Leveraging Quantum Annealing for Large MIMO Processing in Centralized Radio Access Networks

Minsung Kim
Princeton University
minsungk@cs.princeton.edu

Davide Venturelli
USRA Research Institute for Advanced Computer Science
DVenturelli@usra.edu

Kyle Jamieson
Princeton Univeristy
kylej@cs.princeton.edu

ABSTRACT

User demand for increasing amounts of wireless capacity continues to outpace supply, and so to meet this demand, significant progress has been made in new MIMO wireless physical layer techniques. Higher-performance systems now remain impractical largely only because their algorithms are extremely computationally demanding. For optimal performance, an amount of computation that increases at an exponential rate both with the number of users and with the data rate of each user is often required. The base station's computational capacity is thus becoming one of the key limiting factors on wireless capacity. *QuAMax* is the first large MIMO centralized radio access network design to address this issue by leveraging quantum annealing on the problem. We have implemented QuAMax on the 2,031 qubit D-Wave 2000Q quantum annealer, the state-of-the-art in the field. Our experimental results evaluate that implementation on real and synthetic MIMO channel traces, showing that 10 μs of compute time on the 2000Q can enable 48 user, 48 AP antenna BPSK communication at 20 dB SNR with a bit error rate of 10^{-6} and a 1,500 byte frame error rate of 10^{-4}.

CCS CONCEPTS

• **Networks** → **Wireless access points, base stations and infrastructure**; • **Hardware** → **Quantum computation**;

KEYWORDS

Wireless Networks, Massive MIMO, Maximum Likelihood Detection, Sphere Decoder, Quantum Computing, Quantum Annealing

ACM Reference Format:
Minsung Kim, Davide Venturelli, and Kyle Jamieson. 2019. Leveraging Quantum Annealing for Large MIMO Processing in Centralized Radio Access Networks. In *SIGCOMM '19: 2019 Conference of the ACM Special Interest Group on Data Communication, August 19–23, 2019, Beijing, China.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3341302.3342072

1 INTRODUCTION

A central design challenge for future generations of wireless networks is to meet users' ever-increasing demand for capacity and throughput. Recent advances in the design of wireless networks to this end, including the 5G efforts underway in industry and academia, call in particular for the use of Large and Massive *multiple input multiple output* (MIMO) antenna arrays to support many users near a wireless access point (AP) or base station sharing the wireless medium at the same time.[1]

Much effort has been and is currently being dedicated to large and massive MIMO, and these techniques are coming to fruition, yielding significant gains in network throughput. An apropos example is Massive MIMO: in LTE cellular and 802.11ac local-area networks, up to eight antennas are supported at the AP *spatially multiplexing* [65] parallel information streams concurrently to multiple receive antennas. The technique is also known as multi-user MIMO (MU-MIMO) and can be used both in the uplink and the downlink of multi-user MIMO networks: in the uplink case, several users concurrently transmit to a multi-antenna AP, while over the downlink, the AP multiplexes different information streams to a number of mobile users.

From a design standpoint, one of the most promising and cost effective architectures to implement 5G technologies is the centralized radio access network (C-RAN) architecture [48, 60]. C-RAN pushes most of the physical-layer processing that currently takes place at the AP to a centralized data center, where it is aggregated with other APs' processing on the same hardware. The C-RAN concept has undergone several iterations since its inception, with more recent work treating the unique demands of small cells [63], centralizing most of the computation and supporting hundreds or thousands of the APs from a centralized data center.

To fully realize Massive MIMO's potential throughput gains, however, the system must effectively and efficiently demultiplex mutually-interfering information streams as they arrive. Current large MIMO designs such as Argos [62], BigStation [76], and SAM [64] use linear processing methods such as zero-forcing and minimum mean squared error (MMSE) filters. These methods have the advantage of very low computational complexity, but suffer when the MIMO channel is poorly-conditioned [50], as is often the case when the number of user antennas approaches the number of antennas at the AP [50, 76]. Sphere Decoder-based *maximum likelihood* (ML) MIMO decoders/detectors [10, 50] can significantly improve throughput over such linear filters, but suffer from increased computational complexity: compute requirements increase exponentially with the number of antennas [30], soon becoming prohibitive (Section 2.1).

The problem of limited computational capacity stems from the requirement that a receiver's physical layer finish decoding a frame before the sender requires feedback about its decoding success or failure. For Wi-Fi networks, *e.g.*, this quantity is on the order of

[1]We use the terms "access point" and "base station" interchangeably in this paper.

tens of μs, the spacing in time between the data frame and its acknowledgement. More sophisticated physical layers, such as 4G LTE, require the receiver to give feedback in the context of incremental redundancy schemes, for the same reason; the processing time available is 3 ms for 4G LTE and 10 ms for WCDMA [19, 76]. As a result, most current systems adopt linear filters, accepting a drop in performance.

New approach: Quantum computation in the data center. This paper explores the leveraging of quantum computation (QC) to speed up the computation required for the ML MIMO decoder. We place our ideas in the context of the QC currently already realized in experimental hardware, and in the context of the dominant C-RAN architecture. Here we imagine a future quantum computer, co-located with C-RAN computational resources in a data center, connected to the APs via high-speed, low-latency fiber or millimeter-wave links.

Optimization is one of the key applications the quantum community has identified as viable in the short-term (*i.e.* before quantum processors become scalable devices capable of error correction and universality). While their potential in optimization is largely unproven, it is believed that it may be possible for *Noisy Intermediate-Scale Quantum* (NISQ) devices to achieve polynomial or exponential speedups over the best known classical algorithms [56]. It is, however, important to leverage understanding from current prototypes in order to inform the design of real-world systems, since performance cannot be predicted or simulated efficiently, especially in the presence of device-specific noise. This is the approach we therefore advocate here. Two main approaches have been identified for quantum optimization in NISQs: *Quantum Annealing* (QA) and *Quantum Approximate Optimization Algorithms* (QAOA). The former approach is a form of analog computation that has been developed theoretically in the early nineties [25], but realized experimentally in a programmable device only in 2011 by D-Wave Systems. We focus on QA here, discussing QAOA briefly in Section 6.

This paper presents *QuAMax*, the first system to apply QA to the computationally challenging ML MIMO wireless decoding problem in the context of a centralized RAN architecture where a QA is co-located in a data center serving one or more wireless APs. The contributions of our paper can be summarized as follows: Firstly, we present the first reduction of the ML MIMO decoding problem to a form that a QA solver can process. Secondly, we introduce a new, communications-specific evaluation metric, *Time-to-BER (TTB)*, which evaluates the performance of the QA as it aims to achieve a target bit error rate (BER) on the decoded data. Finally, we evaluate QuAMax with various scenarios and parameter settings and test their impact on computational performance. To achieve a BER of 10^{-6} and a frame error rate of 10^{-4}, ML MIMO detection on the D-Wave 2000Q quantum annealer requires 10–20 μs of computation time for 48-user, 48-AP antenna binary modulation or 30–40 μs for 14×14 QPSK at 20 dB SNR, and with the real-world trace of 8×8 MIMO channel, the largest spatial multiplexing MIMO size publicly available for experiments [61], QuAMax requires 2 μs for BPSK and 2–10 μs for QPSK.

Table 1: Sphere Decoder visited node count [50], complexity over 10,000 instances, and practicality on a Skylake Core i7 architecture.

BPSK	QPSK	16-QAM	Complexity (Visited Nodes)
12×12	7×7	4×4	$\approx$ 40 (feasible)
21×21	11×11	6×6	$\approx$ 270 (borderline)
30×30	15×15	8×8	$\approx$ 1,900 (unfeasible)

Paper roadmap. The next section is a background primer on ML detection and QA. Section 3 details our programming of the ML problem on the QA hardware. Section 4 describes QuAMax implementation in further detail, followed by our evaluation in Section 5. We conclude with a review of related work (Section 6), discussion of current status of technology and practical considerations (Section 7), and final considerations (Section 8).

2 BACKGROUND

In this section, we present primer material on the MIMO ML Detection problem (§2.1), and Quantum Annealing (§2.2).

2.1 Primer: Maximum Likelihood Detection

Suppose there are N_t mobile users, each of which has one antenna, each sending data bits to a multi-antenna ($N_r \geq N_t$) MIMO AP based on OFDM, the dominant physical layer technique in broadband wireless communication systems [49]. Considering all users' data bits together in a vector whose elements each comprise a single user's data bits, the users first map those data bits into a complex-valued *symbol* $\bar{\mathbf{v}}$ that is transmitted over a radio channel: $\bar{\mathbf{v}} = [\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_{N_t}]^\intercal \in \mathbb{C}^{N_t}$. Each user sends from a *constellation* O of size $|O| = 2^Q$ (Q bits per symbol). The MIMO decoding problem, whose optimal solution is called the ML solution, consisting of a search over the sets of transmitted symbols, looking for the set that minimizes the error with respect to what has been received at the AP:

$$\hat{\mathbf{v}} = \arg\min_{\mathbf{v} \in O^{N_t}} \|\mathbf{y} - \mathbf{H}\mathbf{v}\|^2 . \tag{1}$$

The ML decoder then "de-maps" the decoded symbols $\hat{\mathbf{v}}$ to decoded bits $\hat{\mathbf{b}}$. In Eq. 1, $\mathbf{H} \in \mathbb{C}^{N_r \times N_t} = \mathbf{H}^I + j\mathbf{H}^Q$ is the wireless channel[2] on each OFDM subcarrier and $\mathbf{y} \in \mathbb{C}^{N_r}$ ($= \mathbf{H}\bar{\mathbf{v}} + \mathbf{n}$) is the received set of symbols, perturbed by $\mathbf{n} \in \mathbb{C}^{N_r}$, additive white Gaussian noise (AWGN). This solution minimizes detection errors, thus maximizing throughput (*i.e.*, throughput-optimal decoding).

The *Sphere Decoder* [1, 20] is a ML detector that reduces complexity with respect to brute-force search by constraining its search to only possible sets $\mathbf{v}$ that lie within a hypersphere of radius $\sqrt{C}$ centered around $\mathbf{y}$ (*i.e.,* Eq. 1 with constraint $\|\mathbf{y} - \mathbf{H}\mathbf{v}\|^2 \leq C$). It transforms Eq. 1 into a tree search [71] by QR decomposition $\mathbf{H} = \mathbf{Q}\mathbf{R}$, where $\mathbf{Q}$ is orthonormal and $\mathbf{R}$ upper-triangular, resulting in $\hat{\mathbf{v}} = \arg\min_{\mathbf{v} \in O^{N_t}} \|\bar{\mathbf{y}} - \mathbf{R}\mathbf{v}\|^2$, with $\bar{\mathbf{y}} = \mathbf{Q}^*\mathbf{y}$. The resulting tree has a height of N_t, branching factor of $|O|$, and $1 + \sum_{i=1}^{N_t} |O|^i$ nodes. ML detection becomes the problem of finding the single leaf among $|O|^{N_t}$ with minimum metric; the corresponding tree path is the

[2]The channel changes every channel *coherence time*, and is practically estimated and tracked via preambles and/or pilot tones. Typical coherence time at 2 GHz and a walking speed is *ca.* 30 ms [67].

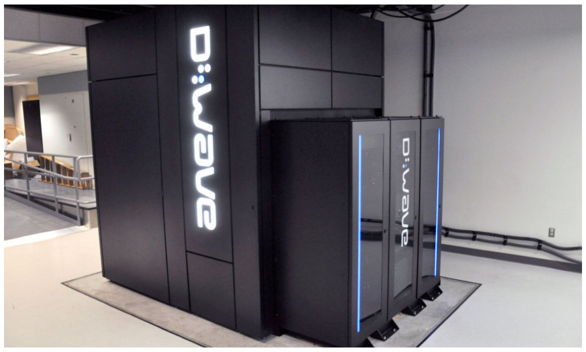


Figure 1: A D-Wave 2000Q (DW2Q) machine at NASA Ames Research Center, which hosts a *Whistler* processor manufactured with 2,048 qubits and 5,019 qubit-coupling parameters. The chip is hosted in a high-vacuum, magnetically shielded enclosure at a temperature of about 13 milliKelvin.

ML solution. Thus, the min in Eq. 1 is a search in an exponentially-large space of transmitted symbols $\{\mathbf{v}\}$, despite Sphere Decoder reductions in the search space size [71].

Table 1 shows the average number of tree nodes visited to perform ML Sphere decoding, with clients transmitting modulation symbols on 50 subcarriers over a 20 MHz, 13 dB SNR (Signal to Noise Ratio) Rayleigh channel. The table is parameterized on the number of clients and AP antennas, and the modulation, highlighting the exponential increase in computation. For 8 clients with 16-QAM symbols, 15 clients with QPSK symbols, or 30 clients sending binary (BPSK) symbols, the Sphere Decoder visits close to 2,000 tree nodes, saturating, for example, Intel's Skylake core i7 architecture, whose arithmetic subsystem achieves an order of magnitude less computational throughput [32]. Since traditional silicon's clock speed is plateauing [14], the problem is especially acute.

