The Quantum Computing Company™

Optimization Performance of QA and QAOA

WHITEPAPER

Summary

We survey the research literature on quantum optimization solvers, comparing the QA approach implemented by D-Wave[1] to the QAOA approach, which runs on gate-model quantum platforms built by other companies.

Taken together, these published works show that QA significantly outperforms QAOA on current-generation quantum platforms, and that QA will continue to dominate the quantum optimization space on future (including error corrected) quantum platforms.

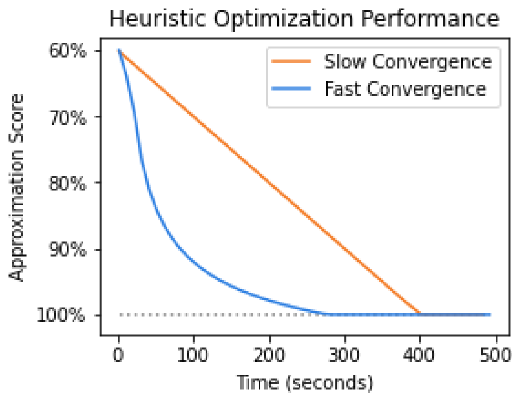

Figure 1: The blue solver outperforms the orange solver because it converges more rapidly to optimality, corresponding to a score of 100%.

Two quantum approaches for tackling hard optimization problems are available today: the quantum annealing (QA) method implemented on D-Wave quantum processing units (QPUs), and the quantum approximate optimization algorithm (QAOA), which runs on NISQ-era gate-model (GM) QPUs built by other companies.[2] This white paper surveys recent published research describing their performance today and on future quantum platforms.

Briefly: empirical tests using today's QPUs show unequivocally that QA significantly outperforms QAOA. Furthermore, recent theoretical results identify some obstacles to viability of QAOA, suggesting that this approach will never be competitive even if run on future (error-corrected) GM QPUs. These obstacles do not apply to the QA approach, which continues to show progress at both scaling up and demonstrating improved performance with every new generation.

Similarities and differences

Both QA and QAOA are designed to heuristically solve hard problems in discrete optimization. Every optimization problem is defined in terms of an objective function that provides a quality score $s = f(x)$ for every solution x; an *optimal* solution has the best score. Heuristics are informal algorithms that do not provide

[1]D-Wave™ and Advantage™ are trademarks of D-Wave Systems Inc. in the United States and other countries. IONQ™ and IONQ ARIA™ are trademarks of IonQ, Inc. in the United States and other countries. IBM™ is a trademark of International Business Machines Corp. registered in many jurisdictions worldwide.

[2]NISQ stands for Noisy Intermediate Scale Quantum computing, which refers to quantum hardware having high error rates but not enough qubits to implement quantum error correction. It is generally believed that GM QPUs will not be commercially viable until some future post-NISQ era when qubit counts will be high enough to support error correction.

guaranteed optimal solutions, but can typically find very good solutions very quickly. Many heuristics have an "effort" parameter allowing the user to specify a computation time limit. As shown in Figure 1, heuristic performance is evaluated according to the tradeoff between solution quality and computation time: here, the blue solver outperforms the orange solver.

The main differences between QA and QAOA lie in the quantum architectures on which they are designed to run. QA is implemented directly in purpose-built quantum hardware. Given a problem with objective function $f(x)$, the control system drives all qubits simultaneously, through a smooth transition (called an anneal) from an initial random state to one that matches the objective, so that the qubits finish in states x corresponding to optimal and near-optimal solutions to $f(x)$.

In contrast, QAOA is implemented in software and runs like any other program on general-purpose GM hardware.[3] QAOA was originally designed to mimic QA [2], except the smooth anneal transition is replaced with a step function having p steps: the user specifies p together with two additional parameters (β, γ) for each step. Increasing the number of steps $p \to \infty$ improves solution quality and allows QAOA to better-approximate the smooth quantum anneal.

