Quantum computing at scale

Yuri Alexeev

Computational Science Division and

Argonne Leadership Computing Facility

QIS Projects in CELS (ALCF, BIO, CPS, DSL, ES, MCS)

Project description	Collaborators	Funding Agency
Advancing Integrated Development Environments for Quantum Computing through Fundamental Research	**LBNL,** ANL, SNL, LANL, ORNL, UChicago	ASCR ARQC
Fundamental Algorithmic Research for Quantum Computing	**SNL,** ANL, LANL, LBNL, ORNL, University of Maryland, Caltech, Dartmouth	ASCR ARQC
Quantum Algorithms, Mathematics and Compilation Tools for Chemical Sciences	**LBNL**, ANL, University of Toronto, University of California Berkeley	ASCR QAT
Illinois-Express Quantum Network	**Fermilab**, ANL, Caltech, Harvard, Northwestern	ASCR TOQNDS
Parameter sweep for SRF cavities using simulators and HPC	**Fermilab**, ANL	HEP QuantiSED
Discovering new microscopic descriptions of lattice field theories with bosons	**ANL**	HEP QuantiSED
Quantum-Enhanced Metrology with Trapped Ions for Fundamental Physics	**NIST**, ANL	HEP
Quantum chemistry algorithms to simulate plasma facing materials with NISQ devices	**GA**, ANL	FES
Two QAOA projects	**External collaborators**	DARPA ONISQ
Quantum circuit cutting	**ANL**, Atos	ANL LDRD
QuaC development	**ANL**	ANL LDRD

Computing Resources

ALCF Supercomputers

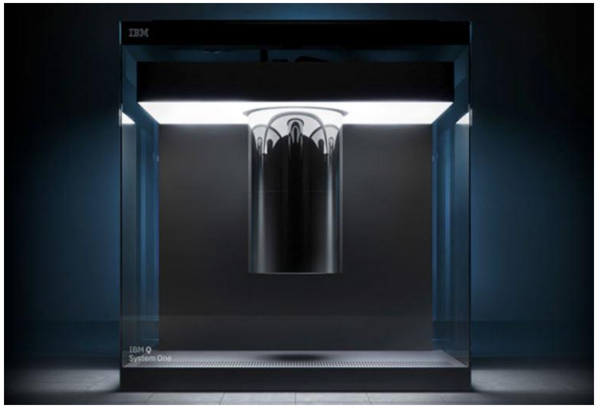
- Theta: Cray XC40, 12 Petaflops peak performance, 4,392 nodes/281,088 cores, 1 PB of memory
- Aurora: Exa-scale supercomputer in 2021

Atos: acquired QLM-35 September 2018

- Strategic partnership announced at SC18
- Internship program

IBM Q Hub

- Signed IBM Q hub agreement October 2018
- Access to 3rd generation 20 qubit (53 qubit soon) quantum computers on the cloud



Quantum computing projects

- Quantum simulators: development and optimization of quantum simulators for supercomputers. Simulators: Intel-QS, QuaC
- Solving various combinatorial optimization problems (Maxcut, community detection, graph partitioning, network alignment, graph coloring, maximum independent set). Scale up calculations using local search and multi-level methods
- Finding optimal optimization parameters for QAOA by using machine learning

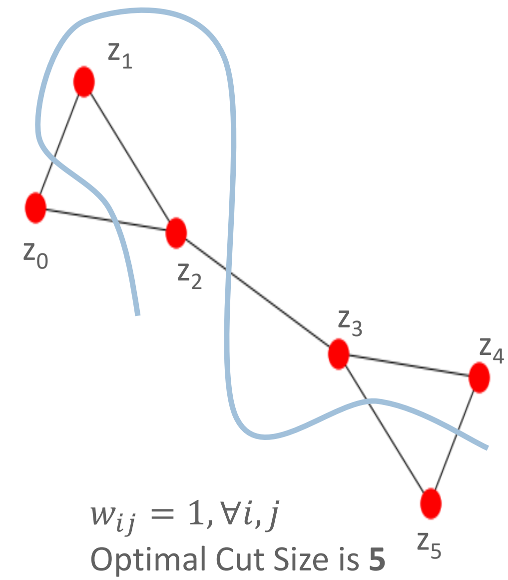
Large scale quantum simulations

- Ported and optimized for 10 PF Theta supercomputer to run 45 qubit simulations using Intel-QS
- Compress state amplitudes up to 10,000 times using SZ package which allowed 61 qubit simulation requiring 32 EB of memory (Theta has ~1 PB), SC19 paper
- Plans to port and optimize QuaC for Aurora exa-scale supercomputer. Ultimate goal using tensor slicing and amplitude compression to execute 100+ qubit simulations

Benchmark	Grover			Random Circuit Sampling				QAOA		QFT
Number of Qubits (Memory Requirement)	61 (32 EB)	59 (8 EB)	47 (2 PB)	5×9 (512 TB)	6×7 (64 TB)	6×6 (1 TB)	7×5 (512 GB)	43 (128 TB)	42 (64 TB)	36 (1 TB)
Number of Gates	314	310	305	227	261	165	208	344	336	3258
Number of Nodes	4096	4096	128	1024	128	1	1	256	128	1
Total System Memory (Sys Mem / Req.)	768 TB (0.002%)	768 TB (0.009%)	24 TB (1.17%)	192 TB (37.5%)	24 TB (37.5%)	192 GB (18.75%)	192 GB (37.5%)	48 TB (37.5%)	24 TB (37.5%)	192 GB (18.75%)
Total Time (Hour)	8.14	3.48	0.49	4.87	8.64	7.96	6.23	5.83	8.65	78.98
Compression Time	1.87%	4.59%	2.04%	55.79%	40.26%	59.10%	58.57%	44.97%	41.02%	57.86%
Decompression Time	1.87%	3.73%	4.08%	31.47%	22.19%	33.78%	30.59%	27.64%	25.52%	37.68%
Communication Time	32.7%	20.98%	36.73%	0.12%	0.57%	0.02%	0.03%	0.22%	0.23%	2.56%
Computation Time	63.47%	70.70%	57.15%	12.60%	36.97%	7.08%	10.8%	27.16%	33.22%	1.9%
Time per Gate (Sec)	93.34	40.49	5.78	64.69	119.22	173.65	107.86	61.02	92.64	87.27
Simulation Fidelity	0.996	0.996	1	0.987	0.993	0.933	0.985	0.999	0.999	0.962
Compression Ratio	7.39×10^4	8.26×10^4	1.06×10^4	6.03	9.40	8.16	10.05	4.85	9.25	21.34

Combinatorial Optimization Problems

- **Combinatorial problems**: find a grouping, ordering, or assignment of a ***discrete, finite set*** of objects that satisfies given conditions.
- **Applications**: logistics, supply chain optimization, security, design & control (DOE application: design of meta materials, control of wild-fire fighting, design of experiments)
- **Graph MaxCut**: partition the vertices into into two disjoint subsets such that the total weight of edges connecting the two subsets is maximized. Formally, $\begin{array}{c} \max \frac{1}{2} \sum_{i<j} w_{ij}(1 - z_i z_j) \\ s.t\ z_i \in \{1, -1\}, \forall i \in [n] \end{array}$

