D-Wave
The Quantum Computing Company™

# The Advantage System: Performance Update

## TECHNICAL REPORT

Catherine McGeoch and Pau Farré

2021-10-01

## Overview

This report presents a high-level overview of the Advantage™ quantum computing system. Advantage quantum processing units (QPUs) can hold application inputs that are almost three times larger, on average, than those that fit on previous-generation D-Wave 2000Q™ QPUs. Beyond the capacity to read bigger problems, comparison of an Advantage performance update (herein referred to as Advantage 4.1) to a D-Wave 2000Q solver demonstrates that Advantage systems can deliver significantly better performance on application-relevant inputs. The Advantage 4.1 solver also outperforms the original Advantage 1.1 solver launched in 2020. We attribute this improved performance to innovations in chip design as well as new fabrication materials and processes.

14-1054A-A
D-Wave Technical Report Series

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**

# Notice and Disclaimer

# Summary

This report presents a high-level overview of D-Wave's Advantage™ quantum computing system. This is an updated version of a report published with the launch of the Advantage product line in September 2020; it contains new performance data for an Advantage performance update made publicly available in October 2021.

Technological advances in the design of the quantum processing unit (QPU) at its core make Advantage processors by far the largest and most powerful quantum computers in existence today:

- Every Advantage processor contains at least 5,000 qubits, about 2.5 times more than found in a D-Wave 2000Q processor. 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 Advantage QPUs. An Advantage QPU can hold inputs between 2 and 5 times larger than similarly-structured inputs that fit on a D-Wave 2000Q QPU, 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 D-Wave 2000Q QPUs.

Beyond the capacity to read and solve larger inputs, Advantage processors can find better-quality solutions than 2000Q processors. This report compares the Advantage performance update (herein referred to as Advantage 4.1) to a previous-generation D-Wave 2000Q solver (referred to as 2000Q), in terms of their physical properties and quality of solutions returned, in tests using application-relevant inputs. Key findings are summarized below.

- On clique problems, the Advantage 4.1 solver found optimal solutions on inputs that were 62 percent larger than the largest optimally-solved inputs on the 2000Q solver. In cases where both found optimal solutions, Advantage 4.1 found them faster, up to 54 times faster in terms of pure anneal time, or about 14 times faster in wall-clock time. Overall, the Advantage 4.1 system found better solutions on 53 percent of inputs, over 16 times more often than the 2000Q system, which won on just 3 percent of inputs (the remaining 44 percent of cases were ties).
- On inputs for the NAE3SAT problem, Advantage 4.1 again outperformed the 2000Q. In one test, the Advantage 4.1 QPU found better solutions on 81 percent of inputs, compared to just 0.9 percent of inputs for the 2000Q QPU (about 90 times more often).
- On inputs for 3D lattice problems, the Advantage 4.1 QPU found optimal solutions up to 10 times faster than the 2000Q QPU, considering pure anneal times.
- Comparison data for the original Advantage system launched in September 2020 appears in Appendix A: The Advantage 4.1 solver outperforms its predecessor (Advantage 1.1) in nearly every metric considered in this report. We attribute this improved

performance to innovations in chip design as well as new fabrication materials and processes.

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 the Advantage™ quantum system. The quantum processing unit (QPU) at its core represents a new pinnacle of design innovation 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.

This is an updated version of a technical report that was published concurrently with the launch of the Advantage generation of quantum processors in September 2020. This version contains new performance data from an Advantage performance update made publicly available in October 2021.

Throughout this report, a reference to "the Advantage QPU" or "the D-Wave 2000Q system" applies generally to solvers in those respective product lines. Sections describing empirical results refer to specific solvers used in those tests: the Advantage performance update is herein referred to as Advantage 4.1; the original Advantage processor launched in 2020 is Advantage 1.1; and a specific D-Wave 2000Q LN (lower noise) solver released in 2019 is referred to as the 2000Q.[1]

Section 2 gives an overview of the main design features of the Advantage generation of quantum processors, in comparison to its predecessor the D-Wave 2000Q system. Section 3 presents a brief performance comparison of two specific solvers known as Advantage 4.1 and 2000Q. (Comparisons to the Advantage 1.1 solver may be found in Appendix A.) Section 4 contains some final remarks.

The key results in this report are summarized below.

- Every Advantage processor contain at least 5,000 qubits, about 2.5 times more than found in D-Wave 2000Q systems. 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 systems.
- 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 Advantage are approximately 2.8 times larger on average than similarly-structured inputs that fit on the 2000Q system. More generally, Advantage QPUs can typically hold inputs that are between 2 and 5 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 Advantage QPUs can return better-quality solutions than 2000Q QPUs when both are given the same computational resources. It also means that Advantage can return

[1] Their official online names are `Advantage_system4.1`, `Advantage_system1.1`, and `DW_2000Q_6`. A "solver" is a specific combination of a physical machine and a control system. The latter determines, for example, which system parameters and features (such as range of anneal times) are supported.

same-quality solutions faster than the 2000Q QPU. Section 3 describes three case studies that demonstrate superior performance from an Advantage 4.1 solver:

  - In a study using clique problems, the Advantage 4.1 solver found optimal solutions on inputs that were 62 percent larger than those solved to optimality on the 2000Q solver (from $n = 40$ to $n = 65$). In cases where both found optimal solutions, Advantage 4.1 was faster, up to 54 times faster in terms of pure anneal time and about 14 times faster in terms of wall clock time (which includes system and I/O overhead). On average, Advantage 4.1 found better solutions on 52.5 percent of inputs, about 16 times more often than the 2000Q solver, which won on just 3.3 percent of inputs (the remaining 44 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, Advantage 4.1 shows superior performance. In one comparison of solution quality, the Advantage 4.1 solver found better solutions in 81.17 percent of inputs, compared to just 0.9 percent of cases for the 2000Q solver (about 90 times more often).
  - On inputs for 3D lattice problems (described more fully in [1]), the Advantage 4.1 solver found optimal solutions about 10 times faster than the 2000Q solver, 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 Advantage and D-Wave 2000Q 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, India and Singapore) 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 basis. In September 2020, the suite of hybrid solvers was 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. As of October 2021, HSS contains three hybrid solvers: the binary quadratic model (BQM) solver for unconstrained binary quadratic problems; the discrete quadratic model (DQM) solver for unconstrained quadratic problems defined on discrete variables; and the constrained quadratic solver (CQM) which supports direct representation of constraints for quadratic problems defined on binary and integer variables. Two D-Wave technical reports [2, 3] describe HSS features and performance.

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

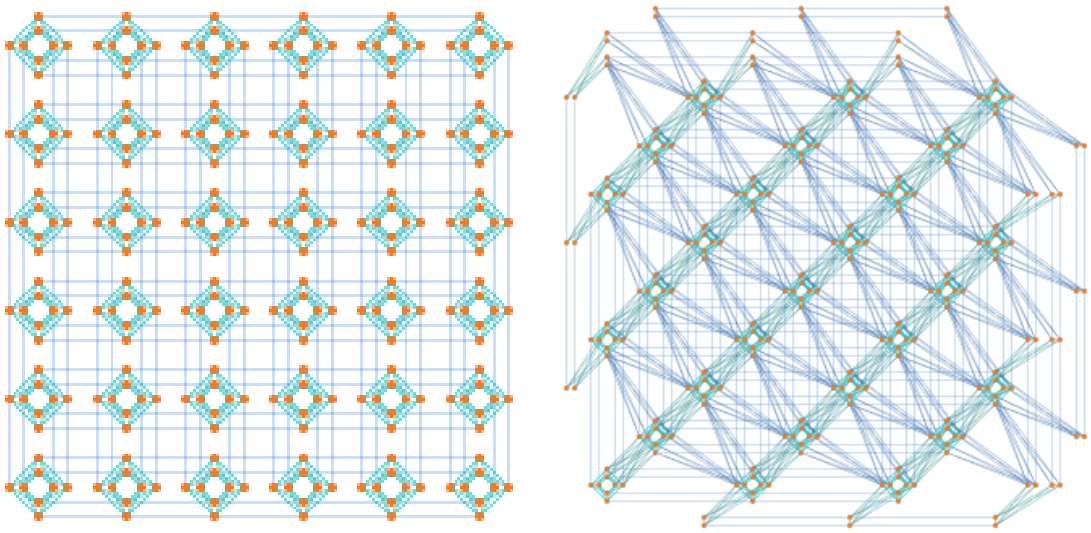


**Figure 1:** A C6 Chimera graph (left) with 36 unit cells containing 288 qubits. A P4 Pegasus graph (right) with 27 unit cells and several partial cells, containing 264 qubits. The comparatively rich connectivity structure of the P4 is clearly seen.

