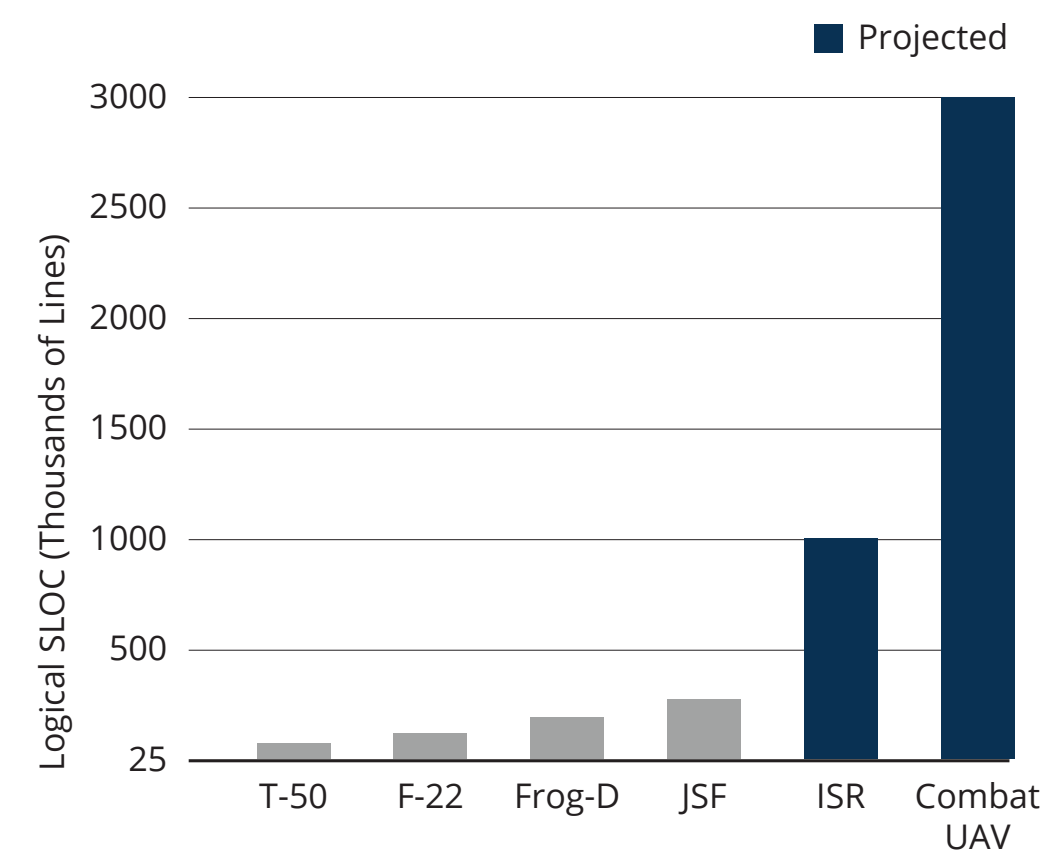


# Projecting Quantum Computational Advantage Versus Classical State of the Art

## Introduction

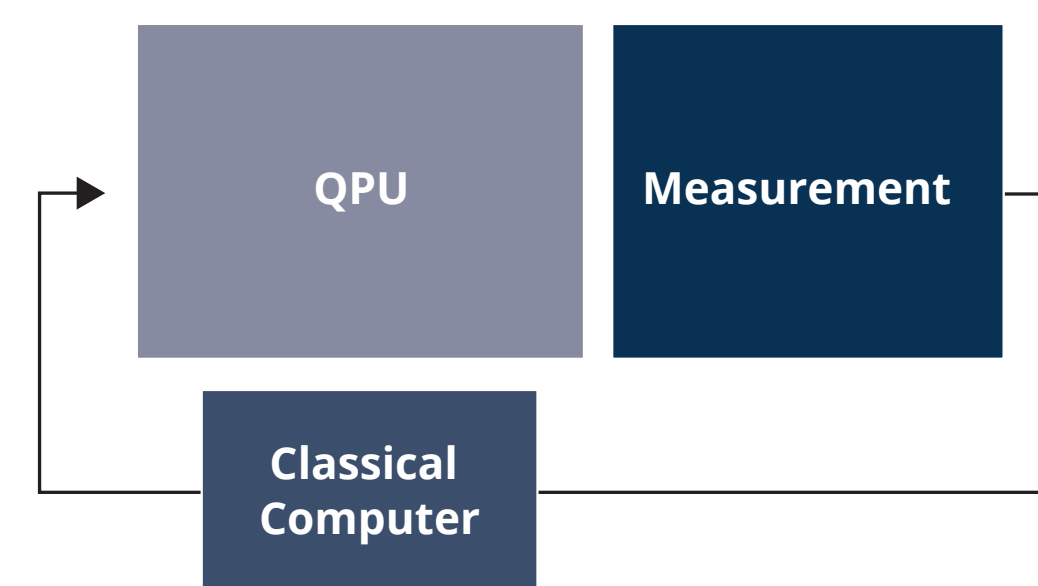
A major milestone in quantum computing research is to demonstrate quantum supremacy, which refers to a quantum computer performing a calculation that is unfeasible for a classical computer [1]. While quantum supremacy may be demonstrable in the near-term noisy intermediate scale quantum computing (NISQ) era [2]-[6], such a demonstration of supremacy does not afford an advantage for practical applications. A common practical problem used in benchmarking high performance classical and quantum computing is Maxcut, with applications in domains such as machine scheduling [7], image recognition [8], electronic circuit layout [9], and software verification and validation [10], [11].