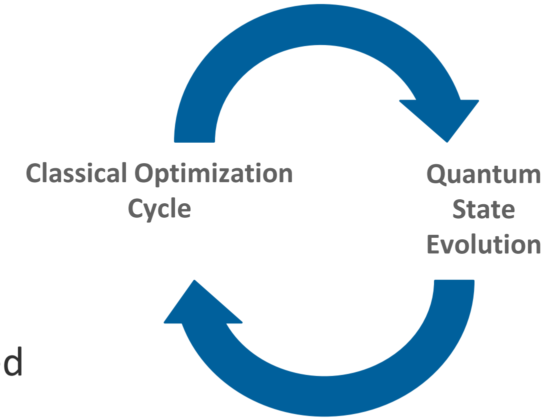


$w_{ij} = 1, \forall i, j$
Optimal Cut Size is **5**

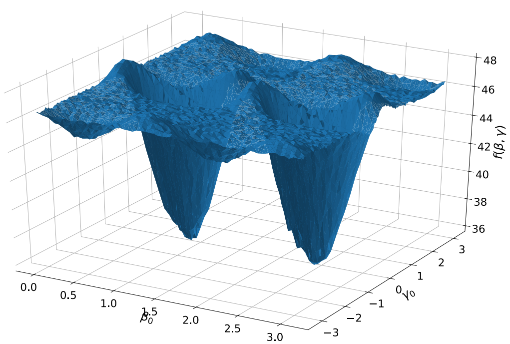
- Other combinatorial problems of interest: community detection and graph partitioning
- **Challenge**: solution space grows exponentially in the problem size.
- Approximation ratio, $\alpha = \frac{C(z)}{max\ \mathrm{C(z)}}$

Quantum Approximate Optimization Algorithm (QAOA)

- A variational hybrid quantum-classical algorithm:



1. Encode the classical objective function in a cost Hamiltonian by promoting each binary variable z_i into a quantum spin σ_i^z
2. Generate a variational wave function ($2p$ parameters) by repeated application (p times for depth p circuit) of the cost Hamiltonian and the transverse field mixer Hamiltonian $H_m = \sum_i \sigma_i^x$ on the prepared uniform superposition state $|\psi_p(\gamma,\beta)\rangle = \left(\prod_{i=\{1,..,p\}} e^{-i\beta_i H_m} e^{-i\gamma_i H_c}\right)|\psi\rangle$
3. Maximize the expected energy of the cost Hamiltonian by new choice of variational parameters γ, β through a classical optimization loop.



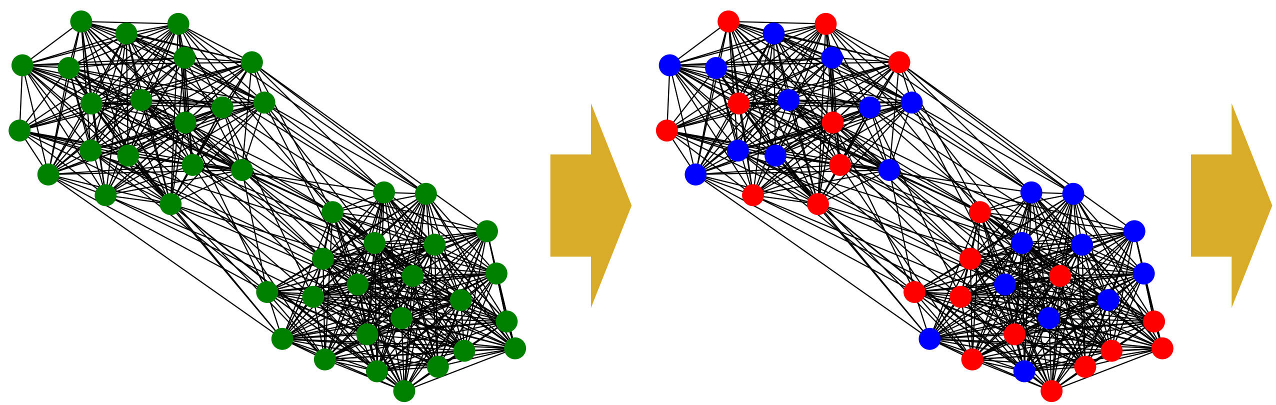
Solve QAOA optimization problems at scale

- Use hybrid/decomposition (local search and multi-level) approaches to solve large NP-hard combinatorial optimization problems
- Implemented on IBM Q hub and D-Wave quantum computers
- The challenge is that only 20 qubits are available on IBM Q quantum devices
- Applied to real-world networks of up to 10,000 nodes using only 16-20 qubits
- Published in Advanced Quantum Technology, IEEE Computer, SC18 Post Moore's Era Supercomputing workshop

Quantum Local Search

- Local search applied to Community Detection
  - **Start with some initial solution**
  - Search its neighborhood on a NISQ device
  - If a better solution is found, update the current solution

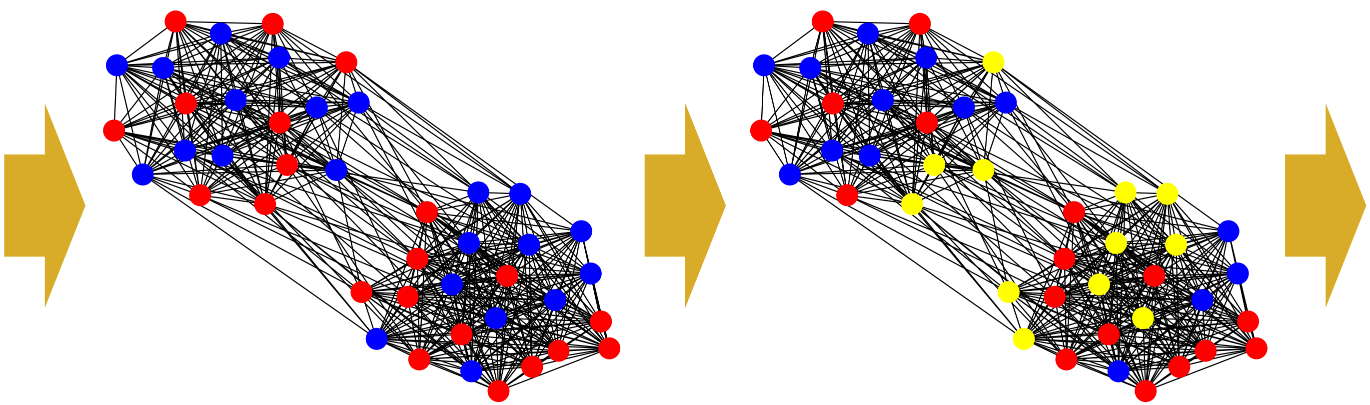
Part 1 (fixed)
Part 2 (fixed)
Optimized on NISQ device



Quantum Local Search

- Local search
  - Start with some initial solution
  - **Search its neighborhood on a NISQ device**
  - If a better solution is found, update the current solution