# 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 D-Wave 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 [4] for discussion of these and other properties.

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 QPU models in terms of typical component counts. The exact number of active qubits and couplers can vary across individual QPUs, 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 that is made available to the public.

The following sections illustrate the connection between higher-degree connectivity struc-

| | 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.

tures, which produce more compact embeddings with shorter chains, and QPU performance in terms of better solution quality and faster computation times.

Section 2.1 discusses general properties of the Advantage and D-Wave 2000Q processor generations. As mentioned in the introduction, Sections 2.2 and 2.3 present results of an empirical study performed on two specific solvers, here referred to as Advantage 4.1 and 2000Q. Appendix A contains results of similar tests run on the original Advantage 1.1 system launched in September 2020.

## 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 [5] 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 2 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.

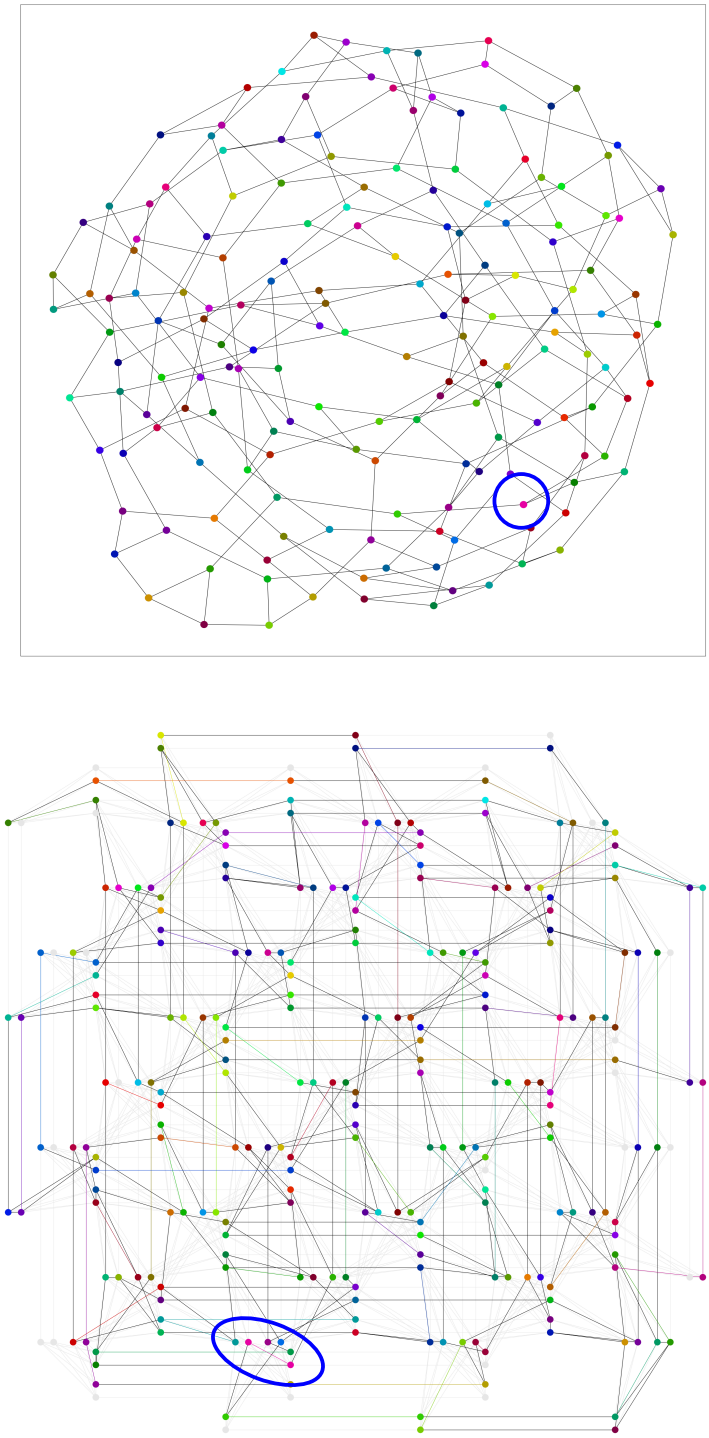


**Figure 2:** Minor embedding of a general QUBO graph (top) onto a P4 Pegasus graph (bottom). In the physical embedding, chain edges are colored to match their nodes, and logical edges are black. The blue circle and oval highlight a pink node that is mapped to a chain of two qubits during minor-embedding.

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 2 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.

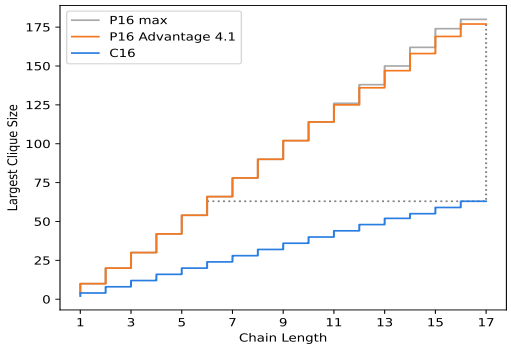
The graph at the top of Figure 3 compares clique embeddings obtained using the Advantage 4.1 P16 topology (orange) and a 2000Q C16 (blue). Each curve shows the largest clique size that can be embedded using a given chain length. With chains of length 17, the P16 can hold cliques of size up to $n = 177$, 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 almost three times as big, using chains that are less than half as long.

The middle table compares maximum embeddable graph sizes for some representative graph types, ordered 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, which are identical to the physical hardware graph and require no chains.

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 its 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 a clause-to-variable ratio of $\rho = 3$; shown in the table are mean degrees for random inputs. Cliques are fully-connected graphs with degree equal to $n - 1$ for problem size $n$. The bottom table compares typical chain lengths for these graphs, when input sizes are matched to fit on both hardware graphs.

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 5* 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 4.1 QPU as on the 2000Q QPU.

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 SDK), which finds embeddings with chains of uneven length. The tables show average degree and maximum chain lengths for NAE3SAT embeddings. See [6, 7] to learn more about D-Wave embedding tools.




| Maximum Graph Sizes | | | | |
|---|---|---|---|---|
| Logical Graph | Degree | C16 $n$ | P16 $n$ | Ratio P16/C16 |
| Native | Chi=6, Peg=15 | 2030 | 5627 | 2.8 |
| 3D lattice (w/defects) | 6 | 512 | 2687 | 5.2 |
| NAE3SAT $\rho = 3$ | 18 | 90 | 252 | 2.8 |
| Clique | Chi=63,Peg=176 | 64 | 177 | 2.8 |

| Chain Lengths at Matched Graph Sizes | | | | |
|---|---|---|---|---|
| Logical Graph | Graph Size | C16 Chain Length | P16 Chain Length | Ratio P16/C16 |
| 3D lattice (defects) | (all $n$) | 4 | 2 | 0.50 |
| NAE3SAT $\rho = 3$ | $n = 80$ | 33 | 11 | 0.33 |
| Clique | $n = 64$ | 17 | 7 | 0.41 |

**Figure 3:** The graph at top shows the largest input size for each chain length, when cliques are minor-embedded onto 2000Q C16 (blue) and Advantage 4.1 P16 (orange) hardware graphs. The gray step curve at top shows results for a perfect defect-free P16 graph, which the Advantage4.1 QPU very nearly achieves. The dotted gray lines illustrate the differences in graph sizes for equal chain lengths (177 vs. 64 for chain length 17), and in chain lengths for equal graph sizes (7 vs. 17 for $N = 64$). The Advantage 4.1 QPU can hold cliques that are almost three times larger, 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 3.4. The bottom table shows maximum chain lengths for these input graph types, matched by size.