T. Belote, C. Elliott, Safe & Secure Systems & Software Symposium 2014.

# Achieving quantum advantage requires classical state of the art.

Classical High-Level Lang	Quantam High-Level Lang
Compiler/OS	IBM Qiskit, Rigetti Grove, Google Cirq, ...
Architecture	
VLSI	
Emulator	Emulator
IC (CPU)	IC (QPU)



## Infrastructure

For practical applications, quantum advantage has to be measured against the best performance that classical computing offers. This work uses the hybrid quantum-classical Quantum Approximate Optimization Algorithm (QAOA) [12].

## QAOA

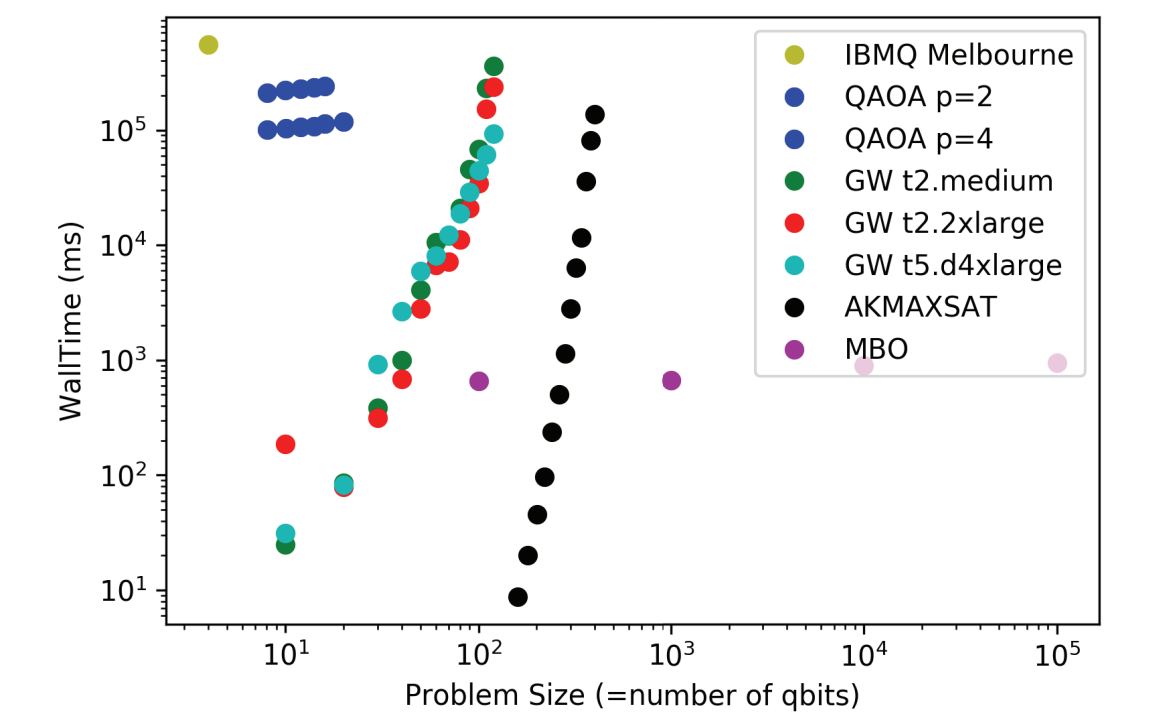
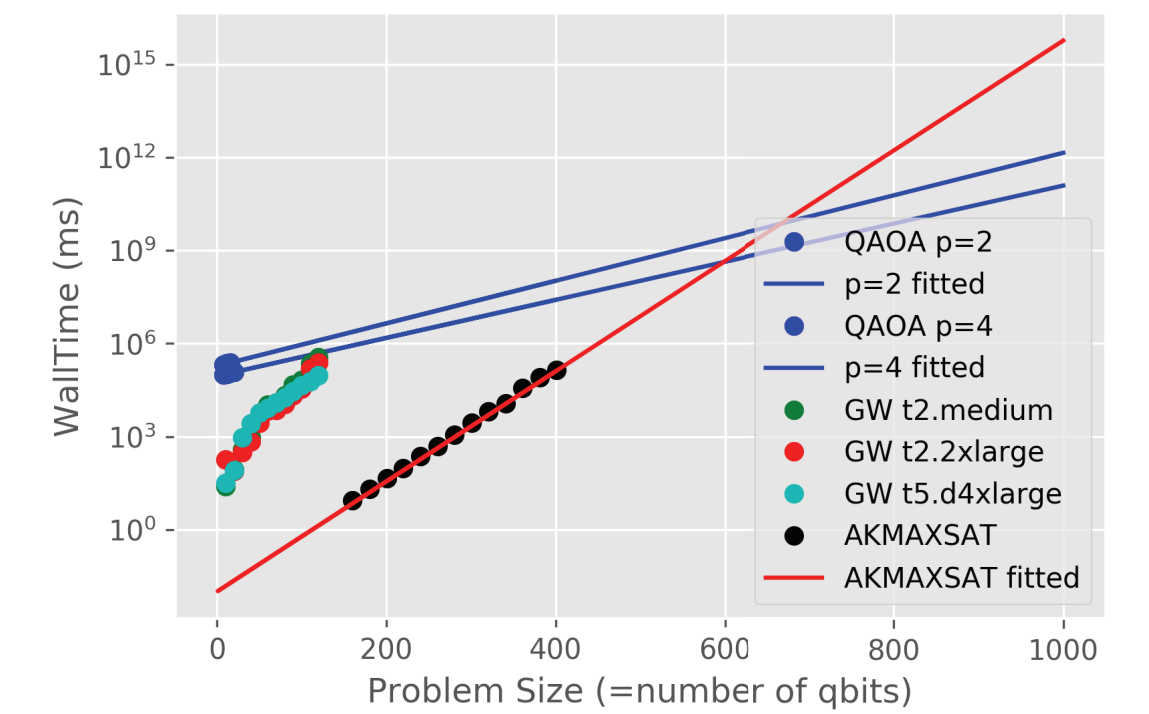
- GW (SDP) [13]
- AKAXSAT (BnB) [14]
- MBO (Laplacian) [15]
- UFO [16]