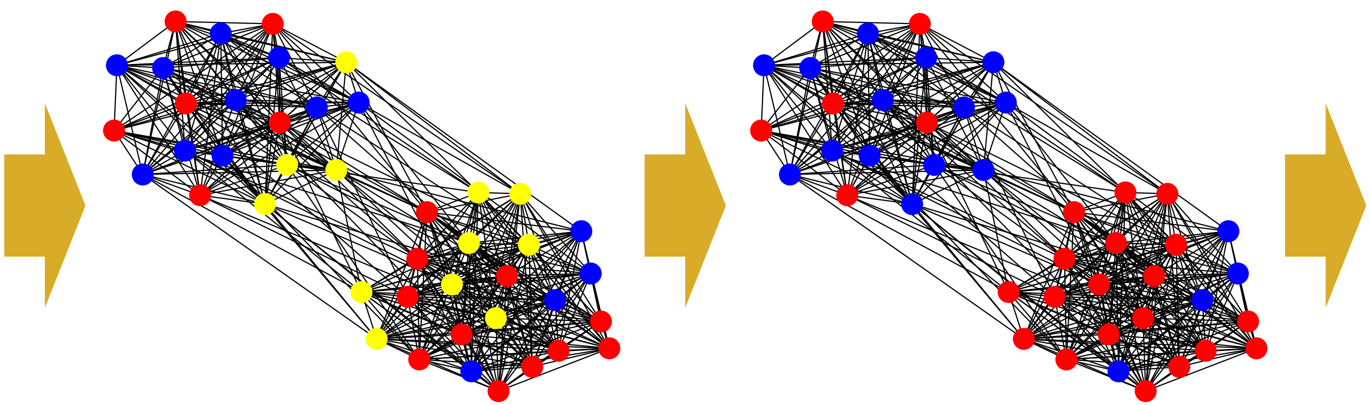
Part 1 (fixed)
Part 2 (fixed)
Optimized on NISQ device



Quantum Local Search

- Local search
  - Start with some initial solution
  - Search its neighborhood on a NISQ device
  - **If a better solution is found, update the current solution**

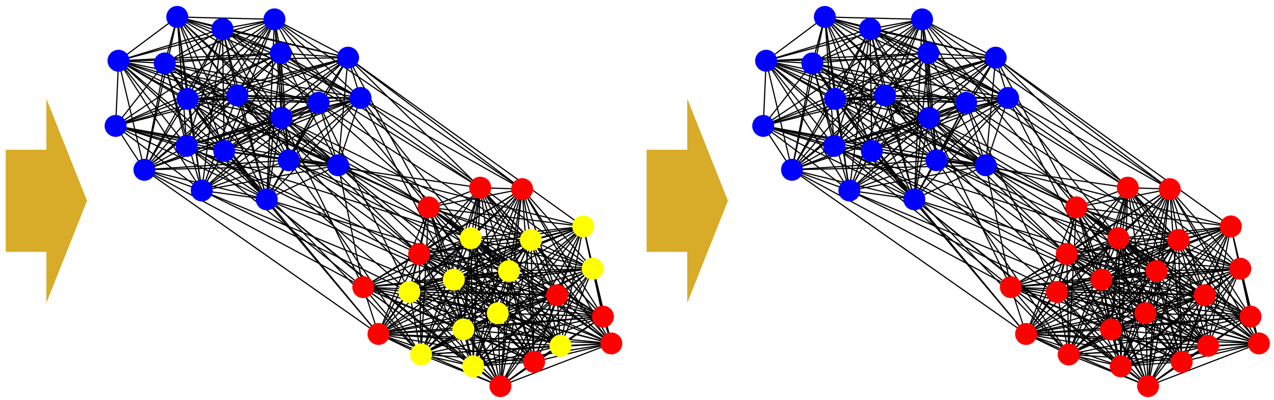
Part 1 (fixed)
Part 2 (fixed)
Optimized on NISQ device



Quantum Local Search

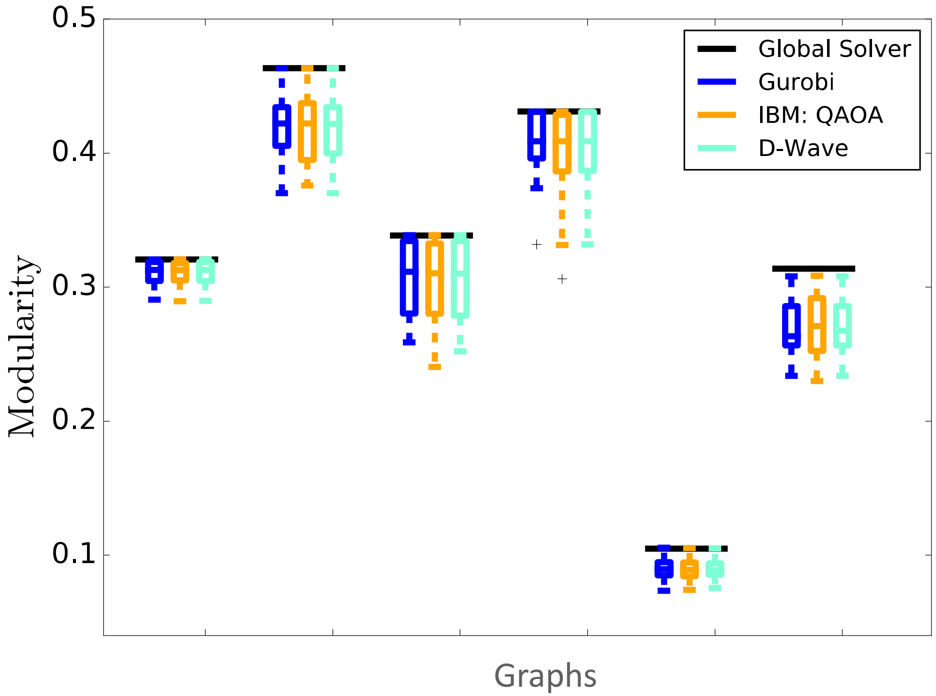
- Local search
  - Start with some initial solution
  - Search its neighborhood on a NISQ device
  - **If a better solution is found, update the current solution**