## 2.3 Chain Length and Chain Strength

A logical input graph $G$ must be minor-embedded onto the hardware graph $H$ before being sent to the QPU. Minor-embedding maps logical nodes with high degree in $G$ into *chains* of lower-degree qubits in $H$; chained qubits are connected by couplers assigned a weight $J_{chain}$ known as the *chain strength*, which is normally set to a large-magnitude negative value to ensure that chained qubits all have the same value 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 logical node in the original problem. Broken solutions can either be discarded from the output sample or repaired using one of the postprocessing utilities available in Ocean, as described in Section 3.1.

As a general rule, embeddings with shorter chains are preferred over embeddings with longer chains simply because short chains have fewer opportunities to break. But chain strength is also important: suppose the logical input weights $(h, J)$ all lie in a range $[-x, x]$. The optimal choice of chain strength relative to $x$ lies in a sweet spot between two hazards, as outlined below.

- If $-J_{chain}$ (a positive value) is small compared to $x$, then the embedded input $(h, J, J_{chain})$ might contain spurious 'optimal' solutions that encode broken chains. Assuming qubit $q$ is incident on $c$ couplers, of which 2 are used as chains and the rest used as logical weights, it would be necessary to set $-J_{chain} > (c-2)x$ so the chain is strong enough to overcome the *torque* created by logical weights (in a worst-case scenario).
- On the other hand, setting $-J_{chain}$ much higher than $x$ creates a danger of compressing the problem weights in $[-x, x]$ (and therefore the gaps between distinct solution energies) beyond the precision limits of the quantum control system. Conceptually, every problem sent to the QPU is first scaled to an energy range $[-1, 1]$ relative to its largest-magnitude weight (i.e. $J_{chain}$).[2] For example, let $d = c - 2$. If $-J_{chain} = d \cdot x$, then problem weights are scaled to $[-1/d, 1/d]$: if $d$ is too large, the QPU may have difficulty distinguishing one weight from another, leading to a deterioration in 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. However, qubits are incident on 15 couplers (excepting boundary qubits), so that the *chain ratio*, of logical couplers to chain couplers per qubit, is typically $13/2$. In a Chimera graph the chain ratio is only $4/2$, which greatly reduces the hazard of setting $J_{chain}$ too high. The ultimate question is whether benefits of shorter chains are enough to overcome the hazards of higher connectivity and greater torque.

Figure 4 shows the net effect of these opposing forces in a comparison of Advantage 4.1 and 2000Q processors. The inputs are random cliques 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 and chain ratios for each QPU: embeddings on Advantage 4.1 have about half the chain length, but almost three times the

[2] The example in the text is presented for simplicity: D-Wave QPUs offer an `extended_J_range` option that scales to the energy range $[-2, 1]$, effectively doubling the compressed scale of logical weights.

| | Chain Length | | | | Chain Ratio |
|---|---|---|---|---|---|
| Problem size $n$: | 30 | 40 | 50 | 60 | |
| 2000Q C16 | 9 | 11 | 14 | 17 | 4/2 |
| Advantage 4.1 P16 | 4 | 5 | 6 | 7 | 13/2 |

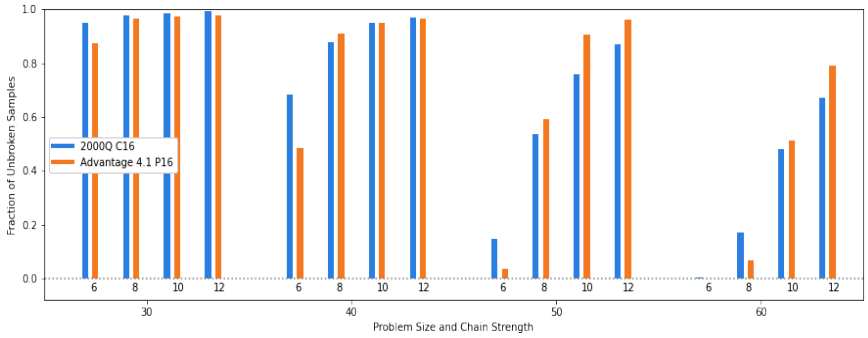



**Figure 4:** The table (top) compares chain lengths and chain ratios for the D-Wave 2000Q and Advantage 4.1 QPUs in clique embeddings of various sizes $n$. The rightmost column shows proportions of chain couplers to problem couplers per node on Chimera and Pegasus graphs. The graph (bottom) shows the percentage of intact samples returned by 2000Q (blue) and Advantage (orange), when different chain strengths $J_{chain} = [2, 4, 6, 8]$ are assigned.

chain ratio, as embeddings on 2000Q.

The graph below shows the effects of problem size and chain strength on the proportion $P$ of unbroken solutions in sampled outputs from the Advantage 4.1 (orange) and 2000Q (blue) QPUs. For each $n$ we generated 10 random inputs and embedded them with the `find_clique_embedding` utility, using four different chain strengths $J_{chain} = 6, 8, 10, 12$. The anneal time was set to 200$\mu$s 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.

The graph shows that across a range of problem sizes and chain strengths, the upgrade to Pegasus is beneficial: Advantage 4.1 returns values of $P$ that are competitive with and sometimes better than, those returned by 2000Q processor. (Note that relative performance in $P$ for very low chain strength 6 is likely due to the introduction of spurious optimal solutions as described above: Advantage 4.1 returns more broken solutions because they are optimal but spurious, created by setting chain strength too low.)

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 Advantage 4.1 and the 2000Q solvers, in terms of the quality of solutions they find. Results of similar tests on the Advantage 1.1 solver may be found in Appendix A.

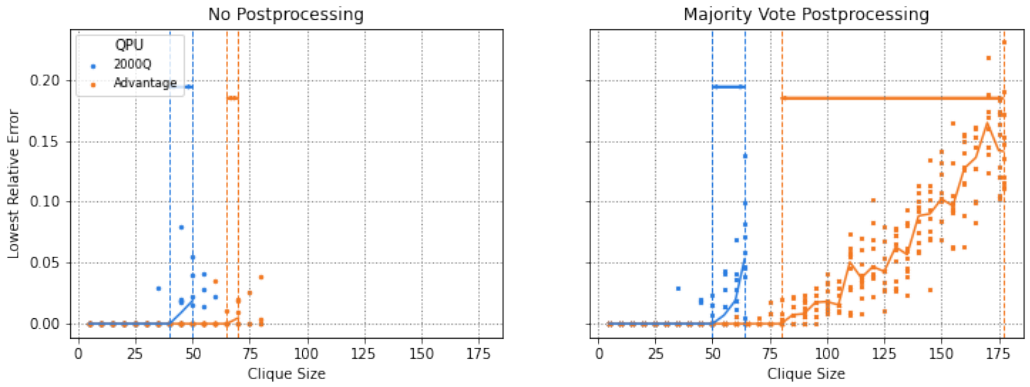



**Figure 5:** The graphs compare solution quality found by the 2000Q (blue) and Advantage 4.1 (orange) QPUs when chain strengths are set to $J_{chain} = 8$ (too low for largest $n$). In the left panel, broken solutions are discarded; in the right panel, broken solutions are repaired using Majority Vote 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 this 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.