## QAOA

- AWS: C5 Intel Xeon
- PSC: Bridges (12 TB)
- IBM Melbourne 16

## Performance Results

The quality of solutions (cut value, bounds) are all comparable among algorithms (classical and QAOA).



## Future Work

- Improve QAOA parameterization.
- Increase number of hardware configs. What is needed from hardware to achieve advantage?
- Understand MBO and UFO performance. Is there a quantum version?

[1] John Preskill. "Quantum Computing in the NISQ Era and Beyond."  
 [2] Sergio Boixo, et al. "Characterizing Quantum Supremacy in Near-Term Devices." 2018.  
 [3] C. Neill. "A Blueprint for Demonstrating Quantum Supremacy with Superconducting Qubits." 2018.  
 [4] H. Wang. "Toward Scalable Boson Sampling with Photon Loss." 2018.  
 [5] Edward Farhi, et al. "Quantum Supremacy through the Quantum Approximate Optimization Algorithm." 2016.  
 [6] Michael J. Bremner. "Achieving Quantum Supremacy with Sparse and Noisy Commuting Quantum Computations." 2017.  
 [7] Bahram Alidaee, et al. "0-1 Quadratic Programming Approach for Optimum Solutions of Two Scheduling Problems." 1994.  
 [8] Hartmut Neven, et al. "Image recognition with an Adiabatic." 2007.  
 [9] Michel Deza and Monique Laurent. "Applications of Cut Polyhedra." 1994.  
 [10] Manu Jose, et al. "Cause Clue Clauses: Error Localization Using Maximum Satisfiability." 2011.

[11] L. Guo, et al. "A Complexity Metric for Concurrent Finite State Machine Based Embedded Software." 2013.  
 [12] Edward Farhi, et al. "A Quantum Approximate Optimization Algorithm." 2014  
 [13] Michel X. Goemans, et al. "Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming." 1995.  
 [14] Adrian Kügel. "Improved Exact Solver for the Weighted Max-SAT Problem." 2011  
 [15] Blaine Keetch, et al. "A Max-Cut Approximation Using a Graph Based MBO Scheme." 2017.  
 [16] V. Ashcay, et al. "Reachability Deficits in Quantum Approximate Optimization." 2019.  
 [17] M.B. Hastings. "Classical and Quantum Bounded Depth Approximation Algorithms." 2019.  
 [18] S.S. Tannu, et al. "Mitigating Measurement Errors in Quantum Computers by Exploiting State-Dependent Bias." 2019  
 [19] S.S. Tannu, et al. "Ensemble of Diverse Mappings: Improving Reliability of Quantum Computers by Orchestrating Dissimilar Mistakes." 2019.

Copyright 2019 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

Internal use:\* Permission to reproduce this material and to prepare derivative works from this material for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use:\* This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

\* These restrictions do not apply to U.S. government entities.

Carnegie Mellon® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM19-1037