- Part 1 (fixed)
- Part 2 (fixed)
- Optimized on NISQ device



Quantum Local Search Results

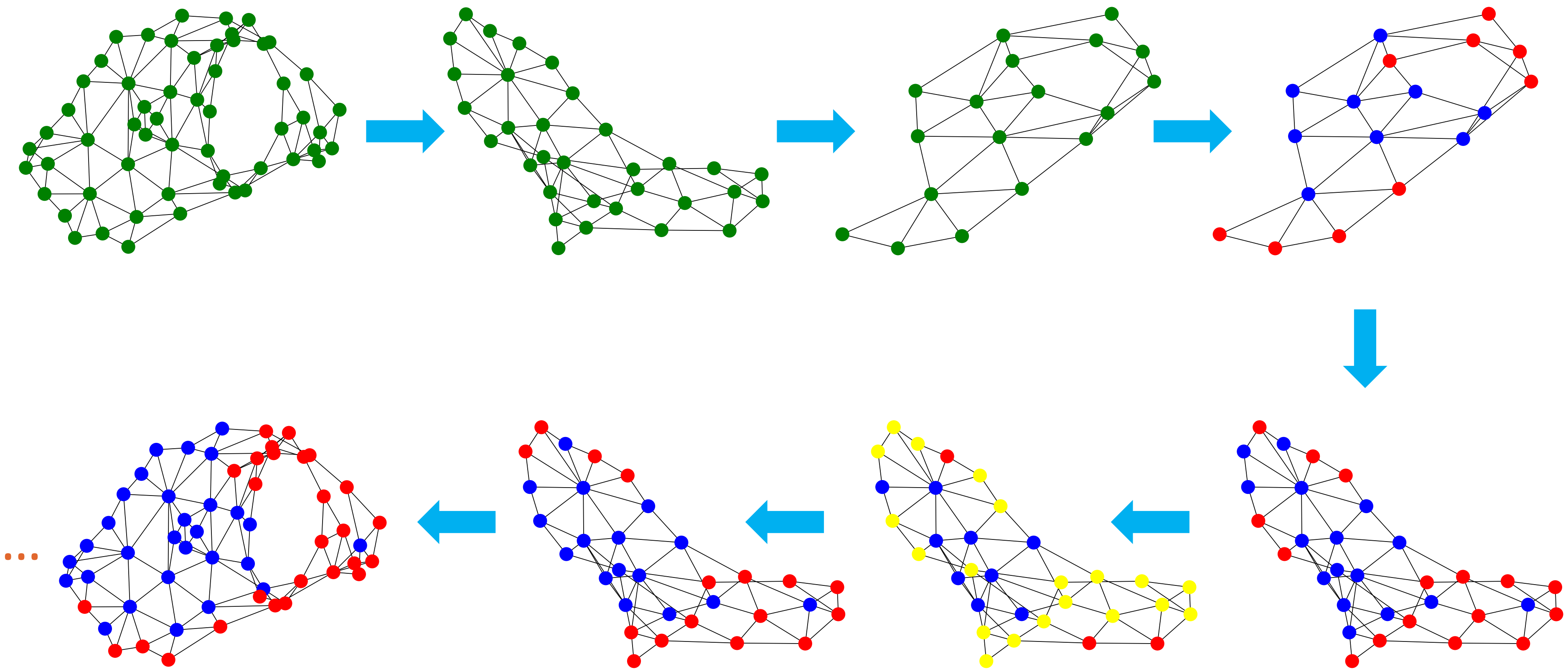
- Use IBM 16 Q Rueschlikon and D-Wave 2000Q as subproblem solvers
- Classical subproblem solver (Gurobi) used for quality comparison
- Fix subproblem size at 16
- Used real-world networks from The Koblenz Network Collection with up to 400 nodes



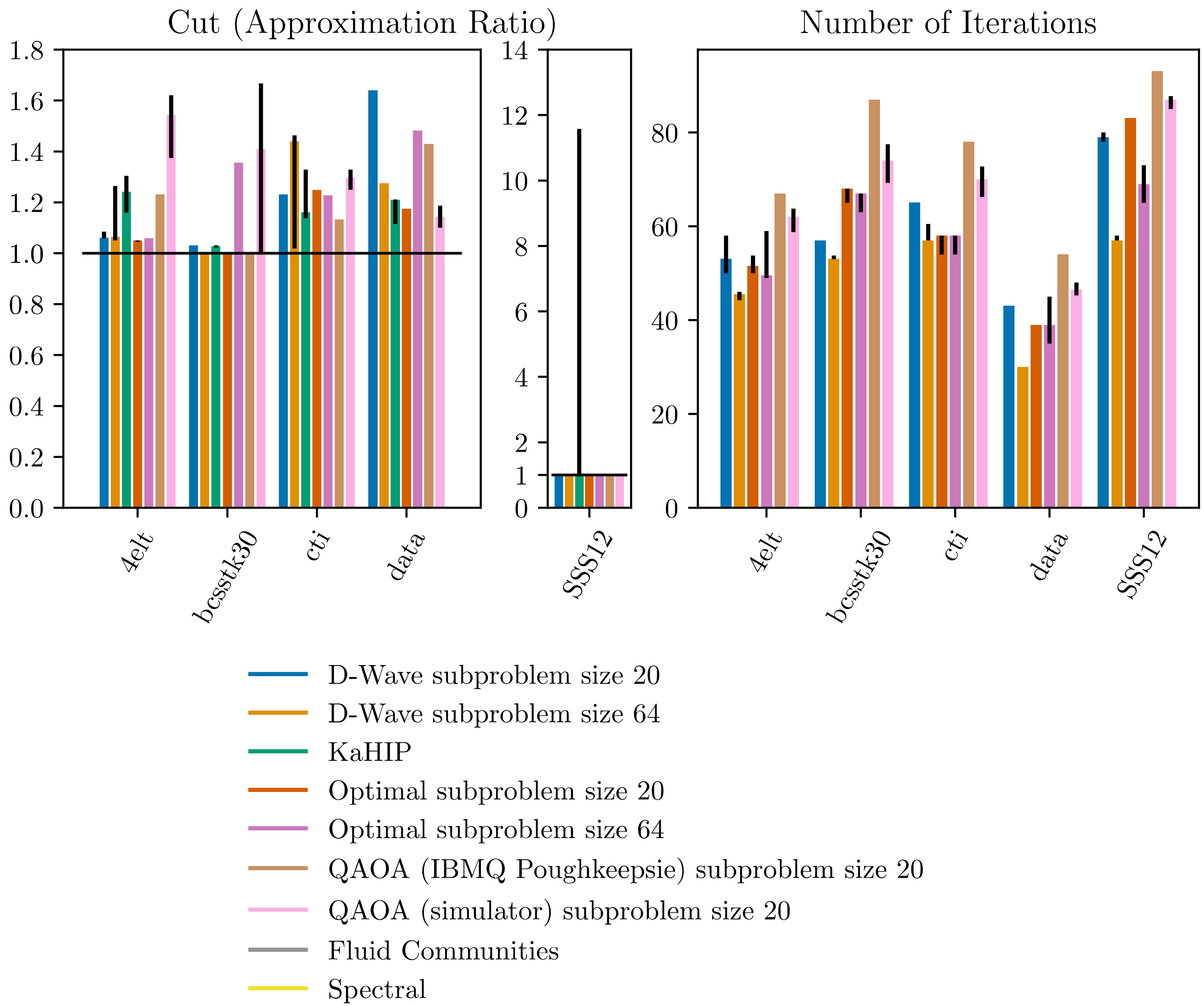
Multiscale QLS (MS-QLS)

- What if our problem is too large to effectively cover with local search iterations?
- Solving 400 node graph with QLS takes ~30 calls to quantum subproblem solver
- The solution is Multiscale Approach
  - Iteratively coarsen the problem
  - Solve coarse problem small enough on NISQ device
  - Uncoarsen
    - Iteratively project solution onto finer level
    - Refine it by running iterations of QLS done using NISQ device

Multiscale QLS (MS-QLS)