Rather than evaluate raw hardware performance, we take a user-centric and system-centric approach by making use of tools and utilities available in the Ocean SDK, 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 SDK [8, 9] 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. Growing experience with Ocean tools suggests that 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 postprocessing 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, in just a few microseconds per output. (Ocean provides other chain-repair strategies not discussed here; see [10] 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 $n = 177$. We set $J_{chain} = 8$ for all $n$, which is too low for the largest problem sizes (see Figure 6), producing a too-high proportion of broken chains.

For each input we request a sample of 1000 solutions, using anneal time 200$\mu$s, with the `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:

$$L = \frac{\Delta(E_{ref}, E_{low})}{|E_{ref}|},$$

Where $\Delta(E_{ref}, E_{low})$ is the positive distance between the two energies, accounting for possible sign differences. 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 $L = \text{NA}$, 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 (unbroken) 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 4.1 (orange) QPUs for each input, and the lines connect medians over $n$. For each QPU a pair of horizontal bars above the data points show regions bounded by two transition points in $n$: from $L = 0$ to $L > 0$ (left bar endpoint) and from $L > 0$ to $L = \text{NA}$ (right bar endpoint). Here are some observations.

- Comparing left endpoints, we see that Advantage 4.1 can find optimal solutions at larger problem sizes than the 2000Q QPU: the transition point is 62% higher (from $n = 40$ to $n = 65$) when considering raw QPU outputs and 60% higher ($n = 50$ to $n = 80$) after postprocessing.
- Comparison of right endpoints shows that Advantage 4.1 finds intact solutions at higher $n$. On the left panel, Advantage 4.1 goes 40% higher $n$ than the 2000Q QPU

($n = 50$ vs. $n = 70$). On the right panel, we see that Advantage 4.1 is able to realize the full potential of higher qubit counts and greater connectivity, finding solutions to problems of size up to $n = 177$, 2.8 times as big as $n = 64$ for the 2000Q QPU. The highest relative error from the 2000Q QPU is around 0.05 at $n = 65$: note that Advantage 4.1 achieves the same-quality performance on problems that are 2.5 times larger, at $n = 125$.

- 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 4.1 QPU can find optimal solutions at problem sizes for which the 2000Q solver 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

Better embeddings on the Pegasus topology, plus better raw performance due to new materials and design processes, means that the Advantage 4.1 QPU can return better-quality solutions than the 2000Q QPU on some inputs. It also means that Advantage 4.1 can find same-quality solutions faster (needing fewer samples), and that Advantage 4.1 can solve problems *to optimality* at higher input sizes $n$.

This section describes three small case studies using embedded inputs to compare performance of the two QPUs under a various performance metrics. In all three cases the Advantage 4.1 QPU achieves equal or superior performance to the 2000Q QPU.

**Random Cliques** In this section we compare performance of an Advantage 4.1 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 200$\mu$s, $R = 1000$ output samples are read, and the `extended_j_range` option is turned on. The MV postprocessing tool was used throughout these tests.

A small exploratory study was performed to identify optimal 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.

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, $\pm 2$ of those shown in the table have nearly imperceptible effect on solution quality as measured by $M$.

| Tuned Chain Strength | | | | | | |
|---|---|---|---|---|---|---|
| $n$ : | 10 | 20 | 30 | 40 | 50 | 60 |
| 2000Q | 4 | 6 | 7.3 | 8.4 | 12 | 10 |
| Advantage 4.1 | 4 | 5.8 | 7.2 | 8 | 8 | 10.2 |

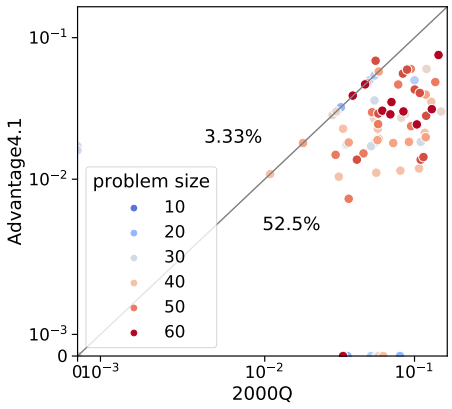



**Figure 6:** The table shows the best chain strength $J_{chain}$ for each QPU and problem size, to minimize the median scaled error $M$. The graph shows an input-by-input comparison of $M$ for the Advantage 4.1 (y-axis) and the 2000Q (x-axis) QPUs using tuned chain strengths. Points are color-coded by problem size $n$. The Advantage 4.1 QPU found better values of $M$ the 2000Q QPU, on about 52.5% of inputs, especially the largest inputs, whereas the 2000Q found better results on just 3.33% of inputs. Percentages do not sum to 100 because many outcomes were ties.

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 4.1 system significantly outperforms the 2000Q system. In this test the win percentage for Advantage 4.1 was 52.5 percent, over 15 times higher than the 3.33 percent observed for the 2000Q QPU.

Finally, Figure 7 compares performance of the two solvers under the *success probability* metric denoted $\pi$, which is estimated by the proportion of solutions in each sample having energies matching the reference energy.

The left panel shows median values of $\pi$ 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 achieved median success probability near $\pi = 0.054$, which is 54 times higher than $\pi \approx 0.001$ from the 2000Q system.

Note that when chain strength is optimally tuned, the 2000Q can find optimal solutions to these inputs for $n \leq 50$, whereas Figure 5 (right panel) indicates that the Advantage QPU (without postprocessing) can find optimal solutions up to $n = 70$, even when chain strength is not optimally tuned. This represents an increase of at least 40 percent in the size

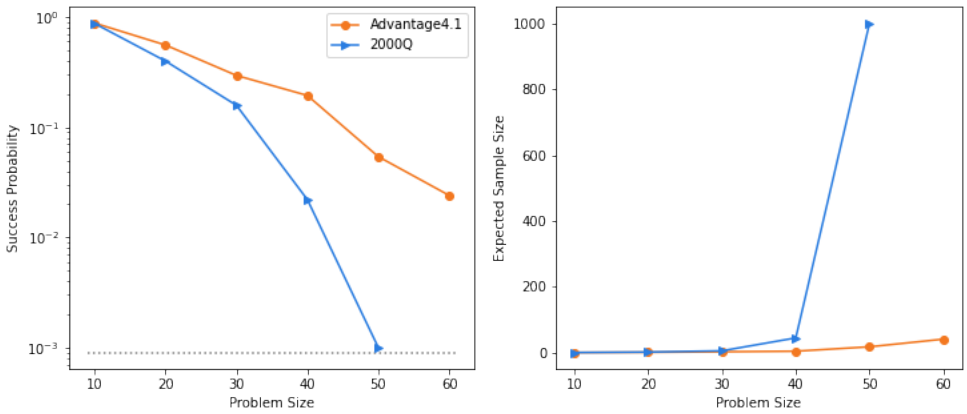



**Figure 7:** The graph shows median success probabilities for Advantage4.1 and 2000Q QPUs on random cliques. In these tests using 1000 reads, a success probability below 0.001 (i.e. below the gray dotted line), as observed for 2000Q at $n = 60$, is recorded as zero and not shown on this logarithmic scale. Throughout this range of problem sizes, the Advantage 4.1 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 about 54 times smaller for Advantage 4.1 than for the 2000Q QPU
.

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 54-fold speedup in $R$ maps to over a 14-fold speedup in wall clock time, from about 0.42 seconds to 0.03 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 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.