2.2 Primer: Quantum Annealing

Quantum Annealers [37, 45] are specialized, analog computers that solve NP-complete and NP-hard optimization problems on current hardware, with future potential for substantial speedups over conventional computing [46]. Many NP-hard problems can be formulated in the Ising model [42] (*cf.* §3.2), which many QA machines use as input [6, 18]. NP-complete and NP-hard problems other than ML MIMO detection in the field of (wireless) networking that potentially benefit from QA include MIMO downlink precoding [44], channel coding [36, 75], network routing [11], security [26], and scheduling [27, 41].

Quantum Annealing hardware. Compared to simulated annealing, the classical algorithm from which QA inherits its name, QA aims to exploit quantum effects such as *tunneling*, *many-body delocalization* and *quantum relaxation* to circumvent computational bottlenecks that would otherwise trap Monte Carlo methods in local minima of the solution landscape. While exploiting QA is technologically challenging, with the appearance of the D-Wave quantum annealer (Fig. 1), the research community is now able to run experiments, and critically, to study under what conditions a noisy-intermediate-scale-quantum (NISQ) machine [56] can use quantum resources to deliver a speedup [34]. For instance, recently Boixo *et al.* [5] and Denchev *et al.* [22] have found evidence that tunneling under ideal conditions can be exploited on an earlier model of the D-Wave 2000Q (DW2Q) machine, delivering many orders of magnitude speedup against CPU-based simulated annealing, which is considered to be one of the best classical competitions to *Quantum Processing Units* (QPUs). QPUs also outperform GPU implementations by several orders of magnitude in random problems whose structure is related to real world optimization problems [38].

The DW2Q is an analog optimizer, meaning that it computes continuously rather than in discrete clock cycles, and that it represents numerical quantities as analog instead of digital quantities. The hardware initializes each of its N constituent *quantum bits*, or *qubits*, to begin in a *superposition state* $1/\sqrt{2}\,(|0\rangle + |1\rangle)$ that has no classical counterpart. In concrete terms, these qubits are metallic circuits in a chip that are maintained in a superconducting state by low temperature and subjected to the influence of tailored magnetic fluxes. The collection of N qubits at this point in time encodes all the possible 2^N outputs in a single state. This initial setting is achieved by exposing all the qubits in the chip to a signal $A(t)$ whose magnitude at this point in time is maximal. Then the system implements an *objective function* which is represented by another signal $B(t)$ and is ramped up from zero, while $A(t)$ is decreased progressively at the same time. The synchronized sequence of signals A and B and their time dependence is the *annealing schedule*. The schedule is essentially the QA algorithm, and has to be optimized so that at the end of the run ($B(t)$ = max and $A(t) = 0$), each qubit in the chip assumes either a value of $|0\rangle$ or $|1\rangle$, corresponding to classical bit values, 0 or 1, respectively. This final state of these qubits collectively represents a candidate solution of the problem, ideally the *ground state* of the system (*i.e.*, the minimum of the optimization objective function) [24, 35].

In practice, at the end of the run, the ground state will be found with a probability that depends on the degree to which the schedule is optimal for the problem at hand, as well as on the effect of uncontrollable QA noise and environmental interference on the annealer. While the quantum community is investigating physics principles to guide schedule parameters, most clearly-understood theoretical principles do not apply to current, imperfect experimental systems [34]. Hence the empirical approach, which we take in this paper, represents current state-of-the-art [58]. Three degrees of freedom are specifically investigated in this work.

- First, there are many ways of mapping a problem to an equivalent Ising formulation that runs on the machine (we investigate one such mapping in Section 3).
- Second, the user may accelerate or delay $A(t)/B(t)$ evolution, thus determining *annealing time* (1–300 μs), the duration of the machine's computation.
- Finally, the user may introduce stops (*anneal pause*) in the annealing process, which have been shown to improve performance in certain settings [43].

3 DESIGN

Starting from the abstract QA problem form (§3.1), QuAMax's design reduces ML detection to form (§3.2), then compiles it on actual hardware, a process called *embedding* (§3.3).

3.1 QA Problem Formulation

The first step in leveraging QA for any problem is to define the problem of interest as an objective function to be minimized, consisting of a quadratic polynomial of binary variables. We now introduce two equivalent forms of this objective functions, as is customary in the QA application literature.

1. Ising spin glass form. In this form the solution variables are traditionally referred to as *spins* $s_i \in \{+1, -1\}$.

$$\hat{s}_1, \ldots, \hat{s}_N = \arg \min_{\{s_1,\ldots,s_N\}} \left(\sum_{i<j}^{N} g_{ij} s_i s_j + \sum_{i}^{N} f_i s_i \right) \tag{2}$$

where N is the number of spin variables, and g_{ij} and f_i are the Ising *model parameters* that characterize the problem. The f_i characterize the preference for each spin to be +1 or −1: positive indicates a preference for −1 while negative indicates a preference for +1, with the magnitude corresponding to the magnitude of the preference for either state. The g_{ij} capture preferred correlations between spins: positive causes the QA to prefer $s_i \neq s_j$, while negative causes the QA to prefer $s_i = s_j$ in its optimization outcome. Analogously to f_i, the magnitude of g_{ij} corresponds to the magnitude of its preference.

2. QUBO form. The *Quadratic Unconstrained Binary Optimization* (*QUBO*) has solution variables q_i that are classical binary bits (zero- or one-valued):

$$\hat{q}_1, \ldots, \hat{q}_N = \arg \min_{\{q_1,\ldots,q_N\}} \sum_{i \leq j}^{N} Q_{ij} q_i q_j, \tag{3}$$

where N is the qubit count and $\mathbf{Q} \in \mathbb{R}^{N \times N}$ is upper triangular. The off-diagonal matrix elements Q_{ij} $(i \neq j)$ correspond to g_{ij} in Eq. 2, and the diagonal elements correspond to f_i.

The two forms are equivalent, their solutions related by:

$$q_i \leftrightarrow \frac{1}{2}(s_i + 1), \tag{4}$$

leading to $g_{ij} \leftrightarrow \frac{1}{4} Q_{ij}$ and $f_i \leftrightarrow \frac{1}{2} Q_{ii} + \frac{1}{4} \sum_{k=1}^{i-1} Q_{ki} + \frac{1}{4} \sum_{k=i+1}^{N} Q_{ik}$.

3.2 ML-to-QA Problem Reduction

We now explain our process for transforming the ML detection problem into the QUBO and Ising forms. Since QuAMax also assumes OFDM where the wireless channel is subdivided into multiple flat-fading orthogonal subcarriers [49], this ML-to-QA reduction is required at each subcarrier.

3.2.1 ML-to-QUBO problem reduction. Let's first consider the transformation of the ML problem into QUBO form—the key idea is to find a *variable-to-symbol transform function* $\mathbf{T}(\cdot)$ that represents the "candidate" vector $\mathbf{v}$ in the ML search process (Eq. 1 on p. 2) instead with a number of QUBO solution variables. Specifically, we represent each of the N_t senders' candidate symbols $v_i \in O$ $(1 \leq i \leq N_t)$, with $\log_2(|O|)$ QUBO solution variables, naturally requiring $N = N_t \cdot \log_2(|O|)$ QUBO variables for N_t transmitters, and form these QUBO variables into a vector $\mathbf{q_i}$ for each sender i: $\mathbf{q_i} = \left[q_{(i-1)\cdot\log_2(|O|)+1}, \ldots, \; q_{i\cdot\log_2(|O|)} \right]$. For example, $\mathbf{T}$ recasts a 2 × 2 QPSK ($|O| = 4$) problem into a QUBO problem with four solution variables, split into two vectors $\mathbf{q_1} = [q_1 \; q_2]$ and $\mathbf{q_2} = [q_3 \; q_4]$. In general, the transform recasts the ML problem of Eq. 1 into the form

$$\hat{\mathbf{q}}_1, \ldots, \hat{\mathbf{q}}_{N_t} = \arg \min_{\mathbf{q}_1,\ldots,\mathbf{q}_{N_t}} \|\mathbf{y} - \mathbf{He}\|^2, \tag{5}$$

where $\mathbf{e} = [\mathbf{T}(\mathbf{q}_1), \ldots, \mathbf{T}(\mathbf{q}_{N_t})]^\top$. Then, the resulting N_t vectors $\hat{\mathbf{q}}_1, \ldots, \hat{\mathbf{q}}_{\mathrm{N_t}}$ correspond to the N QUBO solution variables, $\hat{q}_1, \ldots, \hat{q}_N$. Continuing our 2 × 2 QPSK example, $\mathbf{e} = [\mathbf{T}(\mathbf{q}_1), \mathbf{T}(\mathbf{q}_2)]^\top = [\mathbf{T}([q_1, q_2]), \mathbf{T}([q_3, q_4])]^\top$. Then, Eq. 5 results in two ML-decoded vectors $\hat{\mathbf{q}}_1$, $\hat{\mathbf{q}}_2$ (noting that $\mathbf{T}(\hat{\mathbf{q}}_1)$, $\mathbf{T}(\hat{\mathbf{q}}_2)$ corresponds to the ML solution $\hat{\mathbf{v}} = [\hat{v}_1, \hat{v}_2]^\top$ in Eq. 1, the nearest symbol vector around received $\mathbf{y}$). The decoded vectors $\hat{\mathbf{q}}_1$,$\hat{\mathbf{q}}_2$ correspond to the four decoded QUBO variables $\hat{q}_1, \hat{q}_2, \hat{q}_3, \hat{q}_4$ in Eq. 3. If the transmitter's bit-to-symbol mapping and QuAMax's variable-to-symbol transform are equivalent, then the decoded $\hat{q}_1, \hat{q}_2, \hat{q}_3, \hat{q}_4$ are the directly de-mapped bits, $\hat{\mathbf{b}}$ from the ML solution in Eq. 1.

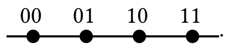
When transform $\mathbf{T}$ is linear the expansion of the norm in Eq. 5 yields a quadratic polynomial objective function, since $q_i^2 = q_i$ for any 0 or 1-valued q_i. Then the ML problem (Eq. 1) transforms directly into QUBO form (Eqs. 3 and 5). Our task, then, is to find variable-to-symbol linear transform functions $\mathbf{T}$ for each of BPSK, QPSK, and 16-QAM.

Binary modulation. If the two mobile transmitters send two signals simultaneously, each with one of two possible information symbols, their transmissions can be described with a two-vector of symbols $\bar{\mathbf{v}} = [\bar{v}_1, \bar{v}_2]^\top \in [\{\pm 1\}, \{\pm 1\}]^\top$. This type of data transmission is called *binary* modulation, of which one popular kind is *binary phase shift keying* (BPSK). The ML problem applied to the BPSK case where symbols v_i are represented by $v_i = \mathbf{T}(\mathbf{q_i}) = 2q_i - 1$ thus results in a QUBO form (a detailed derivation can be found in Appendix A).

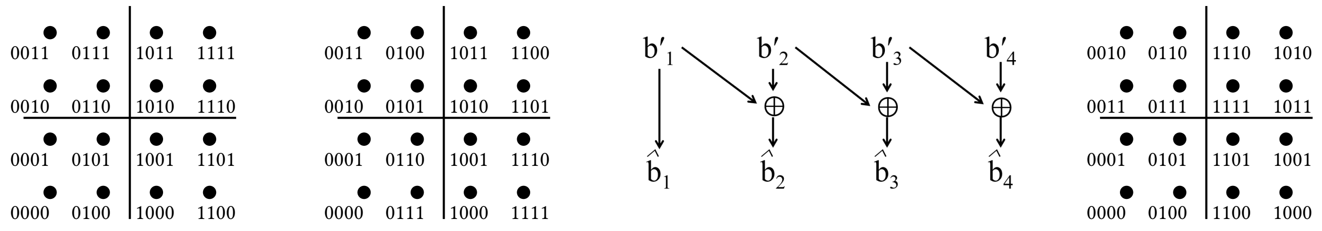
We next consider higher-order modulations, which send one of M possible information symbols with each channel use (where $M > 2$), resulting in higher communication rates.

QPSK modulation. In the case of *quadrature phase shift keying* (QPSK), each sender transmits one of four possible symbols $\bar{v}_i \in \{\pm 1 \pm 1j\}$. Since it can be viewed as a two-dimensional BPSK $v_i = v_i^I + jv_i^Q$, we represent each possibly-transmitted QPSK information symbol with the linear combination of one QUBO variable, plus the other QUBO variable times the imaginary unit. Transforming q_{2i-1} and q_{2i} to v_i^I and v_i^Q respectively leads to the transform $v_i = \mathbf{T}(\mathbf{q_i}) = (2q_{2i-1} - 1) + j(2q_{2i} - 1)$.

Higher-order modulation. 16 *quadrature amplitude modulation* (16-QAM) and higher-order modulations increase spectral efficiency, but utilize multiple amplitudes (levels) so require a $\mathbf{T}$ that inputs more than one (binary) solution variable per I or Q dimension. First consider a transform $\mathbf{T}$ for the simplest multi-level 1-D constellation:



. $\mathbf{T} = 4q_1 + 2q_2 - 3$ maps these bits to the values −3, −1, +1, +3. Now to generalize this to 2-D, let the first two arguments of $\mathbf{T}$, q_{4i-3}, q_{4i-2}, represent the I part and the next two, q_{4i-1}, q_{4i} represent the Q part. We call this transform, shown in Fig. 2(a), the 16-QAM *QuAMax transform.* It has the desirable property that it maps solution variables to symbols linearly, *viz.* $v_i = \mathbf{T}(\mathbf{q_i}) = (4q_{4i-3} + 2q_{4i-2} - 3) + j(4q_{4i-1} + 2q_{4i} - 3)$, thus results in a QUBO form.