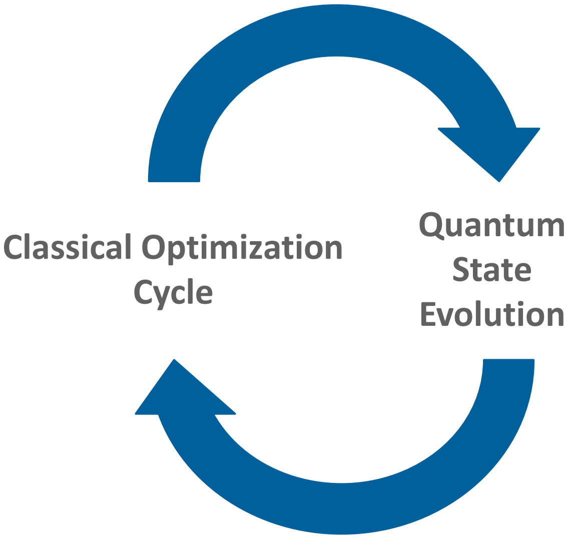
Quantum Local Search Results



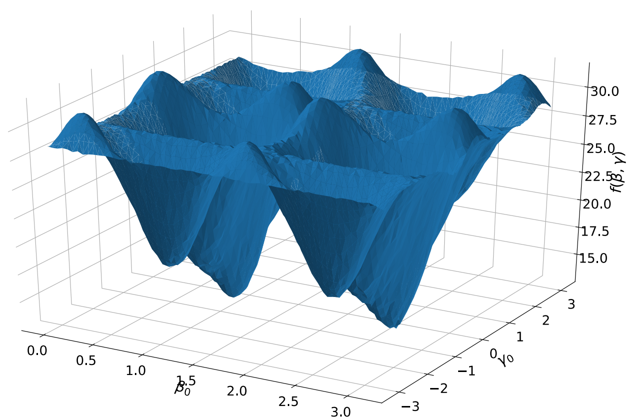
Results

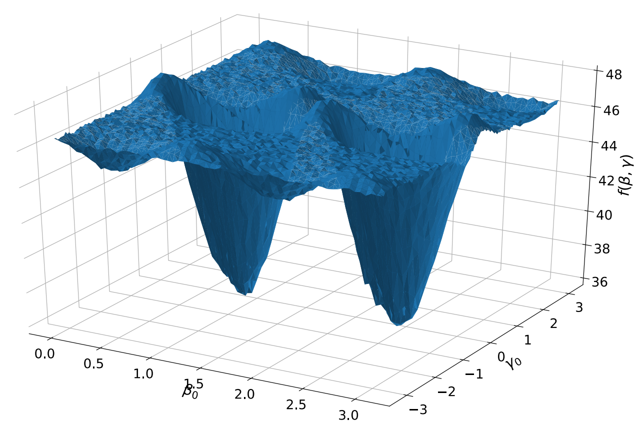
- Solve 22k node graphs **with just 20 qubits** in ~ 100 iterations
- Projected time is seconds – given better hardware
- **Competitive with classical state-of-the-art in terms of quality of the solution and speed** for real-world-scale problems

QAOA optimization algorithm



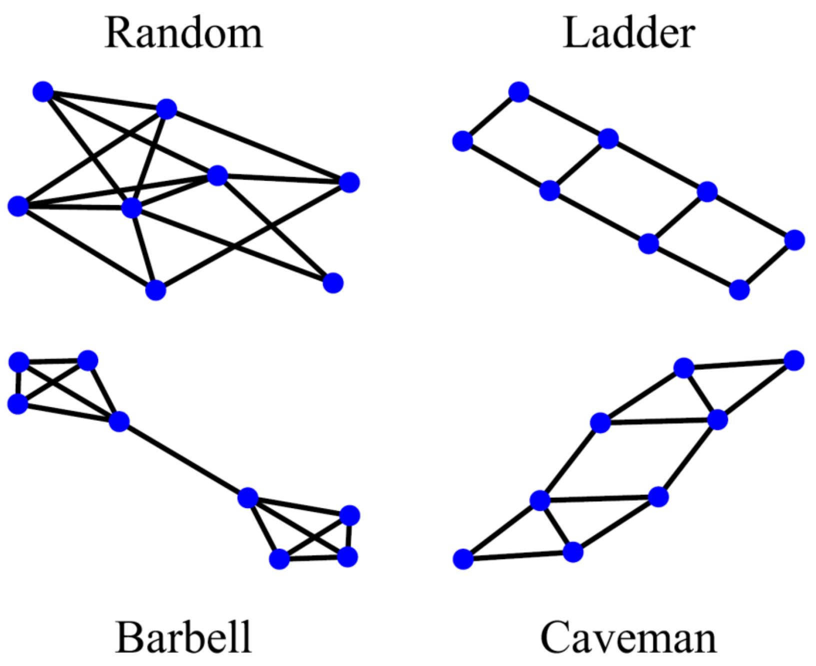
- It is important to be able to find quickly beta and gamma parameters
- It can be in some cases NP-hard problem





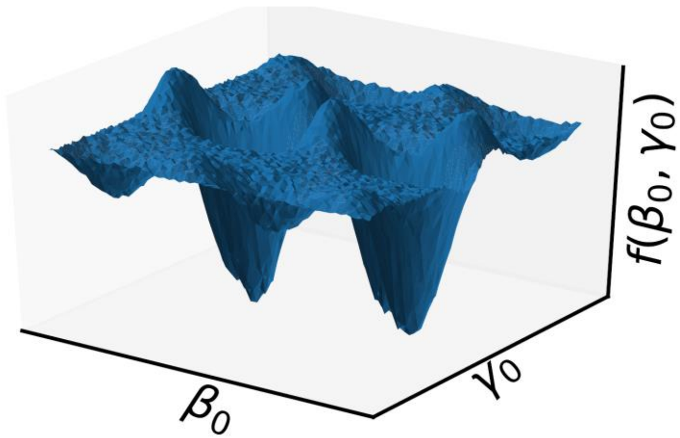
Finding QAOA parameters using machine learning

- Use machine learning methods (including Bayesian optimization) and sequential optimization to find optimal parameters beta and gamma for QAOA applied to Maxcut and community detection
- Build machine-learned mixer Hamiltonian using DeepHyper (reinforcement learning package) developed by Prasanna Balaprakash
- Looking for a collaboration with other national laboratories in the area of ML-assisted quantum computing