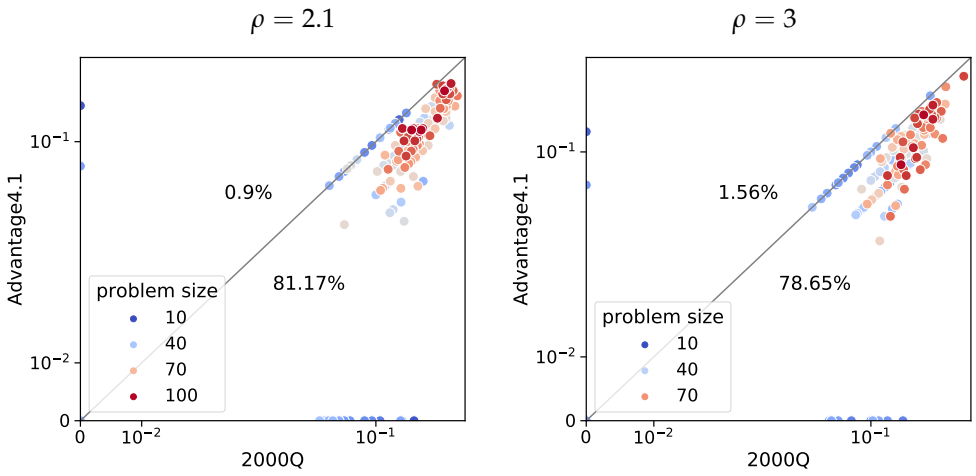



**Figure 8:** The left and right panels show results for $\rho = 2.1$ and $\rho = 3$, respectively. Maximum input size is $n = 100$ in the left panel and $n = 90$ in the right panel. The graphs present input-by-input comparisons of median relative error $M$, for the Advantage 4.1 (y-axis) and 2000Q (x-axis) QPUs. In each panel, points below the diagonal line indicate cases where Advantage 4.1 returned superior solutions. The percentages do not sum to 100 because many ties were observed.

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, and different runs will produce different embeddings. See [6, 7] 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 4.1 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 logical problems. For each input we took 1000 samples from each QPU. Anneal time was set to 200$\mu$s; 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 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 QPU returned strictly better solutions than the 2000Q QPU. The win percentages shown in the panels do not sum to 100 because the two QPUs tied in many cases.

As before, the Advantage system finds better solutions than the 2000Q systems on a higher proportion of inputs. For $\rho = 2.1$ the 81.17% win percentage is about 90 times higher, and for $\rho = 3$ the 78.65% percentage is about 50 times higher.

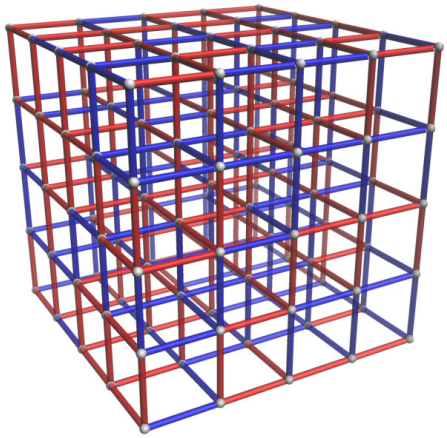


**Figure 9:** A $5 \times 5 \times 5$ 3D spin glass, from [1]. The edge weights are assigned $J_{ij} = +1$ (red) or $J_{ij} = -1$ (blue) uniformly at random.

**3D Lattices** We finish the section with a summary of a performance study described by King [1]. That work compares performance of an Advantage QPU (specifically the Advantage 1.1 system described in Appendix A) and a D-Wave 2000Q LN (lower noise) QPU, at the task of sampling ground states in 3D lattices.

This section presents an updated comparison using the Advantage 4.1 solver, which was not available at the time that paper was published. Results from the published comparison to the Advantage 1.1 solver may be found in Appendix A.

Several recent papers describe projects in scientific research that use D-Wave 2000Q QPUs to simulate quantum processes and to sample low-energy states in regular lattice structures such as these (see [11, 12]).

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:[3] 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

[3] The 2000Q QPU in this test has 2041 operable qubits out of 2045, while the Advantage 4.1 QPU has 5627 operable qubits out of 5750.

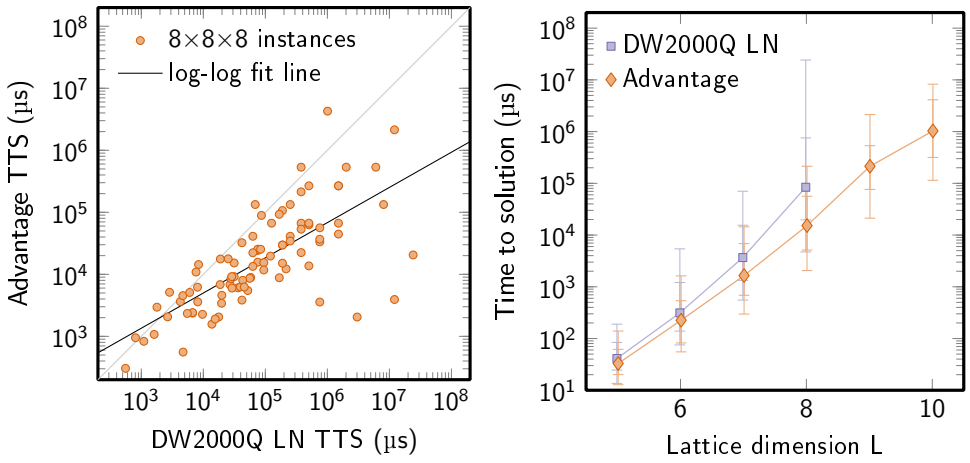



**Figure 10:** Time to solution (TTS). This is an updated version of results from [1], using the Advantage 4.1 solver. The left panel shows a scatterplot of mean TTS for random $8 \times 8 \times 8$ problems. Points below the diagonal line indicate cases where the Advantage 4.1 QPU outperforms the 2000Q QPU. The right plot shows median TTS for each system as a function of input size. Beyond the capability of solving larger problems, the Advantage QPU outperforms the 2000Q QPU both in terms of absolute computation times and in better scaling on 3D spin glasses.

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 $\mathrm{TTS}(t_a)$ (over all $t_a$ in the test), defined as follows:

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

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 4.1 processor is about 10 times lower than TTS for the 2000Q system. The better scaling (lower slope) of the Advantage processor on these inputs is also notable.

# 4 Conclusions

This report describes features of the Advantage quantum computer in comparison to its predecessor, the D-Wave 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 SDK.

Because of higher qubit counts and greater connectivity, Advantage QPUs can hold inputs that are typically between 2 and 5 times larger than those that fit on 2000Q QPUs, and chain

lengths are generally half as long.

The performance comparison in Section 3.2 compares a specific Advantage system (referred to here as Advantage 4.1) to a specific 2000Q LN system (referred to as 2000Q), using three categories of embedded inputs: cliques, NAE3SAT inputs, and 3D lattices.

Results from this section demonstrates that, beyond the ability to solve larger problems, the Advantage 4.1 solver can find better-quality solutions than the 2000Q solver when both are allowed equivalent computational resources (cliques and NAE3SAT problems); Advantage 4.1 can also find equivalent-quality solutions faster, and can solve problems to optimality at larger problem sizes (cliques and 3D lattices).

We attribute this performance improvement to the new Pegasus graph topology and its support for shorter and stronger chains in minor-embedded inputs. Furthermore, the Advantage 4.1 solver incorporates several technological innovations — in chip design, materials, and fabrication processes — to achieve better overall performance than the first Advantage 1.1 solver.

The tests described in this report focus on performance as would be experienced by practitioners with real-world applications, 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 SDK.

The preliminary results presented here are only the beginning: we look forward to more extensive performance studies and to more demonstrations of superior performance in comparison to both quantum and classical alternatives, for Advantage quantum systems, as well as to future generations of D-Wave annealing-based quantum computes.

# References


1 A. King and W. Bernoudy, "Performance benefits of increased qubit connectivity in quantum annealing 3-dimensional spin glasses," arXiv:200912479[quant-ph] (2020).