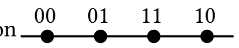


(a) QuAMax transform **(b) Intermediate code** **(c) Differential bit encoding** **(d) Gray code**

Figure 2: QuAMax's bitwise post-translation for 16-QAM (64-QAM and higher-order modulations follow an analogous translation).

However, transmitters in practical wireless communication systems use a different bit-to-symbol mapping, the *Gray code* shown in Fig. 2(d), which minimizes bit errors. This means that the QuAMax receiver's bit to symbol mapping differs from the sender's. Thus one further step remains so that we may map the decoded QUBO variables into the correct Gray-coded transmitted bits.

A naïve approach is simply for QuAMax to use the Gray-coded bit-to-symbol mapping as its transform **T**. The Gray-coded mapping results in a one-dimensional *4-PAM* constellation



assuming bits 00, 01, 11, and 10 are transformed to −3, −1, +1, and +3 without loss of generality. The transform $v_i^I = 2(2q_{4i-3} - 1) + 2(q_{4i-3} - q_{4i-2})^2 - 1$ would map between a 4-PAM symbol v_i^I and two QUBO variables q_{4i-3}, q_{4i-2}, but the resulting expansion of the ML norm would yield cubic and quartic terms $q_r q_k q_l (q_p)$ for $r \neq k \neq l (\neq p)$, requiring quadratization with additional variables to represent the problem in QUBO form [8, 33].

Instead, we retain Gray coding at the transmitter and the QuAMax transform at the receiver. To correct the disparity, we develop a *bitwise post-translation* that operates on QuAMax-transformed solution output bits at the receiver, translating them back into Gray-coded bits (*i.e.*, moving from Fig. 2(a) to Fig. 2(d)). Starting with the QuAMax transform shown in Figure 2(a), if the second bit $\hat{q}_{4i-2}$ of the QUBO solution bits $\hat{q}_{4i-3}$, $\hat{q}_{4i-2}$, $\hat{q}_{4i-1}$, $\hat{q}_{4i}$ is `1`, then the translation flips the third bit $\hat{q}_{4i-1}$ and the fourth bit $\hat{q}_{4i}$ (*e.g.* `1100` to `1111`), otherwise it does nothing. This can be generalized to 2^{2n}-QAM ($n \geq 2$) as an operation that flips even numbered columns in the constellation upside down. We term the result b' an *intermediate code*, shown in Figure 2(b). Next, we apply the differential bit encoding transformation of Figure 2(c) to the intermediate code b' to obtain the Gray-coded bits $\hat{b}$ in Figure 2(d) (*e.g.* translating `1111` to `1000`).

QuAMax decoding example. To clarify processing across all stages, here we present a complete QuAMax decoding example. Suppose a client maps a bit string b_1, b_2, b_3, b_4 onto $\bar{v}_1$, one of the Gray-coded 16-QAM symbols in Figure 2(d), and sends $\bar{\mathbf{v}} = [\bar{v}_1]$ to an AP through wireless channel **H**. The AP receives $\mathbf{y} = \mathbf{H}\bar{\mathbf{v}} + \mathbf{n}$, the transmitted signal perturbed by AWGN. The steps of QuAMax's decoding are:

(1) Form the ML QUBO equation using **H**, **y**, and $\mathbf{v} = [v_1] = [\mathbf{T}(\mathbf{q_1})]$, where $\mathbf{T}(\mathbf{q_1}) = (4q_1 + 2q_2 - 3) + j(4q_3 + 2q_4 - 3)$, a linear transform based on the QuAMax transform in Figure 2(a).
(2) Solve the QUBO form of the ML detection problem on the QA machine, resulting an ML-decoded vector $\hat{\mathbf{q}}_1$, comprised of QUBO variables $\hat{q}_1, \hat{q}_2, \hat{q}_3, \hat{q}_4$.
(3) Apply the above bitwise translation from the decoded QUBO solution output $\hat{q}_1, \hat{q}_2, \hat{q}_3, \hat{q}_4$ to Gray-coded received bits $\hat{b}_1, \hat{b}_2, \hat{b}_3, \hat{b}_4$ (from Figure 2(a) to Figure 2(d)).

If $\hat{b}_1, \hat{b}_2, \hat{b}_3, \hat{b}_4 = b_1, b_2, b_3, b_4$, decoding is done successfully, noting that in the case of a symbol error, we preserve the aforementioned advantage of Gray coding.

3.2.2 ML-to-Ising problem reduction. The Ising spin glass form of the ML problem can be obtained by simply transforming the resulting QUBO form (§3.2.1) into the Ising form by Eq. 4. Due to the fact that DW2Q implements an Ising model, QuAMax works by using the following generalized Ising model parameters:

BPSK modulation. Given a channel matrix **H** and vector of received signals **y**, we obtain the following Ising model parameters:

$$
\begin{aligned}
f_i(\mathbf{H}, \mathbf{y}) &= -2\left(\mathbf{H}^I_{(:,i)} \cdot \mathbf{y}^I\right) - 2\left(\mathbf{H}^Q_{(:,i)} \cdot \mathbf{y}^Q\right), \\
g_{ij}(\mathbf{H}) &= 2\left(\mathbf{H}^I_{(:,i)} \cdot \mathbf{H}^I_{(:,j)}\right) + 2\left(\mathbf{H}^Q_{(:,i)} \cdot \mathbf{H}^Q_{(:,j)}\right),
\end{aligned}
\tag{6}
$$

where $\mathbf{H}_{(:,i)}$ denotes the i^{th} column of channel matrix **H**.

QPSK modulation. In the case of QPSK, the following is the resulting Ising parameter f_i for QPSK:

$$
f_i(\mathbf{H}, \mathbf{y}) = \begin{cases}
\text{if } i = 2n, \\
-2\left(\mathbf{H}^I_{(:,i/2)} \cdot \mathbf{y}^Q\right) + 2\left(\mathbf{H}^Q_{(:,i/2)} \cdot \mathbf{y}^I\right), \\
\text{otherwise}, \\
-2\left(\mathbf{H}^I_{(:,\lceil i/2 \rceil)} \cdot \mathbf{y}^I\right) - 2\left(\mathbf{H}^Q_{(:,\lceil i/2 \rceil)} \cdot \mathbf{y}^Q\right).
\end{cases}
\tag{7}
$$

Since the real and imaginary terms of each symbol are independent, the coupler strength between s_{2n-1} and s_{2n} (or q_{2n-1} and q_{2n}) is 0. For other s_i and s_j, the Ising coupler strength for QPSK is:

$$
g_{ij}(\mathbf{H}) = \begin{cases}
\text{if } i + j = 2n, \\
2\left(\mathbf{H}^I_{(:,\lceil i/2 \rceil)} \cdot \mathbf{H}^I_{(:,\lceil j/2 \rceil)}\right) + 2\left(\mathbf{H}^Q_{(:,\lceil i/2 \rceil)} \cdot \mathbf{H}^Q_{(:,\lceil j/2 \rceil)}\right), \\
\text{otherwise}, \\
\pm 2\left(\mathbf{H}^I_{(:,\lceil i/2 \rceil)} \cdot \mathbf{H}^Q_{(:,\lceil j/2 \rceil)}\right) \mp 2\left(\mathbf{H}^I_{(:,\lceil j/2 \rceil)} \cdot \mathbf{H}^Q_{(:,\lceil i/2 \rceil)}\right),
\end{cases}
\tag{8}
$$

where $i < j$ and the sign of the latter case of Eq. 8 is determined by whether $i = 2n$ (when $i = 2n$, then '+' and '−').

16-QAM modulation. Ising parameters follow the same structure as BPSK and QPSK and can be found in Appendix C.

In summary, the process to obtain the Ising spin glass form can be simplified with these generalized Ising model parameters; a QuAMax system simply inserts the given channel **H** and received signal **y** at the receiver into these generalized forms accordingly, not requiring any computationally expensive operations (*i.e.* directly considering the expansion of the norm in Eq. 5). Thus, computational time and resources required for ML-to-QA problem conversion are insignificant and can be neglected.

Table 2: Logical (physical) number of qubits required for various configurations of the elementary adiabatic quantum ML decoder. For each configuration, **bold font** indicates non-feasibility on the current (2,031 physical qubit) D-Wave machine with Chimera connectivity.

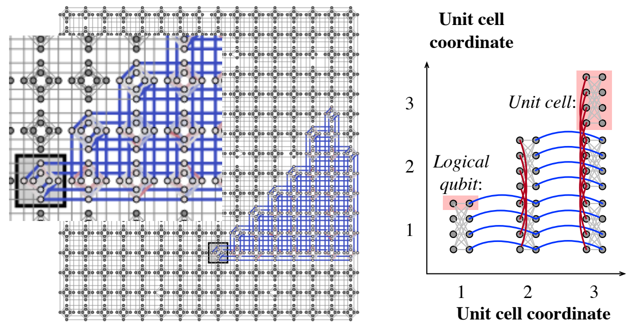
Config.	BPSK	QPSK	16-QAM	64-QAM
10×10	10 (40)	20 (120)	40 (440)	60 ($1K$)
20×20	20 (120)	40 (440)	80 ($2K$)	**120 (4K)**
40×40	40 (440)	80 ($2K$)	**160 (7K)**	**240 (15K)**
60×60	60 ($1K$)	**120 (4K)**	**240 (15K)**	**360 (33K)**

3.3 Embedding into QA hardware

Once the ML detection problem is in quadratic form, we still have to compile the corresponding Ising model onto actual QA hardware. The D-Wave machine works by implementing an Ising model objective function energetically hardcoded into the chip, so the problem (Eq. 2 on p. 4) can support a certain coefficient g_{ij} to be non-zero only if variables s_i and s_j are associated to physical variables (*qubits* or *physical qubits*) located on the chip in such a way that the qubits are energetically coupled. In the case of the DW2Q machine we use the coupling matrix is a *Chimera graph*, shown in Figure 3(a), with each node corresponding to a qubit. Once Ising coefficients are passed to the annealer, the hardware assigns them to the edges of the Chimera graph, which are divided (along with their connected nodes) into *unit cells*. Note however that, while the Ising problem generated from Eq. 1 is almost *fully connected* (*i.e.*, $g_{ij} \neq 0$ for most (i, j) pairs), the Chimera graph itself has far from full connectivity, and so a process of *embedding* the Ising problem into the Chimera graph is required.

One standard method of embedding is to "clone" variables in such a way that a binary variable becomes associated not to a single qubit but to a connected linear chain of qubits instead: a *logical qubit*, as shown in Figure 3(b).[3] We show an embedding of a fully-connected graph of 12 nodes. Each unit cell on the diagonal holds four logical qubits (a chain of two qubits), while the other unit cells are employed in order to inter-connect the diagonal cells. Specifically, suppose unit cell $[1, 1]$ includes logical qubits 1–4 and unit cell $[2, 2]$ includes logical qubits 5–8. The left side of unit cell $[2, 1]$ has a vertical clone of qubits 5–8 and the right side has a horizontal clone of logical qubits 1–4. Then, logical qubits 1–4 and 5–8 are all connected by means of the single unit cell $[2, 1]$. The unit cell hosting the next four logical qubits 9–12 is placed at coordinates $[3, 3]$. Two unit cells below, $[3, 1]$ and $[3, 2]$, are used for connections between 9–12 and 1–4, and 9–12 and 5–8 respectively.

[3]The optimal assignment problem, in the general case, is equivalent to the NP-Hard "minor embedding" problem of graph theory [13], however for fully-connected graphs very efficient embeddings are known [7, 39, 69].



(a) DW2Q qubit connections: A 32×32 BPSK problem is shown embedded in the chip's substrate.

(b) Logical qubits and unit cells in the QuAMax decoder.

Figure 3: A comparison between the quantum hardware graph of the used machine (which misses some nodes due to manufacturing defect), and the topology of our elementary quantum ML hardware graph before embedding into the hardware graph.

Given a number N of spin variables (*i.e.*, logical qubits) in Ising form, this embedding represents each with a chain of $\lceil N/4 \rceil + 1$ qubits, for a total of $N (\lceil N/4 \rceil + 1)$ qubits. Recall that $N = N_t \cdot \log_2(|O|)$.

Table 2 summarizes the size of the embedding in both logical and physical qubits, as a function of the MIMO detection problem's parameters—number of users and AP antennas, and modulation type. Color coding and bold font indicate whether or not the given parameters fit into the number of qubits available on current D-Wave machines.

The embedded version of the Ising problem. After embedding into Chimera graph we need to recast the Ising problem into an equivalent problem that has the same ground state, but also satisfies the Chimera graph constraints. We also need to introduce a constant penalty term (J_F) to quantify the relatively large coupling that constrains all physical qubits belonging to the same logical qubit to prefer the same state. Appendix B contains additional detail, but we discuss important experimental considerations for choosing J_F in Section 5.3.