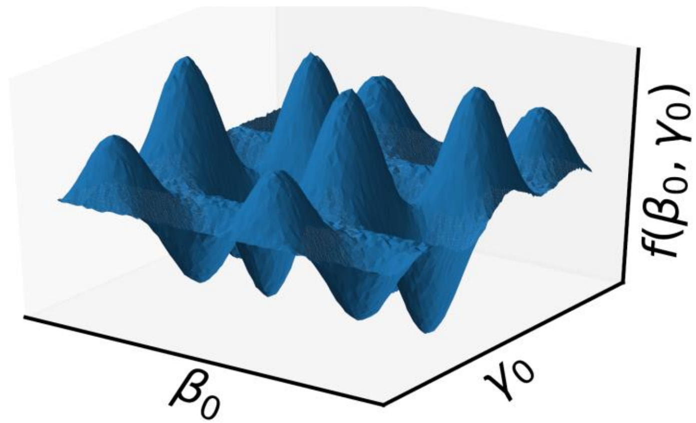


Finding QAOA parameters using machine learning

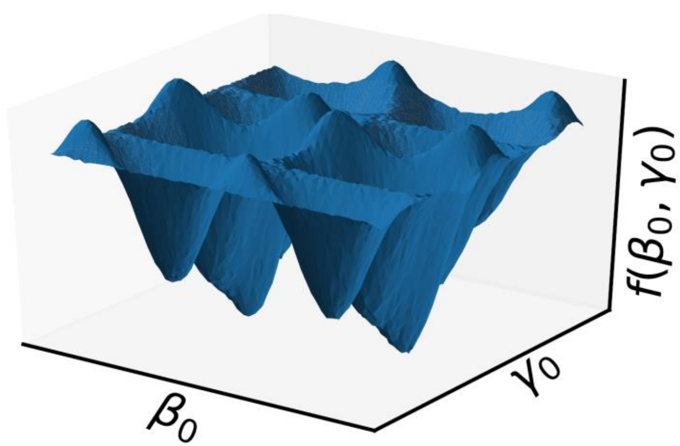
Random



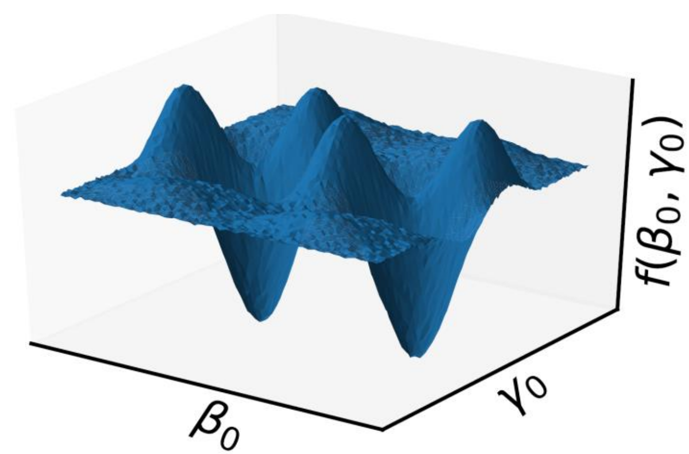
Ladder



Barbell

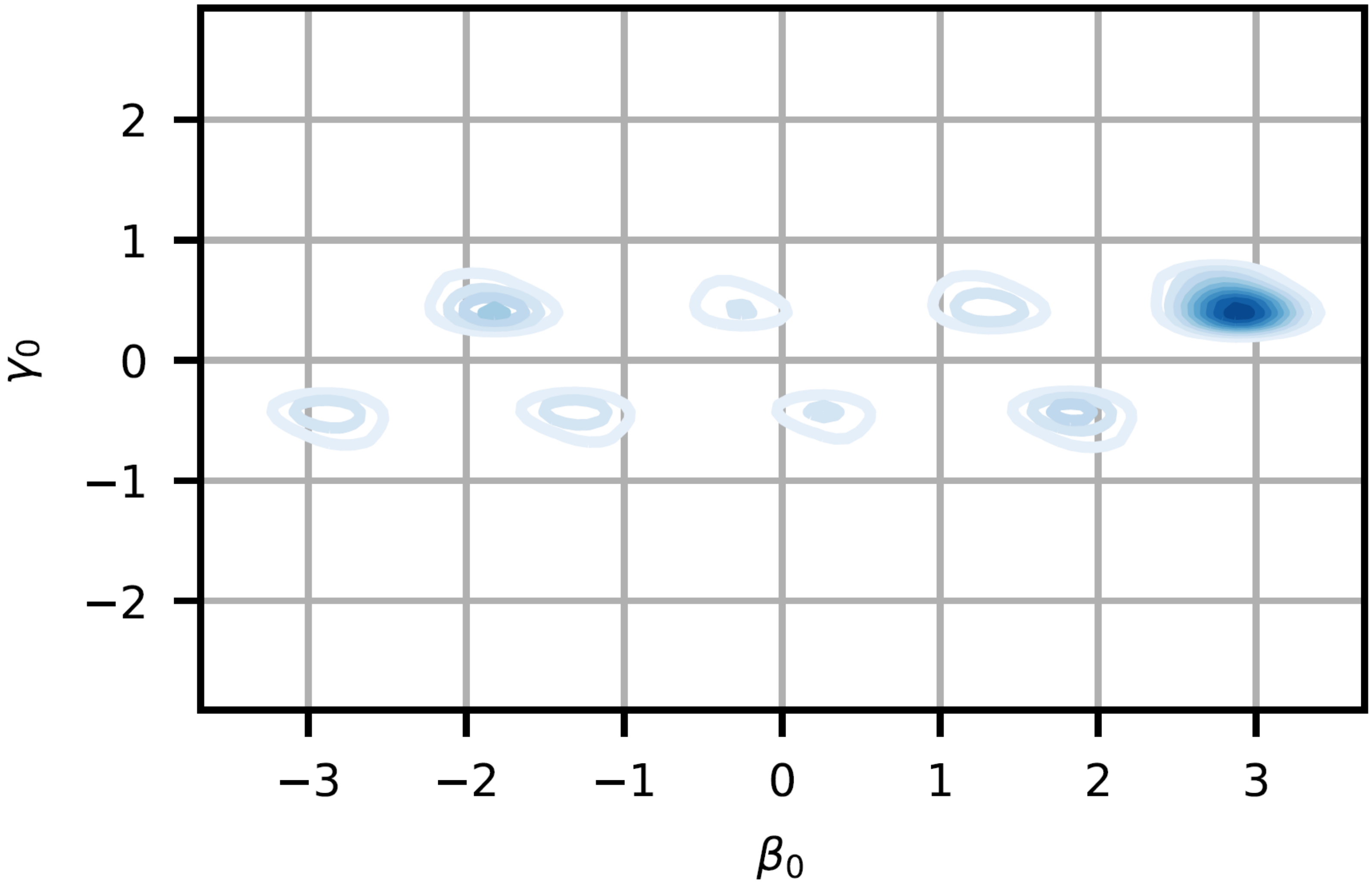


Caveman

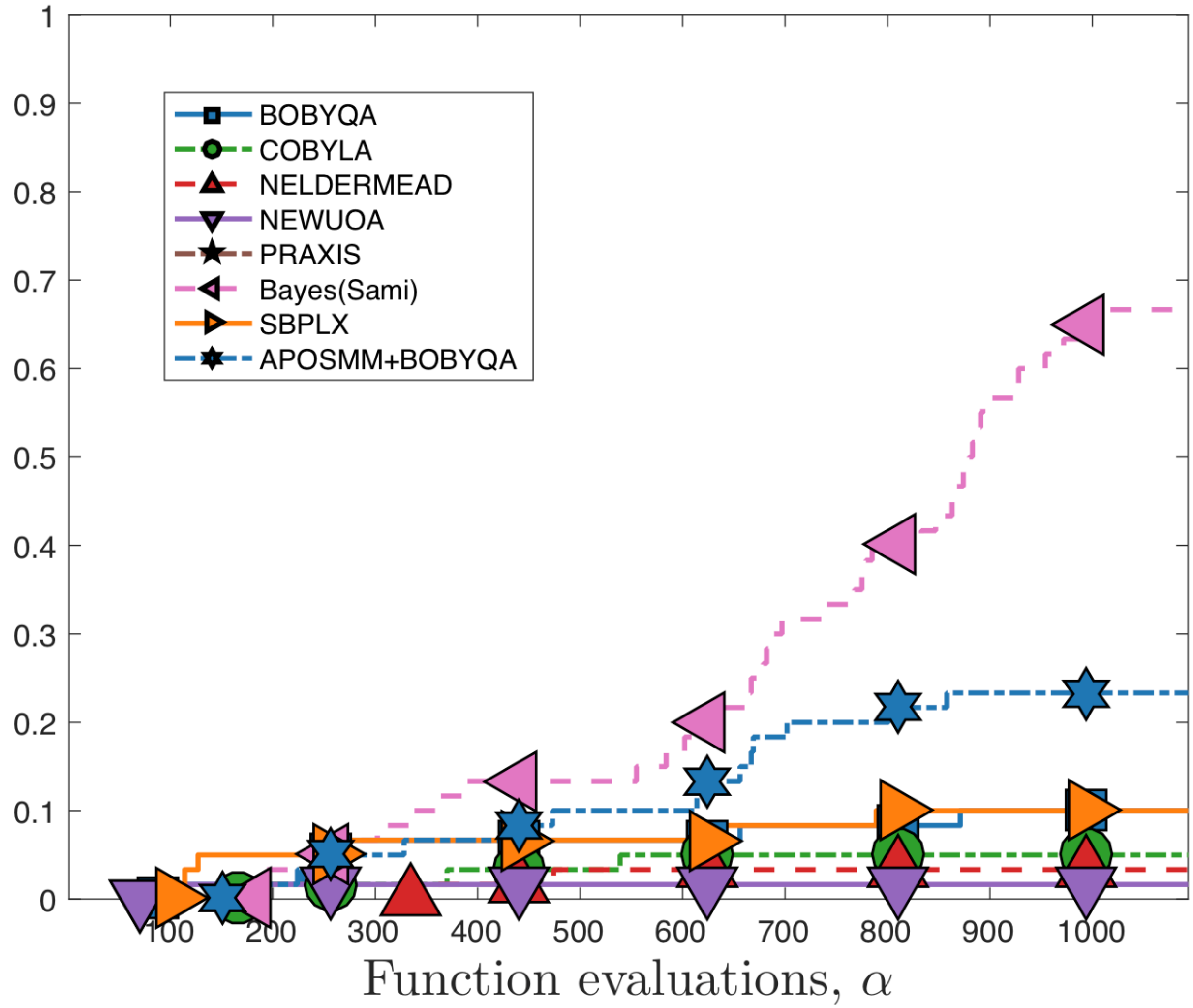


Finding QAOA parameters using machine learning

Density projection for various instances



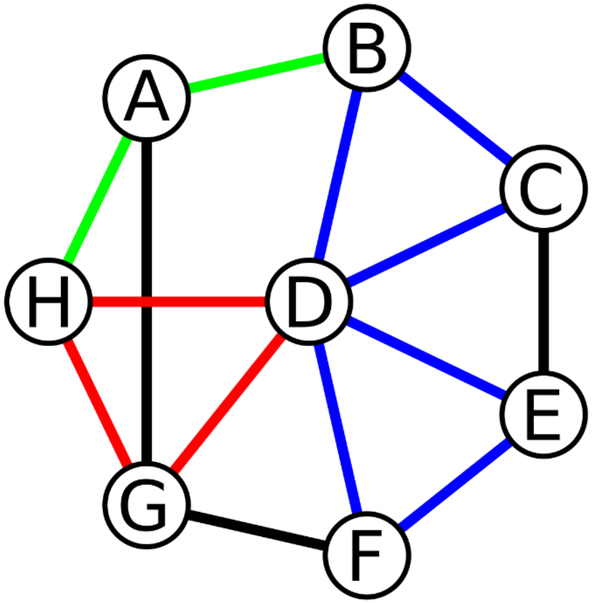
Results



BOBYQA
COBYLA
NELDERMEAD
NEWUOA
PRAXIS
Bayes(Sami)
SBPLX
APOSMM+BOBYQA
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
100
200
300
400
500
600
700
800
900
1000
Function evaluations, α

Analytical formulas

- "The Quantum Approximation Optimization Algorithm for MaxCut: A Fermionic View", Zhihui Wang, Stuart Hadfield, Zhang Jiang, and Eleanor G. Rieffel https://arxiv.org/pdf/1706.02998.pdf
- Formula to find parameters of a special case Maxcut, the ring of disagrees, or the 1D antiferromagnetic ring



$$F = 2\sin(4\beta)\sin(4\gamma)\sum_k \sin^2\theta_k \tag{57}$$

$$= \begin{cases} n\sin(4\beta)\sin(4\gamma) & \text{for } n = 2 \\ \frac{n}{2}\sin(4\beta)\sin(4\gamma) & \text{for } n > 2\,. \end{cases} \tag{58}$$

The optimal angles are $(\gamma_1^*, \beta_1^*) = \pi \cdot (3/8, 1/8)$ or $\pi \cdot (1/8, 3/8)$.