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

3 C. McGeoch and P. Farré, *Hybrid Solver for Constrained Quadratic Models*, dwavesys.com/resources (2021).

4 K. Boothby et al., *Next-Generation Topology of D-Wave Quantum Processors*, dwavesys.com/resources (2019).

5 A. Lucas, "Ising formulations of many NP problems," Frontiers in Physics **2** (2014).

6 "D-Wave Ocean Software Documentation, keyword: minor embedding," http://docs.ocean.dwavesys.com/en/stable/index.html (2020).

7 "D-Wave Ocean Software Documentation, keyword: find clique embedding," http://docs.ocean.dwavesys.com/en/stable/index.html (2020).

8 "D-Wave's Ocean Software," http://ocean.dwavesys.com (2020).

9 "D-Wave Ocean Software Documentation," http://docs.ocean.dwavesys.com/en/stable/index.html (2020).

10 "D-Wave Ocean Software Docuentation, keyword: post processing," http://docs.ocean.dwavesys.com/en/stable/index.html (2020).

11 R Harris, "Phase transitions in a programmable quantum spin glass simulator," Science **316**, 162–165 (2018).

12 A. King et al., "Observation of topological phenomenon in a programmable lattice of 1,800 qubits," Nature **560**, 456–460 (2018).

13 McGeoch, C. and Farré, P., *The Advantage System: An Overview*, dwavesys.com/resources (2020).

# Appendix A: The Advantage 1.1 Solver

The launch of the first Advantage-generation processor in September 2020 (here referred to as Advantage 1.1) was accompanied by publication of a D-Wave Technical Report (TR) [13], comparing its performance to a previous-generation D-Wave 2000Q LN (lower noise) QPU.

The main text of this TR updates the performance analysis from that previous TR, with a comparison of the same 2000Q solver to an Advantage performance update (called the Advantage 4.1 solver) that was made publicly available in October 2021.

This appendix preserves the data from the original TR, which describes results from identical tests carried out on Advantage 1.1 and the 2000Q solver. These results, together with some new direct comparisons of Advantage 1.1 and Advantage 4.1 QPUs, are presented in Section A.2.

In addition, Section A.1 describes a small study that compares performance of the Advantage 4.1 and Advantage 1.1 solvers using native (unembedded) inputs. This study highlights performance upgrades within the same generation of Advantage quantum systems.

Overall, the Advantage 4.1 solver outperforms both the 2000Q solver and the Advantage 1.1 solver in nearly every performance metric used in our tests.

Superior performance of Advantage-generation systems compared to 2000Q systems is largely due to the introduction of the Pegasus architecture, which has several beneficial features compared to the Chimera architecture used in D-Wave 2000Q and earlier-generation quantum systems.

Performance improvements within the same-generation Advantage product line may be attributed to technological advances that make possible higher qubit and coupler yields, as well as to innovations in chip design, and improvements in fabrication materials and processes.

## A.1 Performance on Native Inputs

A *native* input for a quantum annealer is mapped directly onto qubits and couplers, and contains no chains. Although native inputs are not tied to real-world applications, they give insight about raw performance of qubits and couplers inside a given quantum processor.

In this study we compare Advantage 4.1 (available in October 2021) to its predecessor Advantage 1.1 (launched in September 2020). For accurate comparison, we use the intersection graph $H^*$ containing only qubits and couplers that are active on *both* processors.

For each problem size, we generate 101 RAN1 inputs on $H^*$, which have $h_i = 0$, and $J_{ij} = \pm 1$ assigned uniformly at random. Problem sizes correspond to square subregions of the Pegasus graph, from P2 up to P16.

QPU parameters are set to anneal time $t_{anneal} = 200\mu$s, and number of reads $R = 1000$ per instance, with 10 spin reversal transforms per read.

For each instance we record the best energy $B$ found among all solutions from both QPUs. For each solution $S$ we compute the positive energy difference $\Delta(B, S)$ (accounting for pos-

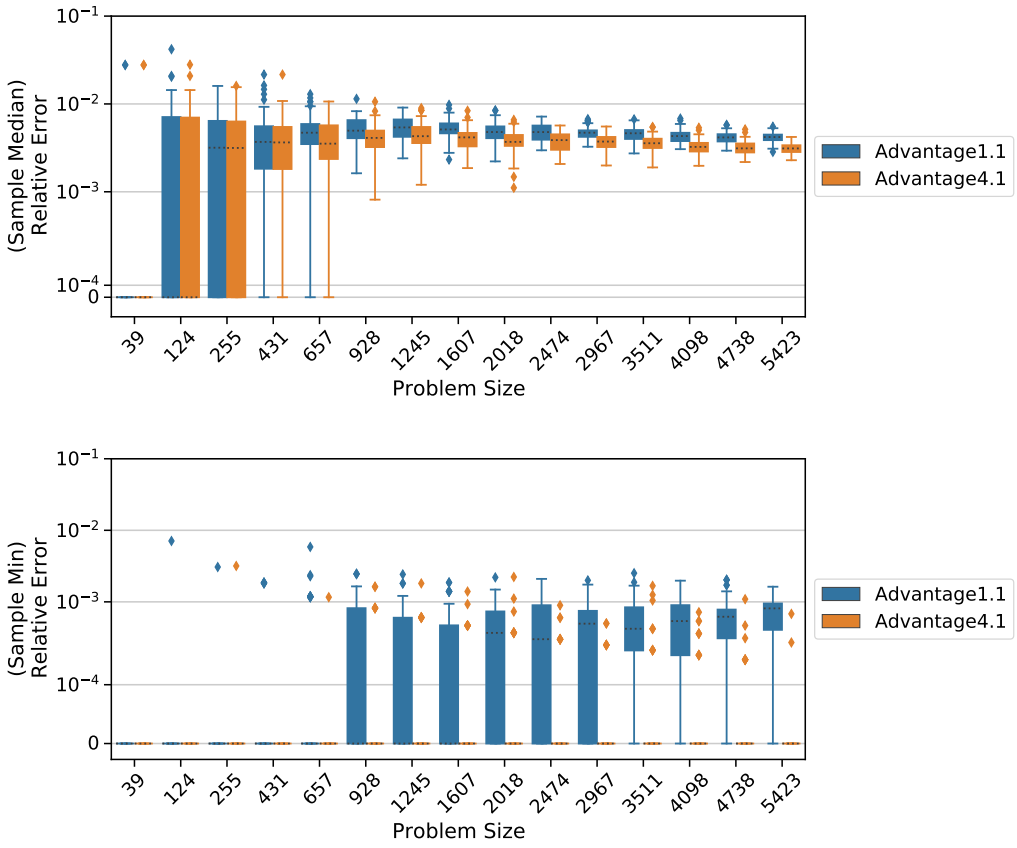



**Figure A1:** Performance on RAN1 inputs. The top panel plots relative errors in sample *medians* on the y axis, versus input size in qubits (corresponding to increasing square Pegasus grids P2 to P16). Boxplots show distributions of sample medians over 101 inputs, for the Advantage 1.1 (blue) and Advantage 4.1 (orange) processors. The data is plotted on a symlog y scale, which is logarithmic except for a small region near $y = 0$, where a linear scale is used. The bottom panel shows results for sample *minimums* from the same experiment.

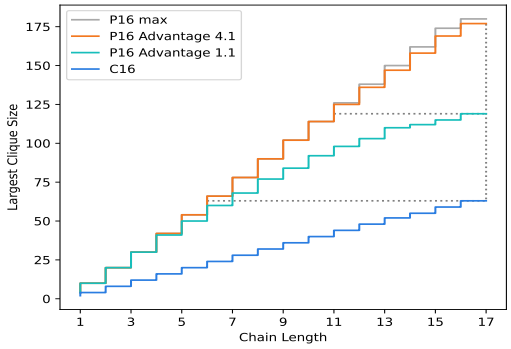
sible sign differences), and the *relative error* $\rho = \Delta(B,S)/|B|$. A minimum relative error $\rho = 0$ is the best possible outcome in this metric.