Unembedding with majority voting. The bit string that the DW2Q returns is expressed in terms of the embedded Ising problem, and so must be *unembedded* in order to have the values of the bits expressed in terms of our ML Ising problem. This is done by checking that all the qubits of a logical chain are either $+1$ or -1. Should not all spins be concordant, the value of the corresponding logical variable is obtained by *majority voting* (in case of a vote tie, the value is randomized). Once the logical variables are determined, each configuration yields the corresponding energy of the Ising objective function by substituting it into the original Ising spin glass equation (Eq. 2).

4 IMPLEMENTATION

This section describes our implementation on the D-Wave 2000Q quantum annealer (DW2Q), explaining the API between the annealer's control plane and its quantum substrate, machine parameters, and their tuning to the problem at hand.

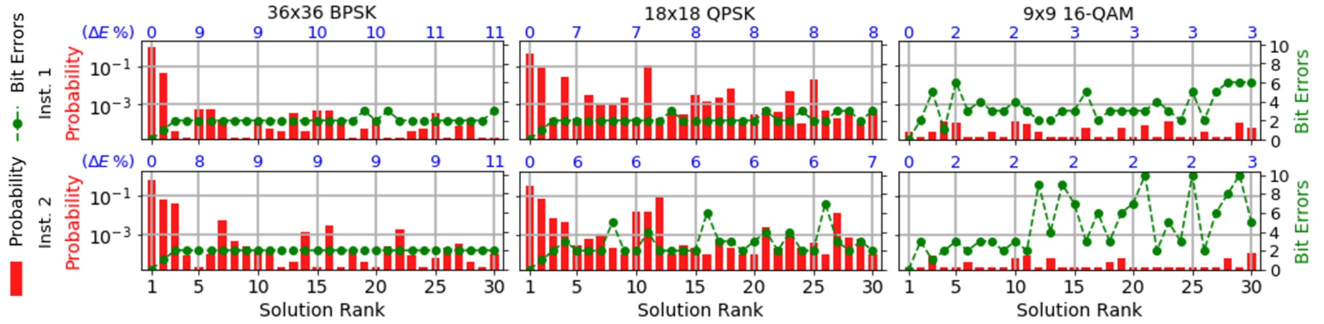


Figure 4: Empirical QA results from six different decoding problems, illustrating relationships between Ising energy, solution rank, BER.

Each *anneal cycle* on the DW2Q yields a configuration of spins (*i.e.,* one decoded bit string). The user programs the annealer to run a batch of N_a annealing cycles (one *QA run*) with the same parameters to accumulate statistics, which means that we have a set of N_a configurations from a DW2Q job submission. The lowest energy configuration among N_a anneals is the best answer found.

Parallelization. Multiple instances (identical or not) can be run physically alongside each other, reducing run time by the *parallelization factor*[4] $P_f \simeq N_{tot}/(N(\lceil N/4\rceil + 1))$—a small 16-qubit problem employing just 80 physical qubits (*e.g.* 16-user BPSK, 8-user QPSK, and 4-user 16-QAM) could in fact be run more than 20 times in parallel on the DW2Q.

Precision Issues. As analog devices, the desired embedded Ising coefficients (Eqs. 10-12 in Appendix B) do not perfectly match their real energy values once hardcoded in the real machine, and hence give rise to *intrinsic control errors* (ICE), an uncontrollable shift in the actual programmed values of the objective function. ICE is appropriately modeled as a noise fluctuating at a time scale of the order of the anneal time, *i.e.*, on each anneal, Ising coefficients are perturbed: $\mathbf{f_i} \longrightarrow \mathbf{f_i} + \langle \delta \mathbf{f_i} \rangle$, $\mathbf{g_{ij}} \longrightarrow \mathbf{g_{ij}} + \langle \delta \mathbf{g_{ij}} \rangle$. where the noise is Gaussian with mean and variance measured $\langle \delta \mathbf{f_i} \rangle \simeq 0.008 \pm 0.02$ and $\langle \delta \mathbf{g_{ij}} \rangle \simeq -0.015 \pm 0.025$ in the most delicate phase of the annealing run [16]. The impact of ICE on performance depends on the problem at hand [12, 78], but it is clear that precision issues will arise if the largest energy scale squeezes the value of the coefficients (in Eqs. 11–12 in Appendix B) to a level where ICE is likely to erase significant information of the problem's ground state configuration.

Annealer Parameter Setting. As discussed in Section 3.3, the $|J_F|$ that enforces a chain of qubits to return a series of values which are all in agreement (either all +1 or −1), and the annealing time T_a are both key performance parameters that determine the net time to find a solution, and hence overall QA performance. We also introduce 1, 10, and 100 μs *pause time* T_p in the middle of annealing (T_a = 1 μs) with various *pause positions* s_p, to see the effect of pausing [43] on our problems. Setting $|J_F|$ too large would wash out the problem information due to ICE, however $|J_F|$ on average should increase with the number of logical chains in fully-connected problems in the absence of ICE [69]. Due to the lack of a first-principles predictive theory on the correct value for a given instance, we present in Section 5.3 an empirical investigation of the best embedding parameters, employing the microbenchmarking methodologies generally accepted [52, 57, 69]. Below we perform a sensitivity analysis on J_F, T_a, T_p, and s_p (§5.3.1) over the ranges $J_F \in \{1.0 - 10.0\ (0.5)\}$, $T_a \in \{1, 10, 100\ \mu s\}$, $T_p \in \{1, 10, 100\ \mu s\}$, and $s_p \in \{0.15 - 0.55\ (0.02)\}$.

[4]While asymptotically the parallelization factor is just the ratio of used physical qubits after embedding to the number of qubits in the chip N_{tot}, in finite-size chips, chip geometry comes into play.

Improved coupling dynamic range. The *dynamic range* of coupler strengths is defined as the ratio between the maximum and minimum values that can be set (g_{ij} in Eq. 2). To strengthen interactions between embedded qubits, the DW2Q is able to double the magnitude of valid negative coupler values, effectively increasing the precision of embedded problems and reducing ICE. However, this *improved range* option, when enabled, breaks the symmetry of the Ising objective function (substituting the opposite signs for connected coefficients and their couplings, into the same problem), and hence precludes averaging over these symmetrical instances as the DW2Q does without the improved range option, to mitigate leakage errors [4]. It is thus unclear whether the use of this feature is beneficial in the end without experimentation, and so we benchmark in Section 5 both with and without improved range.

5 EVALUATION

We evaluate QuAMax on the DW2Q Quantum Annealer machine shown in Figure 1. We consider the same number of antennas at the clients and AP (*i.e.*, $N_t = N_r$). Section 5.1 introduces QA results, while Section 5.2 explains our experimental methodology. After that in Section 5.3 we present results under only the presence of the annealer's internal thermal noise (ICE). Sections 5.4 and 5.5 add wireless AWGN channel noise and trace-based real-world wireless channels, respectively, quantifying their interactions with ICE on end-to-end performance. Over 8×10^{10} anneals are used in our performance evaluation.

5.1 Understanding Empirical QA Results

We begin with a close look at two runs of the QA machine, to clarify the relationships between Ising energy, Ising energy-ranked solution order, and BER. Figure 4 shows six QA problem instances (all of which require 36 logical qubits), corresponding to two different wireless channel uses for each of varying modulation and number of users. The solutions are sorted (ranked) by their *relative Ising energy difference* ΔE from the minimum Ising energy (blue numbers atop selected solutions), where red bars show each solution's frequency of occurrence (in the rare case of two or more tied distinct solutions, we split those solutions into multiple solution ranks). The number of bit errors associated with each solution appears as a green curve. For statistical significance, this data summarizes 50,000 anneals, more than QuAMax's practical operation. As modulation order increases and number of users decreases (from left to right in Figure 4), the probability of finding the ground state tends to be lower, while the search space size remains constant, leading eventually to higher BER and FER.[5] The relative Ising energy gap also trends smaller,[6] and is likely to be inversely correlated with the impact of ICE on the problem instance [2, 78].

5.2 Experimental methodology

In this section, we introduce performance metrics and figures of merit that give insight into the operation of the QA machine as it solves the ML MIMO decoding problem.

We note that in our performance evaluation we exclude from consideration programming time and post-programming time of the Ising coefficients on the chip, and readout latency of the qubit states after a single anneal. Currently, these times dominate the pure computation time (*i.e.* total anneal time) by several orders of magnitude (milliseconds), due to engineering limitations of the technology. However, these overheads do not scale with problem size and are not fundamental performance factors of the fully integrated QuAMax system, and so this is in accordance with experimental QA literature convention. We discuss these overheads in Section 7.

5.2.1 Metric: Time-to-Solution (TTS). Suppose we find the ground state (corresponding to the minimum energy solution within the search space of 2^N bit strings, where N is the variable count) with probability $\mathcal{P}_0$. In the absence (but not presence) of channel noise, this ground state corresponds to a correct decoding. Each anneal is an independent, identically-distributed random process, meaning that the expected *time to solution*, or $TTS(\mathcal{P})$, is the anneal time of each anneal T_a multiplied by the expected number of samples to be able to find the ML solution with probability $\mathcal{P}$: $TTS(\mathcal{P}) = T_a \cdot \log(1-\mathcal{P})/\log(1-\mathcal{P}_0)$. TTS is commonly used in the QA literature, setting $\mathcal{P} = 0.99$ [58].

5.2.2 Our Metrics: BER and Time-to-BER (TTB). TTS reflects the expected time to find the ground state, but does not characterize the expected time our system takes to achieve a certain *Bit Error Rate* (*BER*, averaged across users), the figure of merit at the physical layer. This quantity differs from TTS, because TTS only considers the ground state, and as illustrated in the example shown in Figure 4, solutions with energy greater than the ground state may also have (rarely) no or relatively few bit errors, while wireless channel noise may induce bit errors in the ground state solution itself. Hence we introduce a metric to characterize the time required to obtain a certain BER p, *Time-to-BER*: *TTB(p)*. This is preferred in our setting, since a low but non-zero bit error rate is acceptable (error control coding operates above MIMO detection).

TTB for a single channel use. Since one QA run includes multiple (N_a) anneals, we return the annealing solution with minimum energy among all anneals in that run. We show an example of this process for one *instance* (*i.e.*, channel use, comprised of certain transmitted bits and a certain wireless channel) in Fig. 4. The annealer finds different solutions, with different Ising energies, ranking them in order of their energy. Considering this order statistic, and the fact that QuAMax considers only the best solution found by all the anneals in a run, the *expected* BER of instance I after N_a anneals can be expressed as

$$\mathbb{E}(BER(N_a)) = \sum_{k=1}^{L}\left[\left(\sum_{r=k}^{L} p_I(r)\right)^{N_a} - \left(\sum_{r=k+1}^{L} p_I(r)\right)^{N_a}\right] \cdot F_I(k)/N, \quad (9)$$

where N is qubit count, L ($\leq N_a$) is the number of distinct solutions, r ($1 \leq r \leq L$) is the rank index of each solution, $p(r)$ is the probability of obtaining the r^{th} solution, and $F_I(k)$ is the number of bit errors of the k^{th} solution against ground truth.[7] To calculate TTB(p), we replace the left hand side of Eq. 9 with p, solve for N_a, and compute TTB(p) = $N_a T_a / P_f$.

Generalizing to multiple channel uses. The preceding predicts TTB for a fixed instance. In the following study we compute TTB and BER across multiple instances (random transmitted bits and randomly-selected wireless channel), reporting statistics on the resulting sampled distributions.

5.3 Performance Under Annealer Noise

This section presents results from the DW2Q annealer for wireless channel noise-free scenarios, in order to characterize the machine's performance itself as a function of time spent computing. Sections 5.4 and 5.5 experiment with Gaussian noise and trace-based wireless channels, respectively.

In this section, we run several instances with unit fixed channel gain and average transmitted power. Each instance has a random-phase channel, randomly chosen transmitted bit string, and is repeated for each of three different modulations (BPSK, QPSK, 16-QAM) and varying numbers of users and AP antennas. Each instance is reduced to Ising as described in Section 3.2, for a total of 780 different problems per QA parameter setting. Unless otherwise specified, this and subsequent sections use the fixed parameter settings defined in §5.3.1. We obtain significant statistics by post-processing up to 50,000 anneals per QA run (except 10,000 anneals for anneal pause analysis in Figure 7).

5.3.1 Choosing Annealer Parameters. In order to isolate the effect of different parameter settings on individual problems, we employ microbenchmarks on TTS. This section explains our choice of parameter settings for our main performance results in §5.3.3, §5.4, and §5.5. Note that while we plot results here only for BPSK and QPSK to save space, our results show that the methods, arguments and observations generalize to higher modulations, unless otherwise indicated. For the purpose of setting the parameters, we restrict the dataset to the ML problems that solve within a median TTS(0.99) of 10 ms for which we have low uncertainty on the measured success probability. We use the determined parameters for all instances regardless of their TTS for the performance analysis.

[5]See section 5.2.2. Frame error rate FER is computed as $1-(1-BER)^{\text{frame size}}$.
[6]The energy distribution of the Ising objective function (Eq. 2) corresponds to the distribution of ML decoder Euclidean distances (Eq. 1).

[7]Note that the metric has omniscient knowledge of ground truth transmitted bits, while the machine does not.

