over the 2000Q system, achieving $L = 0$ up to $n = 54$ versus $n = 52$. Averaging over the range $54 \leq n \leq 90$, the Advantage system returns $L = .028$ (within 97.2 percent of the reference energy) compared to $L = .032$ for the 2000Q system.

Recall from Figure 3 that the Advantage system reads and solves problems up to size $n = 242$, which the 2000Q system ($n \leq 90$) cannot attempt.

3D Lattices We finish the section with a summary of a performance study described by King [1]. This work compares performance of an Advantage QPU and a (low noise) 2000Q QPU at the task of sampling ground states in 3D lattices. Several recent papers describe projects in scientific research that use D-Wave 2000Q quantum processors to simulate quantum processes and to sample low-energy states in regular lattice structures such as these ([**lattice3**, 10, 11]).

A 3D lattice problem consists of a logical grid of size $L \times L \times L$, as shown in Figure 9, with $h_i = 0$, and with edge weights $J_{ij} \in \{\pm 1\}$ (red and blue in the figure) uniformly at random to lattice edges.

One test described in [1] generates 100 random instances for each $L \in [5, 6, \ldots, 10]$, and embeds them onto a 2000Q C16 and an Advantage P16 using a custom embedder that exploits the regular structures of lattices and hardware graphs. Lattices of size $L > 8$ cannot fit onto the C16 graph. Embeddings in the Pegasus graph have chain length 2, whereas embeddings in the Chimera graph have chain length 4.

Note that in these tests the logical lattices contain some number of *defects* (missing nodes), since the hardware graphs in both QPUs contain inoperable qubits:[1] if some node in the lattice cannot be represented on one of the two hardware graphs, it is removed from the logical problem. Thus, both QPUs are given identical lattice structures to solve. Chain strength was set to $J_{chain} = -2$, near the optimal value for both. The majority vote postprocessor was turned on.

A pilot study was used to find putative ground states. Then, for each input, a batch of 500 solutions was read for each anneal time $t_a \in [2, 4, 8, \ldots, 256]\mu$s, iterating for at least 100 batches, or until the putative ground state was found at least 50 times. The ground state probability $p_{GS}(t_a)$ was estimated as the proportion of ground state solutions found in the full sample set, for each anneal time t_a.

The performance metric used here is time to solution (TTS), based on the total number of reads needed to ensure that the ground state is observed with confidence least 99 percent. For each input, TTS is the minimum value of TTS(t_a) (over all t_a in the test), defined as follows:

$$\text{TTS}(t_a) \quad = \quad t_a \frac{\log(.01)}{\log(1 - p_{GS}(t_a))}.$$

The results are shown in Figure 10. The error bars at each L indicate percentiles 10, 25, 75, 90 over 100 instances; the lines join median points for each distribution. At $L = 8$, the largest problem that fits on the 2000Q, median TTS for the Advantage processor is about 10 times lower than TTS for the 2000Q system; indeed the median TTS for the 2000Q QPU is near the highest 90th percentile TTS for the Advantage QPU. The lower slope (better scaling) of the Advantage processor on these inputs is also notable.

4 Conclusions

This report describes features of the D-Wave Advantage quantum computer in comparison to its predecessor, the 2000Q quantum computer, focusing on the quantum processing units (QPUs) at the core of each system, and on features of the software stack in the Ocean developer's kit.

Because of higher qubit counts and greater connectivity, the Advantage QPU can hold inputs that are typically between 2 and 4 times larger than those that fit on the 2000Q

[1]The 2000Q QPU in this test has 2041 operable qubits out of 2045, while the Advantage QPU has 5510 operable qubits out of 5750. There are no defects in either QPU for lattices of size $L = 5, 6$.

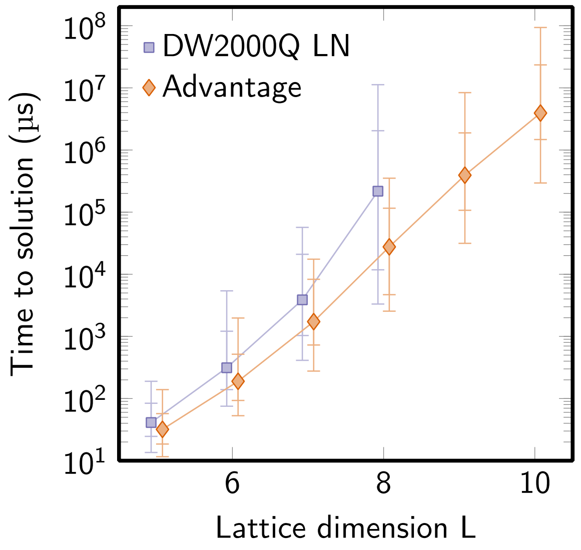


Figure 10: Time to Solution (TTS), from [1]. Beyond the capability of solving larger problems, the Advantage processor outperforms the 2000Q processor both in terms of absolute computation times and in better scaling on 3D spin glasses.

QPU. Chain lengths on Advantage QPUs are generally half as long as those on the 2000Q QPU.

The performance comparison in Section 3.2 shows that, beyond the ability to solve larger problems, the Advantage system can sometimes find better-quality solutions than the 2000Q system when both are allowed equivalent computational resources (cliques and NAE3SAT problems) and can find equivalent-quality (optimal) solutions faster (cliques and 3D lattices). We attribute this performance improvement to the new Pegasus graph topology and its support for shorter, stronger chains in minor-embedded inputs.

The tests described here focus on performance as would be experienced in applications practice, by considering logical input classes that must be embedded onto the respective QPUs. Furthermore, our tests consider the performance of the QPUs when used in combination with tools found in the Ocean developer's toolkit. This hybrid approach to performance analysis gives a realistic view of what would be observed in applications practice.

The preliminary results presented here are only the beginning: we look forward to more extensive performance studies of the Advantage quantum processing system, and to further demonstrations of superior performance in comparison to both quantum and classical alternatives.

References

[1] A. King, "Performance benefits of increased qubit connectivity in quantum annealing 3-dimensional spin glasses," `dwavesys.com/resources/publications` (to appear) (2020).

[2] C. McGeoch et al., *Hybrid Solver Service + Advantage: Technology Update*, `dwavesys.com/resources/publications` (2020).

[3] K. Boothby et al., *Next-Generation Topology of D-Wave Quantum Processors*, `dwavesys.com/resources/publications` (2019).

[4] A. Lucas, “Ising formulations of many NP problems,” Frontiers in Physics **2** (2014).

[5] “D-Wave Ocean Software Documentation, keyword: minor embedding,” `http://docs.ocean.dwavesys.com/en/stable/index.html` (2020).

[6] “D-Wave Ocean Software Documentation, keyword: find clique embedding,” `http://docs.ocean.dwavesys.com/en/stable/index.html` (2020).

[7] “D-Wave’s Ocean Software,” `http://ocean.dwavesys.com` (2020).

[8] “D-Wave Ocean Software Documentation,” `http://docs.ocean.dwavesys.com/en/stable/index.html` (2020).

[9] “D-Wave Ocean Software Docuentation, keyword: post processing,” `http://docs.ocean.dwavesys.com/en/stable/index.html` (2020).

[10] R Harris, “Phase transitions in a programmable quantum spin glass simulator,” Science **316**, 162–165 (2018).

[11] A. King et al., “Observation of topological phenomenon in a programmable lattice of 1,800 qubits,” Nature **560**, 456–460 (2018).