Figure A1 shows the results of this experiment. The top panel shows a boxplot of the distribution of relative errors from the *median* of each sample of size $R$ returned by each processor. On smallest inputs, processor performance is nearly equivalent; on largest inputs the distribution of median relative errors is consistently better for Advantage 4.1.

The bottom panel shows the distribution of relative errors from sample *minimums* in each test. At all input sizes, the bulk of the data distribution for Advantage 4.1 — covered by boxplot and tails — is located entirely at zero. This indicates that except for a few outliers (orange diamonds), this the Advantage 4.1 solver found best sample minimums in nearly every test on nearly every input. Advantage 1.1 achieves similar solution qualities at smallest input sizes, but the center and quartiles of the distribution of relative errors steadily increase as input sizes grow above $q = 657$.

## A.2 Tests from Sections 2 and 3

The figures appearing in this section may be compared to the same-numbered figures in Sections 2 and 3 of the main text. See those sections for detailed descriptions of each test and its outcomes. Most graphs in this Appendix are taken from an earlier technical report comparing Advantage1.1 to the 2000Q QPU. A few additional graphics have been included to facilitate direct comparison of all three solvers: Advantage 4.1, Advantage 1.1, and the 2000Q.




| Maximum Graph Sizes | | | | |
|---|---|---|---|---|
| Logical Graph | Degree | C16 $n$ | P16 $n$ | Ratio 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 Size | C16 Chain Length | P16 Chain Length | Ratio 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 A3:** Compare to Figure 3 of the main text. The graph at top shows the largest graph size for each chain length for all three processors: 2000Q C16 (blue), Advantage 1.1 (teal) and Advantage 4.1. A gray curve showing what is possible for a fully yielded Pegasus graph is also shown: Advantage 4.1 very nearly achieves perfect performance in this regard. The table below shows typical embeddable sizes and chain lengths for representative graphs on Advantage 1.1. Note that some test parameters (input sizes) are slightly different in some cases from those in the main text.

| Chain Length per Problem Size | | | | |
|---|---|---|---|---|
| Problem size $n$: | 30 | 40 | 50 | 60 |
| 2000Q C16 | 9 | 11 | 14 | 17 |
| Advantage 1.1 P16 | 4 | 6 | 7 | 9 |
| Advantage 4.1 P16 | 4 | 5 | 6 | 7 |

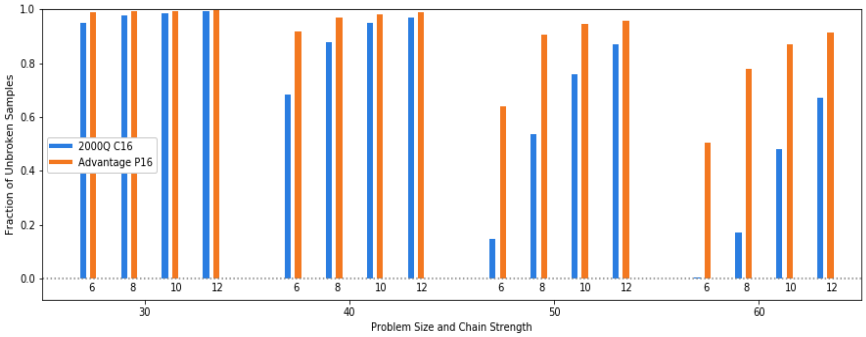



**Figure A4:** Compare to Figure 4 in the main text. The table (top) compares chain lengths for the 2000Q, Advantage 1.1 and Advantage 4.1 solvers, in clique embeddings of various sizes $n$. The graph (bottom) shows the percentage of intact samples returned by the 2000Q (blue) and the Advantage 1.1 (orange), when different chain strengths $J_{chain} = [2, 4, 6, 8]$ are assigned. Interestingly, the fraction of unbroken samples is slightly lower for Advantage 4.1 than for Advantage 1.1, even though solution quality is uniformly better. This phenomenon will be examined further in a followup study.

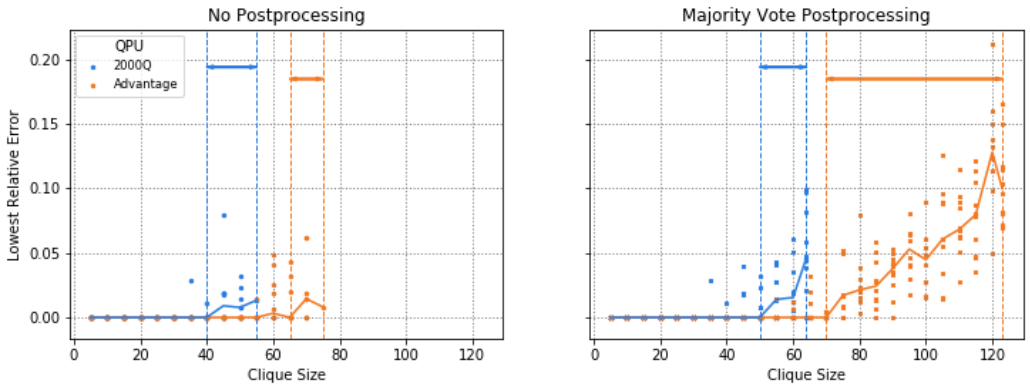



**Figure A5:** Compare to Figure 5 in the main text. The graphs compare solution quality found by the 2000Q (blue) and Advantage 1.1 (orange) QPUs when chain strengths are set to $J_{chain} = 8$, too low for large $n$. In the left panel, broken solutions are discarded; in the right panel, broken solutions are repaired using majority vote postprocessing.

| Tuned Chain Strength | | | | | | |
|---|---|---|---|---|---|---|
| $n$ : | 10 | 20 | 30 | 40 | 50 | 60 |
| 2000Q | 4 | 6 | 7.3 | 8.4 | 12 | 10 |
| Advantage 1.1 | 4 | 6 | 6 | 8 | 8 | 10 |
| Advantage 4.1 | 4 | 5.8 | 7.2 | 8 | 8 | 10.2 |

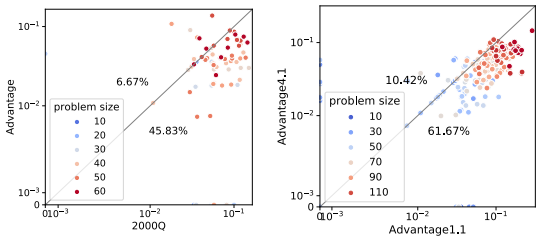



**Figure A6:** Compare to Figure 6 in the main text. The table shows the best chain strength $J_{chain}$ for each QPU by problem size, to minimize the median scaled error $M$. The left panel shows an input-by-input comparison of $M$ for the Advantage 1.1 (y-axis) and the 2000Q (x-axis) QPUs using tuned chain strengths. The right panel shows a comparison of the Advantage 4.1 (y-axis) and the Advantage 1.1 (x-axis) QPUs. Percentages do not sum to 100 because many outcomes were ties.

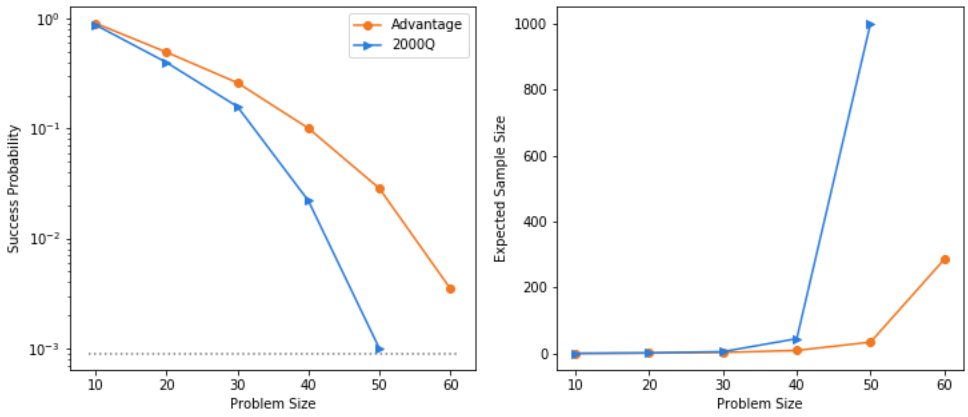



**Figure A7:** Compare to Figure 7 in the main text. The left panel shows median success probabilities $\pi$ for Advantage 1.1 and 2000Q QPUs on random cliques; the right panel shows $1/\pi$, corresponding to the expected number of reads needed to observe an optimal solution. In these tests using 1000 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.