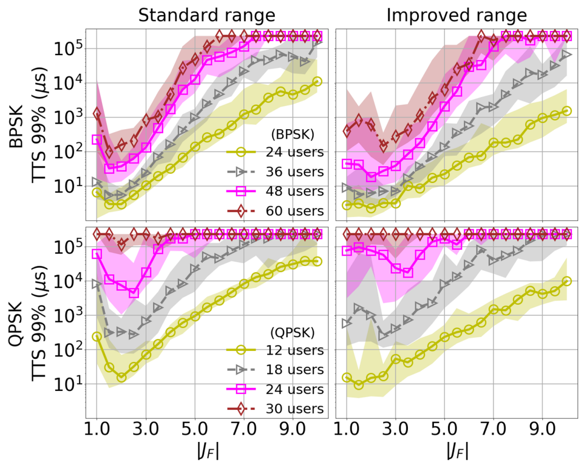


Figure 5: Time-to-Solution comparison of different strengths of ferromagnetic coupling within logical qubits, $|J_F|$. *Upper:* BPSK, *lower:* QPSK, *left:* standard range, *right:* improved range; results obtained for $T_a = 1\ \mu s$. Lines report median of 10 instances; shading reports 10th. and 90th. percentiles.

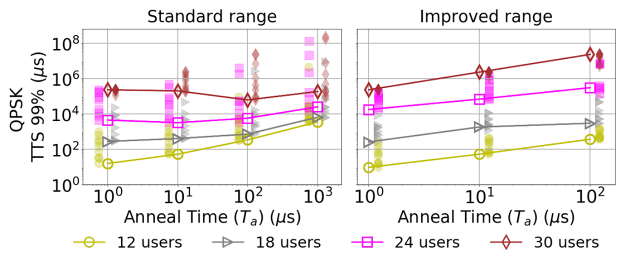


Figure 6: TTS analysis of different anneal times for different numbers of users, for QPSK. Scatter points correlate median results obtained for different $|J_F|$, while lines highlight the best $|J_F|$ measured from Fig. 5 and reporting the median across 10 random instances.

Ferromagnetic couplings: We examine median TTS(0.99) versus $|J_F|$ over 10 random instances of different sizes both with and without extended dynamic range. In Fig. 5, we observe that while there is a performance optimum that depends on the problem size for standard dynamic range, extended dynamic range shows less sensitivity to $|J_F|$, obtaining roughly the optimal $|J_F|$ performance of standard dynamic range. **Anneal time:** As we vary T_a, we observe greater sensitivity when we use non-optimal J_F, as the scatter points next to each data point in Fig. 6 (*left*) show. On the other hand, Fig. 6 (*right*) shows that an extended dynamic range setting achieves best results at $T_a = 1\ \mu s$ regardless of problem size, showing less sensitivity to different $|J_F|$. **Anneal Pause Time and Location:** When we apply improved dynamic range at $T_a = 1\ \mu s$, we observe a slight independence (Fig. 7) of s_p and J_f on T_p, and as T_p increases, so does TTS. While the dynamic range setting has shown less sensitivity to $|J_F|$, anneal pause with extended dynamic range shows more sensitivity. Note that TTS of 18-user QPSK at $T_p = 1\ \mu s$ is slightly improved, compared to the best results in Figs. 5 and 6.

Annealer Parameter Optimization. Based on the previous sensitivity analysis, we select a default QA parameter setting. First, we choose **improved dynamic range** since it is relatively robust to

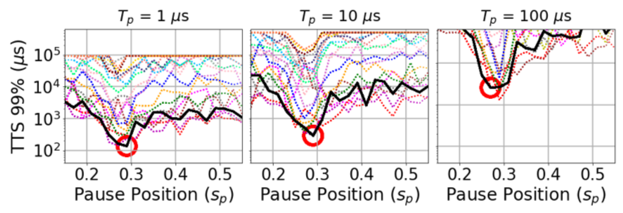


Figure 7: TTS analysis of anneal pause time and position for 18-user QPSK. Colored dotted lines join results obtained for different $|J_F|$, while the thicker black line highlights the best $|J_F|$ measured from Fig 5. Lines report the median across 10 random instances. The red circle indicates the best s_p for chosen $|J_f|$.

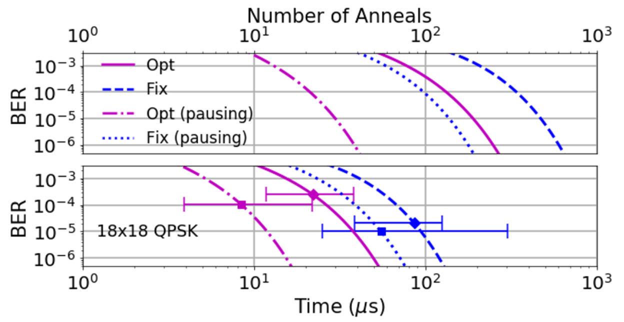


Figure 8: BER of different optimization settings as a function of the number of anneals (*upper*) and time (*lower*) for 18 × 18 QPSK (median across 20 instances). Error bars indicate 15th. and 85th. percentiles.

choice of $|J_F|$, nearly equaling the best performance of the standard dynamic range. Second, we choose $\mathbf{T_p = 1\ \mu s}$, since it shows better results and greater pause times dominate the anneal time.

5.3.2 *Choosing whether to pause.*

With the above default QA parameters, we now use TTB to explore whether or not we should use the QA pause functionality, as TTB encompasses both algorithms' BER performance as well as wall clock running time (*cf.* TTS). We first define a *fixed parameter setting* by selecting the best estimated choices for the non-pausing algorithm and for the pausing algorithm, meaning the parameters which optimize medians across a sample of instances belonging to the same problem class (*e.g.* 18×18 QPSK). This approach is to be compared against an *oracle* that optimizes $\{J_F, T_a\}$ or $\{J_F, s_p\}$ instance by instance. In the figures, we denote the two parameter setting methods as ***Fix*** (fixed) and ***Opt*** (optimal), respectively.

Our motivation for considering *Opt* is that it provides a bound to what can be achieved by the methods that seek to optimize machine parameter settings instance by instance [68, 70], currently under investigation. With our traces we compute BER as a function of N_a using Eq. 9; the median result across 20 random instances is shown in Figure 8 (*upper*). Figure 8 (*lower*) shows the corresponding BER as a function of time (*i.e.*, TTB). Note that the pausing algorithm has a better performance than the non-pausing algorithm (regardless of *Opt* and *Fix* strategy) despite the fact that each anneal in the former ($T_a + T_p$) takes twice as much time than the latter (when $T_a = 1\ \mu s$). Based on this empirical finding, we will present the results in the following section only for the protocol that includes a pause.

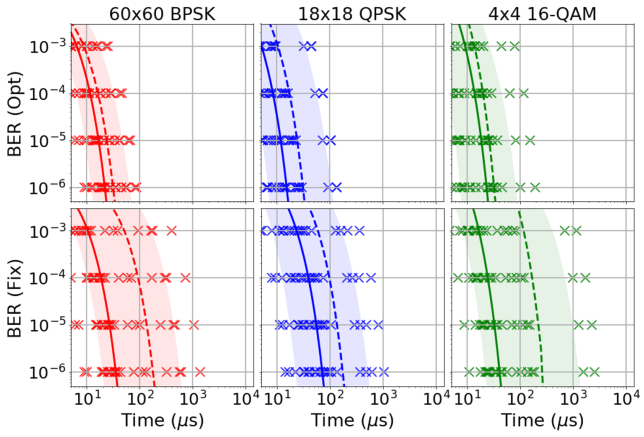


Figure 9: Time-to-BER (TTB) comparison across different user numbers and modulations. *Upper*: ideal scheme using *Opt*. *Lower*: QuAMax's performance optimizing with *Fix*. Solid lines and dashed lines report median and mean TTB across 20 instances, respectively. Shading reports 10th. and 90th. percentiles of average BER at a certain time and each × symbol reports each instance's TTB (x-value) for a certain target BER.

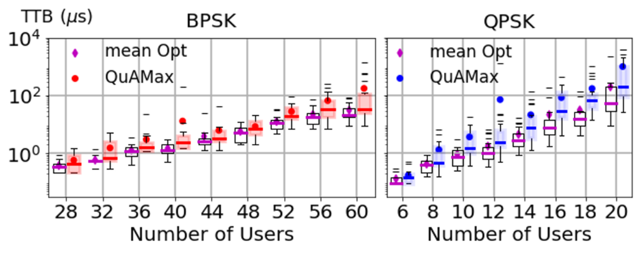


Figure 10: TTB with target BER 10^{-6} for different modulations and user numbers across 20 instances. Colored boxes report QuAMax (5th., 95th. as whiskers, upper/lower quartiles as boxes, median as the thick horizontal mark, and thin horizontal marks for outliers.)

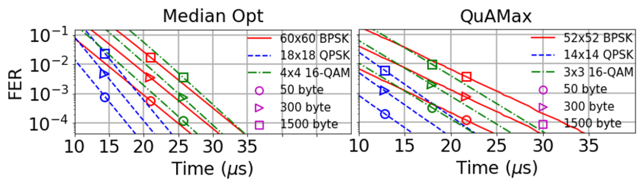


Figure 11: Time-to-FER for different users, modulations, and frame sizes; *left*: median Opt (idealized), *right*: mean *Fix* (QuAMax).

5.3.3 QuAMax: End-to-End performance. We now evaluate the TTB and TTF (Time-to-FER) of QuAMax, comparing:

(1) **QuAMax:** Fixed-parameter, average-case performance.
(2) **Oracle:** Median-case *Opt* performance (§5.3.2: outlier data points have minimal influence on the median order statistic), optimizing QA parameters.[8]

Figure 9 shows the TTB with varying user numbers and modulations at the edge of QuAMax's performance capabilities. Solid and dashed lines report median and average BER, respectively. We note that mean TTB dominates median TTB due to a small number of long-running outliers. QuAMax accordingly sets a time deadline (measured median TTB for the target BER) for decoding and after that discards bits, relying on forward error correction to drive BER down. Next considering the relationship between TTB and problem size, Fig. 10 explores TTB for target BER 10^{-6}, for each instance that reaches a BER of 10^{-6} within 10 ms as well as average performance. ML problems of these sizes are well beyond the capability of conventional decoders (*cf.* Table 1), and we observe that *Opt* achieves superior BER within 1–100 μs and that QuAMax achieves an acceptable BER for use below error control coding. Note that instances with TTB below the minimum required time (*i.e.*, $T_a + T_p$) caused by parallelization require (an amortized) 2 μs.

Next, we consider frame error rate performance, measuring mean and median FER QuAMax achieves. Results in Fig. 11 show that tens of microseconds suffice to achieve a low enough (below 10^{-3}) FER to support high throughput communication for 60-user BPSK, 18-user QPSK, or four-user 16-QAM suffices to serve four users with the idealized median performance of *Opt*. QuAMax (mean *Fix*) achieves a similar performance with slightly smaller numbers of users. Furthermore, our results show low sensitivity to frame size, considering maximal-sized internet data frames (1,500 bytes) all the way down to TCP ACK-sized data frames (50 bytes).

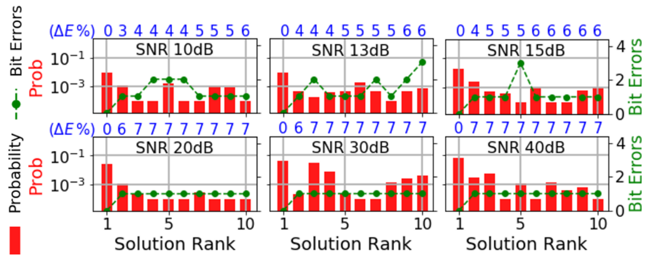


Figure 12: Detailed view (*cf.* Fig. 4) of an example wireless channel at six different SNRs (18-user QPSK).

5.4 Performance under AWGN Noise

We next evaluate the impact of AWGN from the wireless channel, testing six SNRs ranging from 10 dB to 40 dB. In order to isolate the effect of noise, the results in this subsection fix the channel and transmitted bit-string and consider ten AWGN noise instances. Looking at the data in depth to begin with, the effect of AWGN channel noise, which is itself additive to ICE, is shown in Fig. 12 for six illustrative examples. As SNR increases, the probability of finding the ground state and the relative energy gap tend to increase. At 10 dB SNR the energy gap between the lowest and second lowest energy solutions narrows to just three percent, leaving minimal room for error. In terms of overall performance, Fig. 13 (*left*) shows TTB at 20 dB SNR, varying number of users and modulation. At a fixed SNR, we observe a graceful degradation in TTB as the number of users increases, across all modulations. Fig. 13 (*right*) shows TTB at a certain user number, varying SNR and modulation. At a fixed user number, as SNR increases, performance improves, noting that the idealized median performance of *Opt* shows little sensitivity to SNR, achieving 10^{-6} BER within 100 μs in all cases.

[8]Outlier mitigation methods for QA may address such outliers in future work [40].