Empirical comparisons

We highlight three research papers describing empirical performance tests of QA and QAOA.

A paper published by researchers at the Quantum Economic Development Consortium (QED-C) compared a D-Wave Advantage QPU (5000+ qubits) with QAOA ($p = 2$ steps) running on two IonQ QPUs (10 qubits) and one IBM QPU (16 qubits) [1]. The table in Figure 2 summarizes some data from the paper.

The blue rows highlight the QPU with best performance in each size category. In terms of both Elapsed and Quantum computation times, the Advantage system was 200 to 9,000 times faster than the GM systems at these problem sizes. In terms of solution quality, the Advantage QPU was able to find optimal solutions on smaller inputs and to get within 97% of optimal on larger inputs, while solution quality for the GM QPUs ranged from 50% (indistinguishable from random outputs in these tests) to about 80% of optimal.

[3]Computability theory tells us that the two quantum approaches are equivalent in computational power. However, as deployed, QA platforms target optimization performance, while GM platforms aim to execute general programs.

Other studies confirm the observation that QA returns better solutions:

- Researchers at Los Alamos National Laboratory (LANL) [3] compared two Advantage QPUs to QAOA ($p = 1, 2$) on an IBM (127 qubit) QPU, using a problem selected to favor QAOA. QA found better solutions in every test. The authors do not directly compare runtimes, but remark that QAOA times were "very high."
- Another LANL paper [4] compares QA and QAOA ($p = 1$) on twelve different QPUs manufactured by D-Wave (3), IBM (7), IonQ (1), and Rigetti (1). D-Wave QPUs nearly always returned optimal solutions, while approximation ratios for QAOA solvers ranged between 50% and 98% of optimal. (Runtimes are not reported.)

In all three papers, the *best* solutions returned by QAOA could not match the *worst* solutions returned by QA. We are not aware of any published research that contradicts these findings (see also [5, 6]).

The next section surveys papers that partially explain this performance gap; some go further and predict that QAOA will never be competitive, even if run on error-corrected GM platforms of the future.

Challenges to QAOA viability

The recent theoretical literature identifies several issues suggesting that QAOA will never be competitive as an optimization heuristic (finding good solutions fast):

- The problem of finding optimal parameters (β, γ) is NP-Hard [7]. This means that in order to find a good-quality solution to $f(x)$, it is first necessary to find $2p$ solutions to a different and provably-hard optimization problem with objective $f_p(\beta, \gamma)$.
- The problem $f_p(\beta, \gamma)$ has barren plateaus [8]. This and other obstacles (see [9]) suggest that classical heuristics (which can work well on some NP-hard problems) are unlikely to be effective. A recent empirical study of five heuristics for this parameter-

QPU	Size (qubits)	Elapsed Time (s)	Quantum Time (s)	Approx. Ratio
IONQ `221006` $p = 2$	10	5000	550	50%
IONQ ARIA $p = 2$	10	na(1)	2000	80%
D-Wave Advantage 4.1	12(2)	.55	.26	100%
IBM `guadalupe` $p = 2$	16	155	120	50%
D-Wave Advantage 4.1	16	.75	.24	100%
D-Wave Advantage 4.1	320	4	.32	97%

Figure 2: (Summarizing data from Figures $7 - 19$ of ref. [1].) Elapsed time (seconds) includes classical setup time and other system overheads; quantum time measures only computations on the QPU. The Approximation Ratio column shows the mean ratio $f(x)/f(Optimal)$ for solutions from each QPU. A score of 50% indicates the solutions were indistinguishable from random outputs, and 100% corresponds to an optimal solution. **Notes:** (1) runtimes were not reported for this test; (2) results are shown for the size nearest to 10 that was tested.

finding task showed that none was fast and accurate enough for QAOA to outperform QA [10].

- Increasing p can improve QAOA solution quality, but it also increases circuit depth. On today's NISQ machines, solution quality deteriorates rapidly with circuit depth, up to a point where outputs look like random bit strings and are therefore unusable. Although cases have been found where QAOA is effective with small fixed p, in general p must grow with problem size [11, 12].