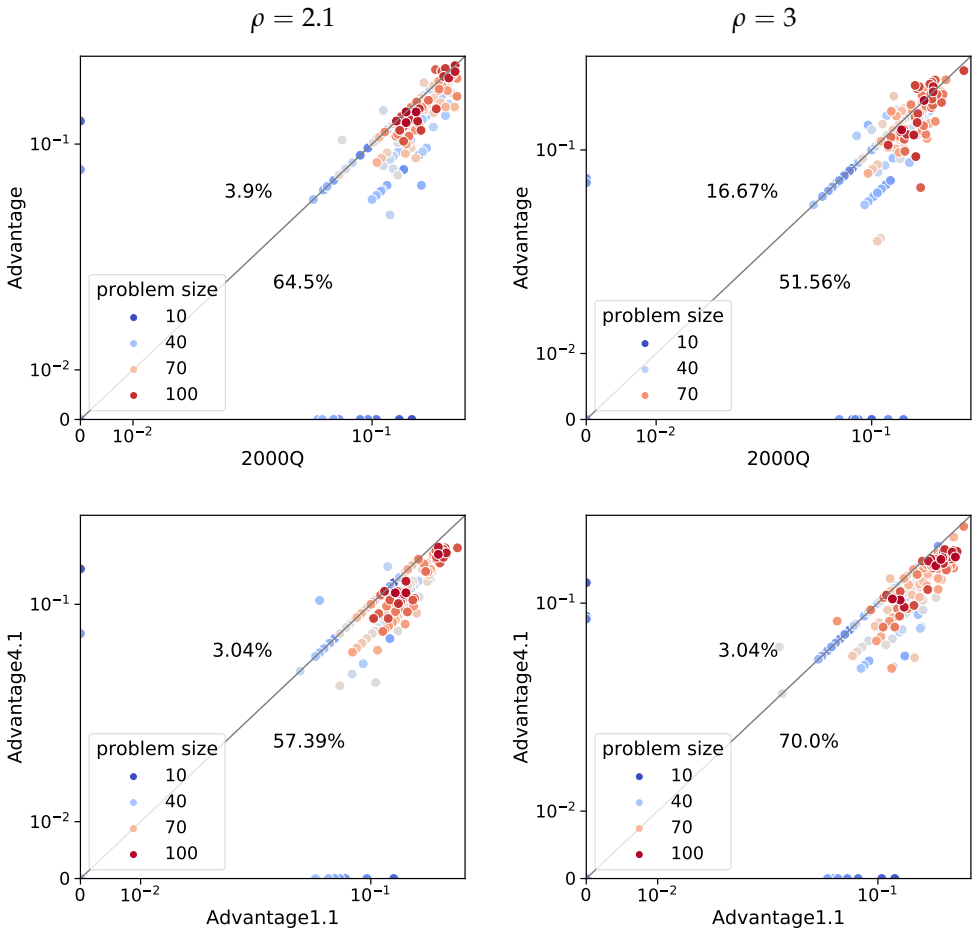



**Figure A8:** Compare to Figure 8 in the main text. The top two panels show results for Advantage 1.1 (y-axis) and the 2000Q QPU (x axis) on NAE3SAT inputs with $\rho = 2.1$ and $\rho = 3$, respectively. Maximum input size is $n = 100$ in the left panel and $n = 90$ in the right panel. Points below the diagonal correspond to cases where Advantage 1.1 returned better results than the 2000Q solver. The bottom two panels show results for Advantage 4.1 (y-axis) versus Advantage 1.1 (x-axis): maximum input sizes are $n = 100$ and $n = 100$, respectively. Points below the diagonal correspond to cases where Advantage 4.1 returned better results than the 2000Q solver. The percentages do not sum to 100 because many ties were observed.

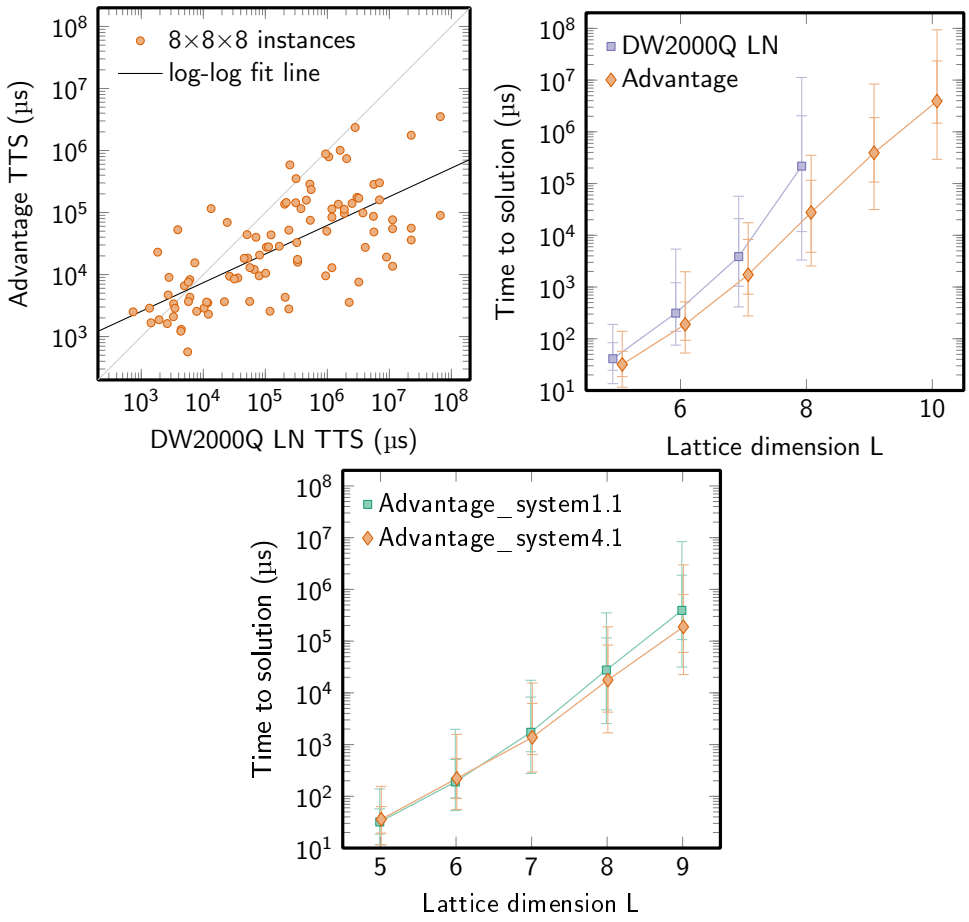



**Figure A10:** Compare to Figure 10 in the main text. Time to Solution (TTS), from [1]. The top left panel shows a scatterplot of mean TTS for random $8 \times 8 \times 8$ problems. Points below the diagonal line indicate cases where Advantage 1.1 outperforms the 2000QLN QPU. The top right panel shows scaling of median TTS for each system as a function of input size. The bottom panel shows a scaling comparison of the two Advantage processors, usiing inputs up to size $9 \times 9 \times 9$. In the TTS metric, Advantage 4.1 is about two times faster than Advantage 1.1 on the largest inputs.