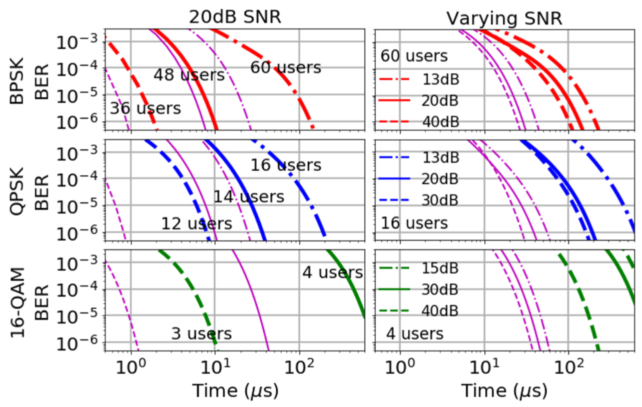


Figure 13: TTB comparison across different user numbers, modulations, and SNRs. *Left*: varying the number of users at SNR 20 dB. *Right*: varying SNR at a certain number of users. Thick lines report QuAMax's performance (mean *Fix*), and same style but thin (purple) lines report the idealized performance (median *Opt*).

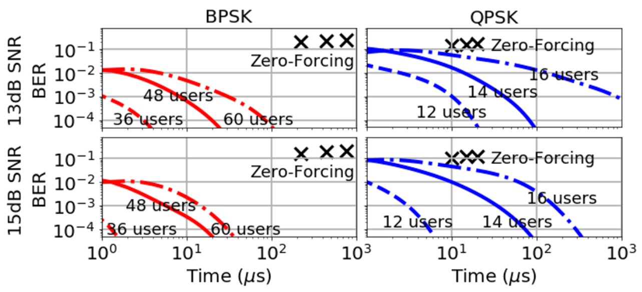


Figure 14: QuAMax's performance comparison against the zero-forcing decoder across different user numbers, modulations, and SNRs. Each × symbol (*left-to-right:* 36, 48, 60 users for BPSK and 12, 14, 16 users for QPSK) reports the zero-forcing decoder's BER and corresponding processing time.

Fig. 14 compares QuAMax's performance versus zero-forcing decoder at bad SNR scenarios, showing the necessity of ML-based MIMO decoders for large MIMO system. Linear decoders such as zero-forcing and MMSE suffer from the effect of the poor channel condition (when $N_t \approx N_r$), requiring $N_r > N_t$ (*i.e.,* more antennas) for appropriate BER performance. In Figure 14, QuAMax reaches the zero-forcing's BER (or even better BER) approximately 10-1000 times faster than zero-forcing in both BPSK and QPSK modulation. Here, computation times for zero-forcing are inferred from processing time using a single core in BigStation [76], one of the current large MIMO designs based on zero-forcing. While this processing time can be reduced proportionally with more cores, BER (*i.e.,* quality of solutions) remains unchanged. The Sphere Decoder achieves comparable BER, but processing time cannot fall below a few hundreds of μs with the given numbers of users and SNRs.[9]

[9]Extreme levels of parallelization or GPU implementation might be able to resolve the issue. However, practical constraints will eventually limit the increase in performance on classical platforms [38]. Contrarily, overheads in QuAMax are apart from pure computation, which can be resolved by engineering design.

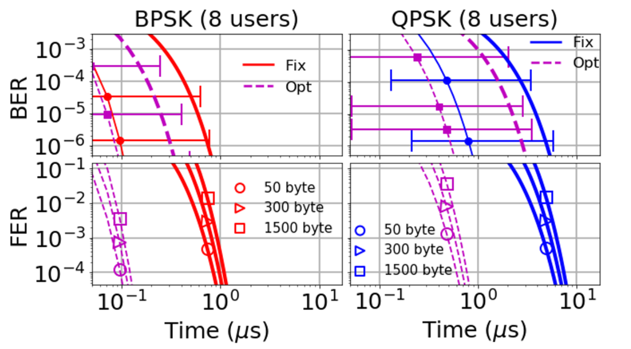


Figure 15: Experimentally measured channel trace [61] results: upper plots report TTB (*Opt*, *Fix*); lower plots report TTF of median *Opt* and mean *Fix* (QuAMax). Thin and thick lines report median and mean, respectively. TTB's error bars indicate 15th. and 85th. percentiles.

5.5 Trace-Driven Channel Performance

We evaluate system performance with real wideband MIMO channel traces at 2.4 GHz, between 96 base station antennas and eight static users [61]. This dataset comprises the largest MIMO trace size currently available. For each channel use, we randomly pick eight base station antennas to evaluate the 8×8 MIMO channel use at SNR *ca.* 25–35 dB: Fig. 15 shows the resulting TTB and TTF. We achieve 10^{-6} BER and 10^{-4} FER within 10 μs for QPSK. For BPSK, considering multiple problem instances operating in parallel, we achieve the same BER and FER within (an amortized) 2 μs period. This implies that tens of microseconds suffice to achieve a low BER and FER even without parallelization of identical problems, which creates an opportunity for QuAMax to parallelize different problems (*e.g.*, different subcarriers' ML decoding).

6 RELATED WORK

Applications of QA. Despite the immaturity of software toolchains, existing quantum annealing machines have been already programmed successfully to solve problems in Planning and Scheduling [57], Databases [66], Fault Diagnostics [55], Machine Learning [54], Finance [59], Data Analysis [47], Chemistry [31], and Space Sciences/-Aeronautics [3]. A Similar problem to ML detection, CDMA multiuser demodulation, was solved using quantum fluctuations controlled by the transverse field (similar as QA) in [53]. Of particular relevance is work on optimization of fully-connected graphs, such as the ones used to map the ML problem [69]; the results of which showed that QA performance could match the most highly optimized simulated annealing code run on the latest Intel processors. For further details on the logical to physical qubit embedding process, see Venturelli *et al.* [69]. Efficient embeddings which do not force the chip coverage to be a triangle are also known [7].

QAOA. Quantum Approximate Optimization Algorithms, invented in 2014 [23], and recently generalized for constrained combinatorial optimization [28], require digital gate-model QC, which became available at reasonable scale only in 2017 (prototypes from IBM, Rigetti Computing, and Google are available). While QA and QAOA require different hardware (the former is analog, the latter digital) they have in common that: **(1)** For problems that don't have hard constraints, the programming step consists in defining a classical combinatorial problem which is cast into QUBO [9, 72] or *Ising* form, hence they both may leverage our formulation §3.2. **(2)** QAOA can be seen in some parameter range as a "digitized" version of QA, and it has been formally demonstrated that it can simulate the results and performance of QA and outperform it, in principle [77]. The first commonality is particularly important since it opens the door to application of our techniques on future hardware capable of running QAOA.

Conventional ML Detectors. Faster silicon based ML detector strategies typically approximate and parallelize the ML computation [15, 73]. In these general directions, much progress has been made to the point that Sphere Decoders have been realized in ASIC hardware [10, 74] but fall short for the reasons noted in Table 1 when the setting demands more antennas at the AP (serving more users), or when the modulation chosen increases [32, 76].

7 DISCUSSION

In this section, we discuss the current status of QA technology and practical considerations.

Computational Power Consumption. The computation in the DWQ2 is performed at zero energy consumption, as dictated by reversible computing, although energy is dissipated in the initialization and readout. The DW2Q draws 16 kW of power, primarily used by the cryogenic refrigeration unit [17]. The computational power (per watt) for QPUs is expected to increase much more rapidly than for conventional computing platforms since the DW2Q power draw is not expected to change much as qubit and coupler counts grow in future generation systems while the computational power substantially increases.

Operating Expenses. Operating the DW2Q results in significant electricity cost, and the dilution refrigerator requires liquid nitrogen 1-2 times a month, for a total yearly cost of about USD $17,000.

Processing Times. The scenario envisioned by QuAMax assumes a centralized RAN architecture where a QPU, co-located with centralized RAN computational resources in a data center, is connected to the APs via high-speed fiber or millimeter-wave links. In this setting, a latency between the APs and data center will not be significant. Nonetheless, QuAMax cannot be deployed today, since additional processing times in the current QPU include approximately 30-50 ms preprocessing time, 6-8 ms programming time, and 0.125 ms readout time per anneal. These overheads are well beyond the processing time available for wireless technologies (at most 3–10 ms). However, these overhead times are not of a fundamental nature and can be reduced by several orders of magnitude by efforts in system integration. By means of extrapolation of improvement trends it is expected that quantum engineering advances in superconducting qubit technology will enable QuAMax to be viable within a decade. Moreover, QuAMax's Ising form (in Section 3.2.2) can be adapted to be run in other emerging physics-based optimization devices based on photonic technologies [29] whose processing times overhead are in principle much faster. Hence, we leave an end-to-end evaluation in a fully centralized RAN architecture, with more advanced hardware, as future work.

8 CONCLUSION

QuAMax is the first design, implementation and experimental evaluation of a quantum-computing solver for the computationally challenging ML MIMO decoding problem. Our performance results establish a baseline for a future fully-integrated systems in the context of the centralized RAN architecture. We show that once engineering efforts optimize the integration between quantum and conventional computation, quantum computation should be considered a competitive technology for the future design of high-capacity wireless networks.

Future Work. There are several improvements over the design we have evaluated here. First, we anticipate that further optimization of $|J_F|$, T_a, and s_p as well as new QA techniques such as *reverse annealing* [68] may close the gap to *Opt* performance. Second, there are changes in QA architecture expected in annealers due this year [21] featuring qubits with 2× the degree of Chimera, 2× the number of qubits and with longer range couplings. Based on similar gains in recent results on different problem domains [29], we anticipate this will permit ML problems of size, *e.g.* 175×175 for QPSK and dramatically increase the parallelization opportunity of the chip due to the reduced embedding overhead where each chain now only requires $N/12 + 1$ qubits.

Going forward, we will benefit from QA technology improvements from the international community manufacturing quantum annealers with advanced capabilities. According to the development roadmap for these next-generation quantum optimizers, it is expected that in *ca.* a decade a system such as QuAMax could be based on chips with tens of thousands of highly-connected qubits, with annealing schedules capable of more advanced quantum effects (*e.g.* non-stoquasticity [51]) and engineering advances will have order-of-magnitude improvements on the aforementioned overhead operation times. While quantum annealers are ahead in terms of number of qubits, gate-model systems offer additional controls that may conceivably increase performance in the future. We will investigate MIMO ML decoding on gate-model QPUs in future work, which currently cannot support algorithms that decode more than 4×4 BPSK.

ACKNOWLEDGEMENTS

We thank our shepherd John Heidemann and the anonymous reviewers for their insightful feedback. We thank the NASA Quantum AI Laboratory (QuAIL) and D-Wave Systems for useful discussions. The research is supported by National Science Foundation (NSF) Awards #1824357 and #1824470 and by the USRA Cycle 3 Research Opportunity Program that allowed machine time on the DW2Q hosted at NASA Ames Research Center. Kyle Jamieson and Minsung Kim are partially supported by the Princeton University School of Engineering and Applied Science Innovation Fund. This work does not raise any ethical issues.

REFERENCES

[1] Erik Agrell, Thomas Eriksson, Alexander Vardy, and Kenneth Zeger. 2002. Closest point search in lattices. *IEEE transactions on information theory* 48, 8 (2002), 2201–2214.

[2] Tameem Albash, Victor Martin-Mayor, and Itay Hen. 2019. Analog errors in Ising machines. *Quantum Science and Technology* 4, 2 (2019), 02LT03.

[3] Rupak Biswas, Zhang Jiang, Kostya Kechezhi, Sergey Knysh, Salvatore Mandrà, Bryan O'Gorman, Alejandro Perdomo-Ortiz, Andre Petukhov, John Realpe-Gómez, Eleanor Rieffel, Davide Venturelli, Fedir Vasko, and Zhihui Wang. 2017. A NASA perspective on quantum computing: Opportunities and challenges. *Parallel Comput.* 64 (2017), 81–98.

[4] Sergio Boixo, Troels F Rønnow, Sergei V Isakov, Zhihui Wang, David Wecker, Daniel A Lidar, John M Martinis, and Matthias Troyer. 2014. Evidence for quantum annealing with more than one hundred qubits. *Nature Physics* 10, 3 (2014), 218.

[5] Sergio Boixo, Vadim N Smelyanskiy, Alireza Shabani, Sergei V Isakov, Mark Dykman, Vasil S Denchev, Mohammad H Amin, Anatoly Yu Smirnov, Masoud Mohseni, and Hartmut Neven. 2016. Computational multiqubit tunnelling in programmable quantum annealers. *Nature Communications* 7 (2016). http://www.nature.com/ncomms/2016/160107/ncomms10327/full/ncomms10327.html

[6] Michael Booth, Steven P Reinhardt, and Aidan Roy. 2017. Partitioning optimization problems for hybrid classical. *quantum execution. Technical Report* (2017), 01–09.

[7] Tomas Boothby, Andrew D King, and Aidan Roy. 2016. Fast clique minor generation in Chimera qubit connectivity graphs. *Quantum Information Processing* 15, 1 (2016), 495–508.

[8] Endre Boros and Aritanan Gruber. 2014. On quadratization of pseudo-Boolean functions. *arXiv preprint arXiv:1404.6538* (2014).

[9] Endre Boros, Peter L Hammer, and Gabriel Tavares. 2007. Local search heuristics for quadratic unconstrained binary optimization (QUBO). *Journal of Heuristics* 13, 2 (2007), 99–132.