QIS Team at Argonne

Co-PIs





Postdoctoral fellow

Computing interns
Spring, Summer ‘19





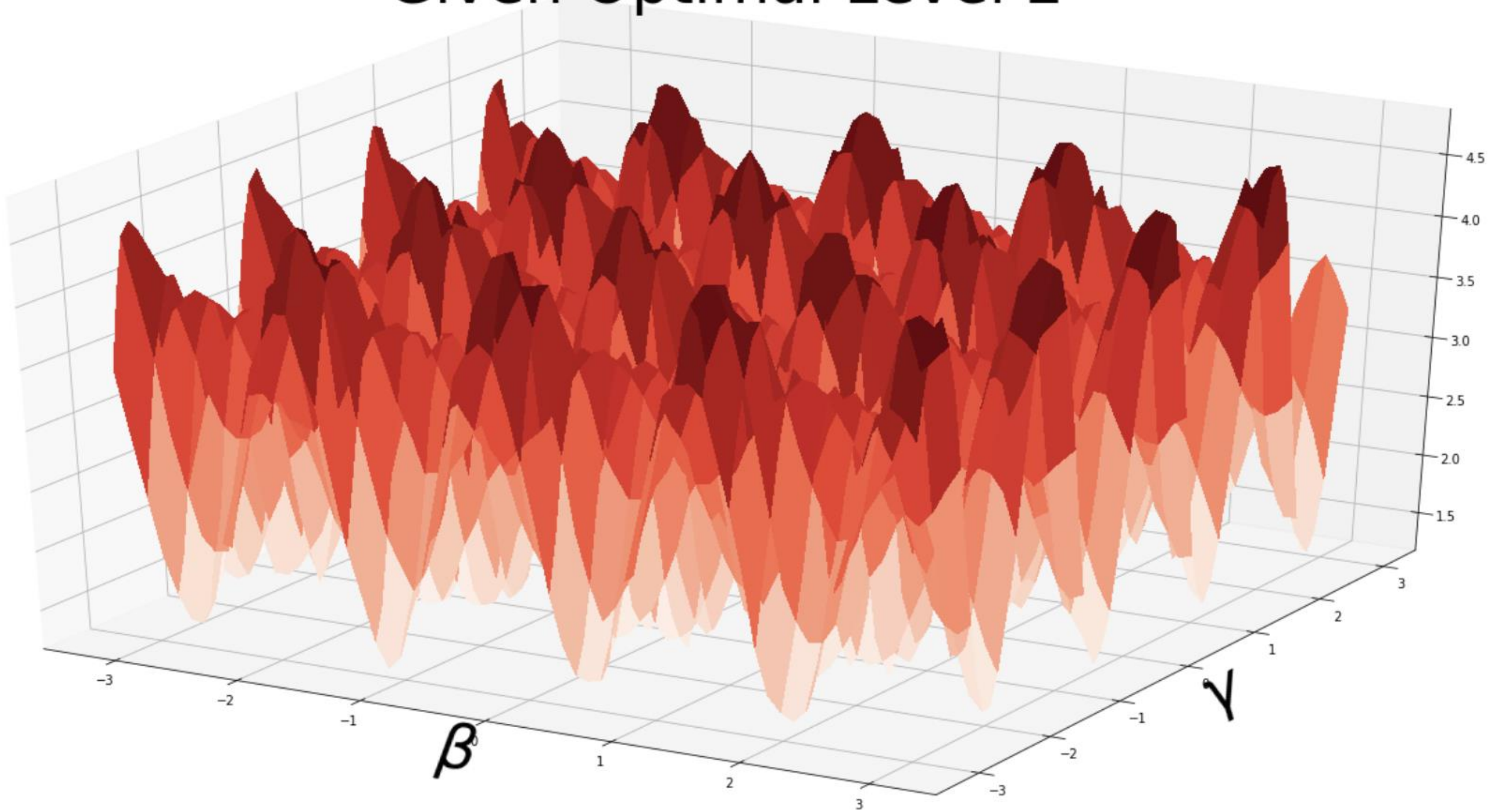
QAOA team

Acknowledgements

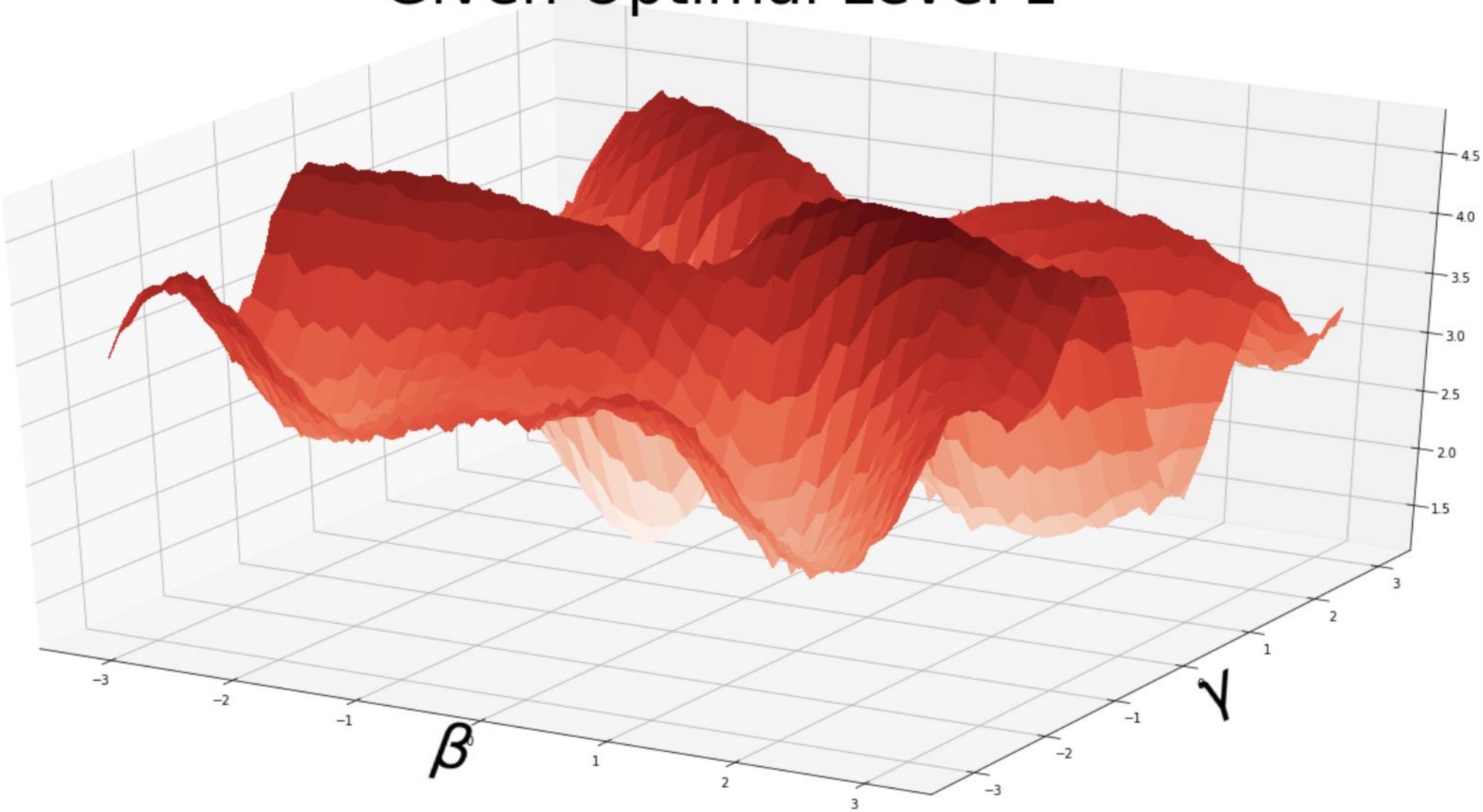
- This research used the resources of the Argonne Leadership Computing Facility, which is a U.S. Department of Energy (DOE) Office of Science User Facility supported under Contract DE-AC02-06CH11357.
- We gratefully acknowledge the computing resources provided and operated by the Joint Laboratory for System Evaluation (JLSE) at Argonne National Laboratory.
- We acknowledge the NNSAs Advanced Simulation and Computing (ASC) program at Los Alamos National Laboratory (LANL) for use of their Ising D-Wave 2X quantum computing resource and D-Wave Systems Inc. for use of their 2000Q resource.
- The LANL research contribution has been funded by LANL Laboratory Directed Research and Development (LDRD). LANL is operated by Los Alamos National Security, LLC, for the National Nuclear Security Administration of the U.S. DOE under Contract DE-AC52-06NA25396.
- We acknowledge access to the IBM Q hub at ORNL
- Clemson University is acknowledged for generous allotment of compute time on Palmetto cluster.

Argonne National Laboratory is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC.

Energy Landscape of QAOA Level 1, Given Optimal Level 2



Energy Landscape of QAOA Level 2, Given Optimal Level 1



Learning a variational Circuit Optimizer

with Deep Reinforcement Learning

- **Can we learn a general optimizer that performs well (i.e., find optimal variational parameters, or suboptimal with high approximation ratio) on new graph instances?**

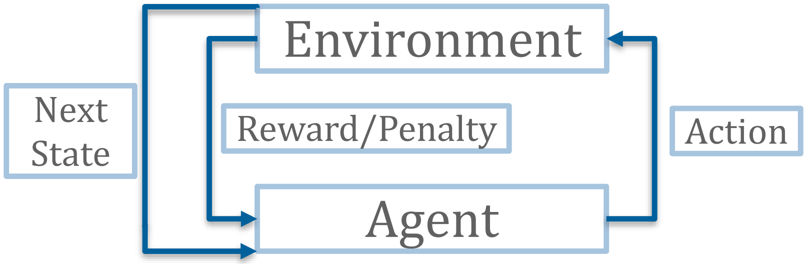
- General iterative optimizer for continuous unconstrained problems,

given: objective function f
$x_0 \leftarrow \textit{random point in the domain}$
for $i = 1,2, \ldots$
$\Delta x \leftarrow \mathcal{G}(x_0, x_1, .., x_{i-1})$
if stopping condition is met,
return x for which f is max
end
$x_i = x_{i-1} + \Delta x$
end for

Gradient Descent:
$\Delta \mathrm{x} = -\gamma \nabla f(x^{(i-1)})$

Newton's Method:
$$\Delta \mathrm{x} = \frac{-\nabla f(x^{(i-1)})}{\mathbb{H}[f(x^{(i-1)})]}$$

- Basic reinforcement learning framework,



- Modeled as a Markov Decision Process (MDP)