- Error-corrected QPUs of the future might be expected to remove this last problem with deteriorating quality as depth increases. However, it has been shown that, under reasonable assumptions, the overhead times needed to implement error correction on GM platforms will likely negate any possible speedup that QAOA might offer over classical optimization methods [13]. Furthermore, error correction does not solve the first two problems, which can lead to enormous runtimes no matter what hardware it runs on.

Conclusions

Results from empirical comparisons are unequivocal: today's QA processors built by D-Wave far outperform GM processors at tasks related to combinatorial optimization. As well, theoretical results have been published suggesting that QAOA will never be able to overcome several obstacles to viability, on near-term and future (error-corrected) GM QPUs.

Moreover, the list of obstacles for QAOA does not apply to QA processors implemented by D-Wave, for several reasons. First, implementation on purpose-built hardware means that there is no p-step function but instead a smoothly-evolving anneal that behaves like $p = \infty$. Second, there is nothing analogous to the task of optimizing parameters (β, γ) in the QA workflow. More precisely, although several QA parameters are available, and can be tuned for better performance, most users find that default parameter settings work fine in everyday use.

Third, while all quantum computations suffer from errors due to noise, the types of errors that arise have different effects on solution quality on QA and GM architectures. Simply put, when qubits decohere in a QAOA computation, they decohere toward their *individual* ground states: in a worst-case scenario, outputs look like random bit strings and the computation is useless. The QA strategy is to align the *collective* qubit ground state with the objective function to be optimized, so that even in noisy conditions, outputs are reliably optimal or near-optimal. These design differences, together with the relative maturity of D-Wave technologies, explain the performance gaps observed in papers surveyed here.

Indeed, recent years have seen a number of papers demonstrating that D-Wave QPUs can compete with and sometimes outperform classical approaches at tasks related to quantum materials simulation, combinatorial optimization and sampling. This trend of dominant performance on ever-broader varieties of problems is expected to continue.

References

1 T. Lubinski et al., "Optimization applications as quantum performance benchmarks," (manuscript in submission) arXiv:2302.02278, The data in this white paper is estimated from several graphical figures. (2023).

2 E. Farhi et al., "A quantum approximate optimization algorithm," arXiv:1411.4028 (2014).

3 E. Pelofske et al., "Quantum annealing vs QAOA: 127 qubit higher-order Ising problems on NISQ computers," Int. Conf. on High Performance Computing (2023).

4 E. Pelofske et al., "Sampling on NISQ devices: Who's the fairest one of all?" Int. Conf. Quantum Computing and Engineering (2021).

5 "A head-to-head comparison of D-Wave and Rigetti QPUs," D-Wave White Paper 14-1025A-D (2018).

6 M. Willsch et al., "Benchmarking the quantum approximate optimization algorithm," Quant. Inf. Proc **19** (2020).

7 L. Bittel and M. Kleisch, "Training variational quantum algorithms is NP-Hard," Phys. Rev. Lett. **127** (2021).

8 S. Wang et al., "Noise induced barren plateaus in variational quantum algorithms," Nature Communications **12** (2020).

9 V. Akshay et al., "Reachability deficits in quantum approximate optimization," Phys. Rev. Lett. **124** (2020).

10 A. R. Mazumder et al., "Benchmarking metaheuristic-integrated QAOA against quantum annealing," arXiv: 2309.16296 (2023).

11 E. Farhi et al., "The quantum approximate optimization algorithm needs to see the whole graph: worst case examples," (2020).

12 J. Wurtz and P. Love, "MaxCut quantum approximate optimization algorithm performance guarantees for $p > 1$," Phys. Rev. A **103** (2023).

13 Y. R Sanders et al., "Compilation of fault-tolerant quantum heuristics for quantum optimization," PRX Quantum **1** (2020).