[10] Andreas Burg, Moritz Borgmann, Markus Wenk, Martin Zellweger, Wolfgang Fichtner, and Helmut Bolcskei. 2005. VLSI implementation of MIMO detection using the sphere decoding algorithm. *IEEE Journal of solid-state circuits* 40, 7 (2005), 1566–1577.

[11] Biao Chen and Jianping Wang. 2002. Efficient routing and wavelength assignment for multicast in WDM networks. *IEEE Journal on Selected Areas in Communications* 20, 1 (2002), 97–109.

[12] Andrew M Childs, Edward Farhi, and John Preskill. 2001. Robustness of adiabatic quantum computation. *Physical Review A* 65, 1 (2001), 012322.

[13] Vicky Choi. 2011. Minor-embedding in adiabatic quantum computation: II. Minor-universal graph design. *Quantum Information Processing* 10, 3 (01 Jun 2011), 343–353. https://doi.org/10.1007/s11128-010-0200-3

[14] Rachel Courtland. 2016. Transistors could stop shrinking in 2021. *IEEE Spectrum* 53, 9 (2016), 9–11.

[15] Tao Cui, Shuangshuang Han, and Chintha Tellambura. 2012. Probability-distribution-based node pruning for sphere decoding. *IEEE Transactions on Vehicular Technology* 62, 4 (2012), 1586–1596.

[16] D-Wave Systems User Manual 2019. *Technical Description of the D-Wave Quantum Processing Unit.*

[17] D-Wave Systems White-paper Series 2017. *Computational Power Consumption and Speedup.*

[18] Edward D Dahl. 2013. Programming with d-wave: Map coloring problem. *D-Wave Official Whitepaper* (2013).

[19] Erik Dahlman, Stefan Parkvall, and Johan Skold. 2013. *4G: LTE/LTE-advanced for mobile broadband.* Academic press.

[20] Mohamed Oussama Damen, Hesham El Gamal, and Giuseppe Caire. 2003. On maximum-likelihood detection and the search for the closest lattice point. *IEEE Transactions on information theory* 49, 10 (2003), 2389–2402.

[21] Nike Dattani, Szilard Szalay, and Nick Chancellor. 2019. Pegasus: The second connectivity graph for large-scale quantum annealing hardware. *arXiv preprint arXiv:1901.07636* (2019).

[22] Vasil Denchev, Sergio Boixo, Sergei Isakov, Nan Ding, Ryan Babbush, Vadim Smelyanskiy, John Martinis, and Hartmut Neven. 2016. What is the Computational Value of Finite Range Tunneling? *Physical Review X* 6 (2016), 031015. https://journals.aps.org/prx/abstract/10.1103/PhysRevX.6.031015

[23] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A quantum approximate optimization algorithm. *arXiv preprint 1411.4028* (2014).

[24] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. 2000. Quantum Computation by Adiabatic Evolution. *arXiv:quant-ph/0001106* (2000).

[25] AB Finnila, MA Gomez, C Sebenik, C Stenson, and JD Doll. 1994. Quantum annealing: a new method for minimizing multidimensional functions. *Chemical Physics Letters* 219, 5-6 (1994), 343–348.

[26] Behrouz A Forouzan. 2007. *Cryptography & network security.* McGraw-Hill, Inc.

[27] Leonidas Georgiadis, Michael J Neely, and Leandros Tassiulas. 2006. Resource allocation and cross-layer control in wireless networks. *Foundations and Trends® in Networking* 1, 1 (2006), 1–144.

[28] Stuart Hadfield, Zhihui Wang, Eleanor G. Rieffel, Bryan O'Gorman, Davide Venturelli, and Rupak Biswas. 2017. Quantum Approximate Optimization with Hard and Soft Constraints. In *Workshop on Post Moores Era Supercomputing (PMES).* https://doi.org/10.1145/3149526.3149530

[29] Ryan Hamerly, Takahiro Inagaki, Peter L McMahon, Davide Venturelli, Alireza Marandi, Tatsuhiro Onodera, Edwin Ng, Carsten Langrock, Kensuke Inaba, Toshimori Honjo, Koji Enbutsu, Takeshi Umeki, Ryoichi Kasahara, Shoko Utsunomiya, Satoshi Kako, Ken-ichi Kawarabayashi, Robert L. Byer, Martin M. Fejer, Hideo Mabuchi, Dirk Englund, Eleanor Rieffel, Hiroki Takesue, and Yoshihisa Yamamoto. 2018. Experimental investigation of performance differences between Coherent Ising Machines and a quantum annealer. *arXiv preprint arXiv:1805.05217* (2018).

[30] Babak Hassibi and Haris Vikalo. 2005. On the sphere-decoding algorithm I. Expected complexity. *IEEE transactions on signal processing* 53, 8 (2005), 2806–2818.

[31] Maritza Hernandez and Maliheh Aramon. 2017. Enhancing quantum annealing performance for the molecular similarity problem. *Quantum Information Processing* 16, 5 (2017), 133.

[32] Christopher Husmann, Georgios Georgis, Konstantinos Nikitopoulos, and Kyle Jamieson. 2017. FlexCore: Massively Parallel and Flexible Processing for Large MIMO Access Points. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17).* 197–211.

[33] Hiroshi Ishikawa. 2009. Higher-order clique reduction in binary graph cut. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).* 2993–3000.

[34] Joshua Job and Daniel Lidar. 2018. Test-driving 1000 qubits. *Quantum Science and Technology* 3, 3 (2018), 030501.

[35] Mark W Johnson, Mohammad HS Amin, Suzanne Gildert, Trevor Lanting, Firas Hamze, Neil Dickson, R Harris, Andrew J Berkley, Jan Johansson, Paul Bunyk, E M Chapple, C Enderud, Karimi Kamran Hilton, Jeremy P and, E Ladizinsky, Nicolas Ladizinsky, T Oh, I Perminov, C Rich, Murray C Thom, E Tolkacheva, Colin J S Truncik, S Uchaikin, J Wang, B Wilson, and Geordie Rose. 2011. Quantum annealing with manufactured spins. *Nature* 473, 7346 (2011), 194.

[36] Rinu Jose and Ameenudeen Pe. 2015. Analysis of hard decision and soft decision decoding algorithms of LDPC codes in AWGN. In *Advance Computing Conference (IACC), 2015 IEEE International.* IEEE, 430–435.

[37] Tadashi Kadowaki and Hidetoshi Nishimori. 1998. Quantum annealing in the transverse Ising model. *Phys. Rev.* E, 58 (1998), 5355–5363.

[38] James King, Sheir Yarkoni, Jack Raymond, Isil Ozfidan, Andrew D King, Mayssam Mohammadi Nevisi, Jeremy P Hilton, and Catherine C McGeoch. 2019. Quantum annealing amid local ruggedness and global frustration. *Journal of the Physical Society of Japan* 88, 6 (2019), 061007.

[39] Christine Klymko, Blair D Sullivan, and Travis S Humble. 2014. Adiabatic quantum programming: minor embedding with hard faults. *Quantum information processing* 13, 3 (2014), 709–729.

[40] Mark Lewis and Fred Glover. 2017. Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis. *Networks* 70, 2 (2017), 79–97.

[41] Xiaojun Liu, Edwin K. P. Chong, and Ness B. Shroff. 2001. Opportunistic transmission scheduling with resource-sharing constraints in wireless networks. *IEEE Journal on Selected Areas in Communications* 19, 10 (2001), 2053–2064.

[42] Andrew Lucas. 2014. Ising formulations of many NP problems. *Frontiers in Physics* 2 (2014), 5. https://doi.org/10.3389/fphy.2014.00005

[43] Jeffrey Marshall, Davide Venturelli, Itay Hen, and Eleanor Rieffel. 2018. The power of pausing: advancing understanding of thermalization in experimental quantum annealers. *arXiv:1810.05881* (2018).

[44] Mahmood Mazrouei-Sebdani and Witold A Krzymien. 2012. Vector perturbation precoding for network MIMO: Sum rate, fair user scheduling, and impact of backhaul delay. *IEEE Transactions on Vehicular Technology* 61, 9 (2012), 3946–3957.

[45] Catherine C McGeoch. 2014. Adiabatic quantum computation and quantum annealing: Theory and practice. *Synthesis Lectures on Quantum Computing* 5, 2 (2014), 1–93.

[46] Catherine C McGeoch and Cong Wang. 2013. Experimental evaluation of an adiabatic quantum system for combinatorial optimization. In *Proceedings of the ACM International Conference on Computing Frontiers.* ACM, 23.

[47] Alex Mott, Joshua Job, Jean-Roch Vlimant, Daniel Lidar, and Maria Spiropulu. 2017. Solving a Higgs optimization problem with quantum annealing for machine learning. *Nature* 550, 7676 (2017), 375.

[48] Shinobu Namba, Takayuki Warabino, and Shoji Kaneko. 2012. BBU-RRH switching schemes for centralized RAN. In *7th International Conference on Communications and Networking in China.* IEEE, 762–766.

[49] Richard van Nee and Ramjee Prasad. 2000. *OFDM for wireless multimedia communications.* Artech House, Inc.

[50] Konstantinos Nikitopoulos, Juan Zhou, Ben Congdon, and Kyle Jamieson. 2014. Geosphere: Consistently turning MIMO capacity into throughput. In *Proc. of the ACM SIGCOMM Conf.* 631–642.

[51] Sergey Novikov, Robert Hinkey, Steven Disseler, James I Basham, Tameem Albash, Andrew Risinger, David Ferguson, Daniel A Lidar, and Kenneth M Zick.

2018. Exploring More-Coherent Quantum Annealing. *arXiv preprint arXiv:1809.04485* (2018).

[52] Bryan O'Gorman, Eleanor G Rieffel, Minh Do, Davide Venturelli, and Jeremy Frank. 2015. Compiling planning into quantum optimization problems: a comparative study. *Constraint Satisfaction Techniques for Planning and Scheduling Problems (COPLAS-15)* (2015), 11.

[53] Yosuke Otsubo, Junichi Inoue, Kenji Nagata, and Masato Okada. 2014. Code-division multiple-access multiuser demodulator by using quantum fluctuations. *Physical Review E* 90, 1 (2014), 012126.

[54] Alejandro Perdomo-Ortiz, Marcello Benedetti, John Realpe-Gómez, and Rupak Biswas. 2017. Opportunities and challenges for quantum-assisted machine learning in near-term quantum computers. *arXiv preprint arXiv:1708.09757* (2017).

[55] Alejandro Perdomo-Ortiz, Joseph Fluegemann, Sriram Narasimhan, Rupak Biswas, and Vadim N Smelyanskiy. 2015. A quantum annealing approach for fault detection and diagnosis of graph-based systems. *The European Physical Journal Special Topics* 224, 1 (2015), 131–148.

[56] John Preskill. 2018. Quantum Computing in the NISQ era and beyond. *arXiv preprint arXiv:1801.00862* (2018).

[57] Eleanor G. Rieffel, Davide Venturelli, Bryan O'Gorman, Minh B. Do, Elicia M. Prystay, and Vadim N. Smelyanskiy. 2015. A case study in programming a quantum annealer for hard operational planning problems. *Quantum Information Processing* 14, 1 (01 Jan 2015), 1–36. https://doi.org/10.1007/s11128-014-0892-x

[58] Troels F Rønnow, Zhihui Wang, Joshua Job, Sergio Boixo, Sergei V Isakov, David Wecker, John M Martinis, Daniel A Lidar, and Matthias Troyer. 2014. Defining and detecting quantum speedup. *Science* 345, 6195 (2014), 420–424.

[59] Gili Rosenberg, Poya Haghnegahdar, Phil Goddard, Peter Carr, Kesheng Wu, and Marcos López de Prado. 2015. Solving the Optimal Trading Trajectory Problem Using a Quantum Annealer. In *ACM Workshop on High Performance Computational Finance (WHPCF)*. https://doi.org/10.1145/2830556.2830563

[60] Peter Rost and Athul Prasad. 2014. Opportunistic hybrid ARQ-Enabler of centralized-RAN over nonideal backhaul. *IEEE Wireless Communications Letters* 3, 5 (2014), 481–484.

[61] Clayton Shepard, Jian Ding, Ryan E Guerra, and Lin Zhong. 2016. Understanding real many-antenna MU-MIMO channels. In *Signals, Systems and Computers, 2016 50th Asilomar Conference on*. IEEE, 461–467.

[62] Clayton Shepard, Hang Yu, Narendra Anand, Erran Li, Thomas Marzetta, Richard Yang, and Lin Zhong. 2012. Argos: Practical many-antenna base stations. In *Proceedings of the 18th annual international conference on Mobile computing and networking*. ACM, 53–64.

[63] Karthikeyan Sundaresan, Mustafa Y. Arslan, Shailendra Singh, Sampath Rangarajan, and Srikanth V. Krishnamurthy. 2016. FluidNet: A Flexible Cloud-based Radio Access Network for Small Cells. *IEEE/ACM Trans. on Networking* 24, 2 (April 2016), 915–928. https://doi.org/10.1109/TNET.2015.2419979

[64] Kun Tan, He Liu, Ji Fang, Wei Wang, Jiansong Zhang, Mi Chen, and Geoffrey M. Voelker. 2009. SAM: Enabling Practical Spatial Multiple Access in Wireless LAN. In *Proc. of the ACM MobiCom Conf.*

[65] Emre Telatar. 1999. Capacity of multi-antenna Gaussian channels. *European transactions on telecommunications* 10, 6 (1999), 585–595.

[66] Immanuel Trummer and Christoph Koch. 2016. Multiple Query Optimization on the D-Wave 2X Adiabatic Quantum Computer. *Proc. VLDB Endow.* 9, 9 (May 2016), 648–659. https://doi.org/10.14778/2947618.2947621

[67] David Tse and Pramod Viswanath. 2005. *Fundamentals of wireless communication*. Cambridge university press.

[68] Davide Venturelli and Alexei Kondratyev. 2018. Reverse Quantum Annealing Approach to Portfolio Optimization Problems. *arXiv:1810.08584* (2018).

[69] Davide Venturelli, Salvatore Mandrà, Sergey Knysh, Bryan O'Gorman, Rupak Biswas, and Vadim Smelyanskiy. 2015. Quantum Optimization of Fully Connected Spin Glasses. *Phys. Rev. X* 5 (Sep 2015), 031040. Issue 3. https://doi.org/10.1103/PhysRevX.5.031040

[70] Davide Venturelli, Dominic JJ Marchand, and Galo Rojo. 2015. Quantum annealing implementation of job-shop scheduling. *arXiv preprint arXiv:1506.08479* (2015).

[71] Emanuele Viterbo and Joseph Boutros. 1999. A universal lattice code decoder for fading channels. *IEEE Trans. Inf. Theory* 45, 5 (1999), 1639–1642.

[72] Di Wang and Robert Kleinberg. 2009. Analyzing quadratic unconstrained binary optimization problems via multicommodity flows. *Discrete Applied Mathematics* 157, 18 (2009), 3746–3753.

[73] Markus Wenk, Lukas Bruderer, Andreas Burg, and Christoph Studer. 2010. Area-and throughput-optimized VLSI architecture of sphere decoding. In *IEEE/IFIP 18th VLSI System on Chip Conference (VLSI-SoC)*. 189–194.

[74] Markus Winter, Steffen Kunze, Esther Perez Adeva, Björn Mennenga, Emil Matûs, Gerhard Fettweis, Holger Eisenreich, Georg Ellguth, Sebastian Höppner, Stefan Scholze, René Schüffny, and Tomoyoshi Kobiry. 2012. A 335Mb/s 3.9 mm 2 65nm CMOS flexible MIMO detection-decoding engine achieving 4G wireless data rates. In *2012 IEEE International Solid-State Circuits Conference*. IEEE, 216–218.

[75] WEI Wu, Sangjin Hong, and Do-Sik Yoo. 2003. Block length of LDPC codes in fading channels. In *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, Vol. 3. IEEE, 1876–1880.

[76] Qing Yang, Xiaoxiao Li, Hongyi Yao, Ji Fang, Kun Tan, Wenjun Hu, Jiansong Zhang, and Yongguang Zhang. 2013. BigStation: Enabling scalable real-time signal processing in large MU-MIMO systems. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 399–410.

[77] Zhi-Cheng Yang, Armin Rahmani, Alireza Shabani, Hartmut Neven, and Claudio Chamon. 2017. Optimizing Variational Quantum Algorithms Using Pontryagin's Minimum Principle. *Physical Review X* 7, 2 (2017), 021027.

[78] Zheng Zhu, Andrew J. Ochoa, Stefan Schnabel, Firas Hamze, and Helmut G. Katzgraber. 2016. Best-case performance of quantum annealers on native spin-glass benchmarks: How chaos can affect success probabilities. *Phys. Rev. A* 93 (Jan 2016), 8. Issue 1. https://doi.org/10.1103/PhysRevA.93.012317

Appendices are supporting material that has not been peer reviewed.

A QUBO FORMS

We demonstrate how to transform 2×2 BPSK MIMO Maximum Likelihood (ML) detection into the QUBO form. ML detection solves Eq. 1, where

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} = \begin{bmatrix} h_{I,11} & h_{I,12} \\ h_{I,21} & h_{I,22} \end{bmatrix} + j \begin{bmatrix} h_{Q,11} & h_{Q,12} \\ h_{Q,21} & h_{Q,22} \end{bmatrix},$$

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} y_{I,1} \\ y_{I,2} \end{bmatrix} + j \begin{bmatrix} y_{Q,1} \\ y_{Q,2} \end{bmatrix} \text{ and } \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}.$$

The norm expansion in Eq. 1 can be expressed as

$$\|\mathbf{y} - \mathbf{H}\mathbf{v}\|^2 = \left\| \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} - \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \right\|^2 = \left\| \begin{matrix} y_1 - h_{11}v_1 - h_{12}v_2 \\ y_2 - h_{21}v_1 - h_{22}v_2 \end{matrix} \right\|^2$$
$$= \left\| \begin{matrix} (y_{I,1} - h_{I,11}v_1 - h_{I,12}v_2) + j(y_{Q,1} - h_{Q,11}v_1 - h_{Q,12}v_2) \\ (y_{I,2} - h_{I,21}v_1 - h_{I,22}v_2) + j(y_{Q,2} - h_{Q,21}v_1 - h_{Q,22}v_2) \end{matrix} \right\|^2$$
$$= \{(y_{I,1} - h_{I,11}v_1 - h_{I,12}v_2)\}^2 + \{(y_{Q,1} - h_{Q,11}v_1 - h_{Q,12}v_2)\}^2$$
$$+ \{(y_{I,2} - h_{I,21}v_1 - h_{I,22}v_2)\}^2 + \{(y_{Q,2} - h_{Q,21}v_1 - h_{Q,22}v_2)\}^2.$$

In the case of BPSK, symbol $v_i \in \{-1, 1\}$ is represented by a QUBO variable q_i. One possible transform is $2q_i - 1$ where $q_i = 0$ corresponds to $v_i = -1$ and $q_i = 1$ to $v_i = 1$. This leads to $\mathbf{v} = [v_1, v_2]^\intercal = [\mathbf{T(q_1)}, \mathbf{T(q_2)}]^\intercal$, where $\mathbf{T(q_1)} = 2q_1 - 1$ and $\mathbf{T(q_2)} = 2q_2 - 1$. Using these relationships, we can express the above norm as

$$\|\mathbf{y} - \mathbf{H}\mathbf{v}\|^2 = \left\| \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} - \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} \mathbf{T(q_1)} \\ \mathbf{T(q_2)} \end{bmatrix} \right\|^2$$
$$= \{(y_{I,1} - h_{I,11}(2q_1 - 1) - h_{I,12}(2q_2 - 1))\}^2$$
$$+ \{(y_{Q,1} - h_{Q,11}(2q_1 - 1) - h_{Q,12}(2q_2 - 1))\}^2$$
$$+ \{(y_{I,2} - h_{I,21}(2q_1 - 1) - h_{I,22}(2q_2 - 1))\}^2$$
$$+ \{(y_{Q,2} - h_{Q,21}(2q_1 - 1) - h_{Q,22}(2q_2 - 1))\}^2.$$

Then we obtain the objective function of ML problem with QUBO variables. Using $q_i^2 = q_i$, minimization of this objective function becomes the QUBO form (Eq. 3):

$$\hat{q}_1, \hat{q}_2 = \arg\min_{q_1, q_2} Q_{11}q_1 + Q_{22}q_2 + Q_{12}q_1q_2, \text{ where}$$
$$Q_{11} = -4h_{I,11}y_{I,1} - 4h_{I,21}y_{I,2} - 4h_{Q,11}y_{Q,1} - 4h_{Q,21}y_{Q,2}$$
$$-4h_{I,11}h_{I,12} - 4h_{I,21}h_{I,22} - 4h_{Q,11}h_{Q,12} - 4h_{Q,21}h_{Q,22},$$
$$Q_{22} = -4h_{I,12}y_{I,1} - 4h_{I,22}y_{I,2} - 4h_{Q,12}y_{Q,1} - 4h_{Q,22}y_{Q,2}$$
$$-4h_{I,12}h_{I,12} - 4h_{I,22}h_{I,22} - 4h_{Q,12}h_{Q,12} - 4h_{Q,22}h_{Q,22},$$
$$Q_{12} = 8h_{I,11}h_{I,12} + 8h_{I,21}h_{I,22} + 8h_{Q,11}h_{Q,12} + 8h_{Q,21}h_{Q,22}.$$

B EMBEDDED ISING

Embedding maps the Ising problem to an equivalent one that has the same ground state, but also satisfies Chimera graph constraints. The QuAMax compiled objective function is:

$$-\sum_{i=1}^{N}\left[\sum_{c=1}^{\lceil N/4\rceil} s_{ic}s_{i(c+1)}\right] \tag{10}$$

$$+\sum_{i=1}^{N}\left(\frac{\mathbf{f_i}}{|J_F|\left(\lceil\frac{N}{4}\rceil+1\right)}\right)\left[\sum_{c=1}^{\lceil N/4\rceil+1} s_{ic}\right] \tag{11}$$

$$+\sum_{i,j=1}^{N}\frac{\mathbf{g_{ij}}}{|J_F|}\sum_{(c_i,c_j)\in\delta_{ij}} s_{ic_i}s_{jc_j} \tag{12}$$

where the original logical variables s_i are now associated to a chain i of $c = 1\ldots(\lceil N/4\rceil+1)$ qubits, indexed with new spins s_{ic}. $|J_F|$ penalizes the condition that $s_{ic} \neq s_{ic'}$, *i.e.*, enforces that all qubits in the chain assume the same value (±1). This enforcement is more likely to happen for large values of $|J_F|$, however the maximum negative energy value is set to -1 by hardware design. In (11) and (12), $|J_F|$ effectively renormalizes all terms in the objective function by the factor $|J_F|^{-1}$. The linear term value $\mathbf{f_i}$ is additionally divided by the number of qubits in a chain ($\lceil N/4\rceil+1$). The term in (12) shows that the duplication of variables ensures the existence of a pair of qubits in the chains such that a physical coupler in the Chimera graph exists (δ_{ij} is the set of pairs of qubits that are connected by a physical bond once the chains i and j are specified).

C 16-QAM ISING MODEL PARAMETERS

Following are the Ising parameters f_i for 16-QAM:

$$f_i(\mathbf{H},\mathbf{y}) = \begin{cases} \text{case } i = 4n-3: \\ -4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{y}^I\right) - 4\left(\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\cdot\mathbf{y}^Q\right), \\ \text{case } i = 4n-2: \\ -2\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{y}^I\right) - 2\left(\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\cdot\mathbf{y}^Q\right), \\ \text{case } i = 4n-1: \\ -4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{y}^Q\right) + 4\left(\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\cdot\mathbf{y}^I\right), \\ \text{case } i = 4n: \\ -2\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{y}^Q\right) + 2\left(\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\cdot\mathbf{y}^I\right). \end{cases} \tag{13}$$

Since real and imaginary terms of each symbol are independent, the coupler strength between s_{4n-3}, s_{4n-2} and s_{4n-1}, s_{4n} is 0. For other s_i and s_j, the Ising coupler strength g_{ij} for 16-QAM is:

$$g_{ij}(\mathbf{H}) = \begin{cases} \text{case } i = 4n-3: \\ \begin{cases} \text{case } j = 4n'-3: \\ 8\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^I_{(:,\lceil j/4\rceil)}\right) + 8\left(\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n'-2: \\ 4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^I_{(:,\lceil j/4\rceil)}\right) + 4\left(\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n'-1: \\ -8\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\right) + 8\left(\mathbf{H}^I_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n': \\ -4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\right) + 4\left(\mathbf{H}^I_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \end{cases} \\ \text{case } i = 4n-2: \\ \begin{cases} \text{case } j = 4n'-3: \\ 4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^I_{(:,\lceil j/4\rceil)}\right) + 4\left(\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n'-2: \\ 2\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^I_{(:,\lceil j/4\rceil)}\right) + 2\left(\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n'-1: \\ -4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\right) + 4\left(\mathbf{H}^I_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n': \\ -2\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\right) + 2\left(\mathbf{H}^I_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \end{cases} \\ \text{case } i = 4n-1: \\ \begin{cases} \text{case } j = 4n'-3: \\ 8\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\right) - 8\left(\mathbf{H}^I_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n'-2: \\ 4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\right) - 4\left(\mathbf{H}^I_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n'-1: \\ 8\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^I_{(:,\lceil j/4\rceil)}\right) + 8\left(\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n': \\ 4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^I_{(:,\lceil j/4\rceil)}\right) + 4\left(\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \end{cases} \\ \text{case } i = 4n: \\ \begin{cases} \text{case } j = 4n'-3: \\ 4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\right) - 4\left(\mathbf{H}^I_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n'-2: \\ 2\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\right) - 4\left(\mathbf{H}^I_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n'-1: \\ 4\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^I_{(:,\lceil j/4\rceil)}\right) + 4\left(\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right), \\ \text{case } j = 4n': \\ 2\left(\mathbf{H}^I_{(:,\lceil i/4\rceil)}\cdot\mathbf{H}^I_{(:,\lceil j/4\rceil)}\right) + 2\left(\mathbf{H}^Q_{(:,\lceil j/4\rceil)}\cdot\mathbf{H}^Q_{(:,\lceil i/4\rceil)}\right). \end{cases} \end{cases} \tag{14}$$