4-1-2019

Quantum Local Search for Graph Community Detection

Ruslan Shaydulin
Clemson University

Hayato Ushijima-Mwesigwa
Clemson University

Ilya Safro
Clemson University

Susan Mniszewski
Los Alamos National Lab

Quantum Local Search for Graph Community Detection

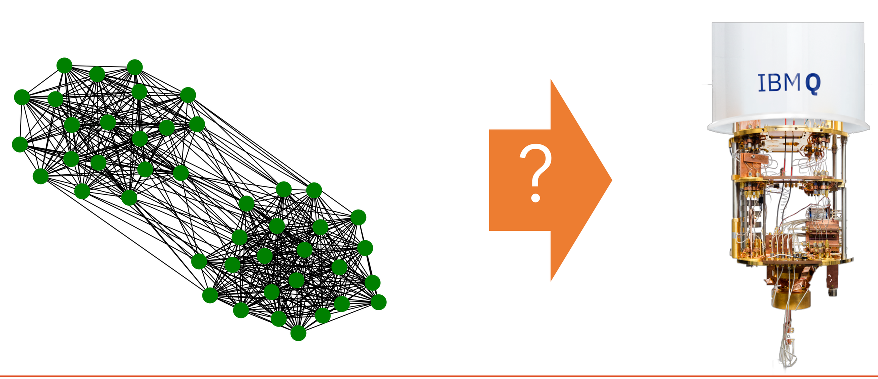
Ruslan Shaydulin, Clemson University
Hayato Ushijima-Mwesigwa, Clemson University
Ilya Safro, Clemson University
Susan Mniszewski, Los Alamos National Lab
Yuri Alexeev, Argonne National Lab

Challenge
1

- Near-term Quantum Computers (QC) are expected to have **small number of noisy, low-quality qubits**
- These computers are commonly called NISQ – Noisy Intermediate-Scale Quantum – Computers
- They have or are expected to have **50-200 qubits**, noise levels low enough on only run **tens to hundreds of gates**

How can we take advantage of near-term quantum computers?

- Typically, algorithms (both quantum and classical) look at a problem "as a whole". The whole problem (e.g. a social network) is **too large to fit** on a NISQ device!

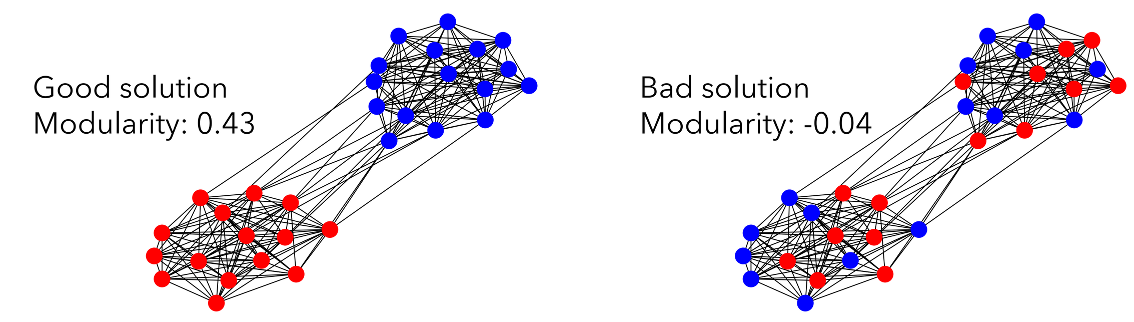


Solution: **decompose the problem**

Community Detection
2

Modularity maximization

- Also known as graph clustering
- Modularity is "the quality" of detected community structure in the network



- Mathematically, modularity is "the number of edges falling within groups minus the expected number in an equivalent network with edges placed at random"
- Formally:

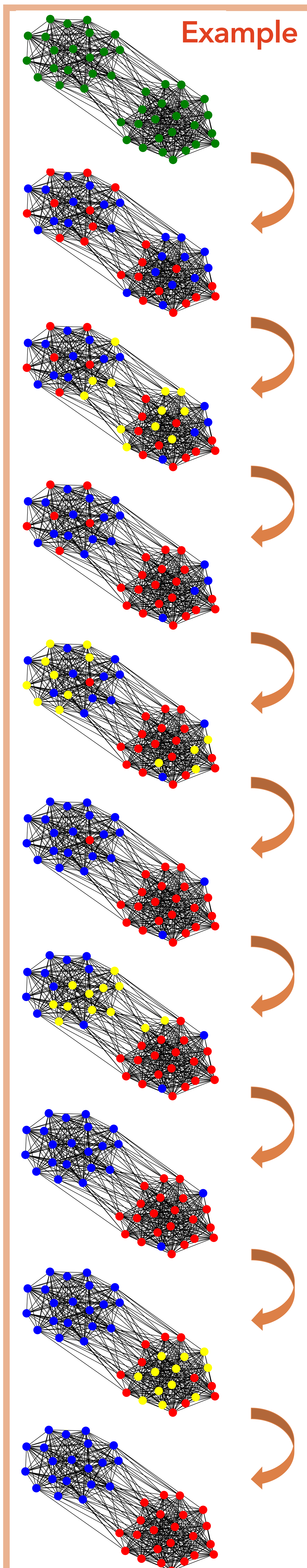
Actual number of edges

Expected number of edges

$$\begin{aligned} &\underset{s}{\text{maximize}} && \frac{1}{4|E|}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m})s_i s_j \\ &\text{subject to} && s_i \in \{-1,+1\} \\ &\text{where} && k_i \text{ is degree of vertex } i, m = \frac{1}{2}\sum_i k_i \\ &&& A \text{ is an adjacency matrix of } G \end{aligned}$$

Community assignment of vertex i

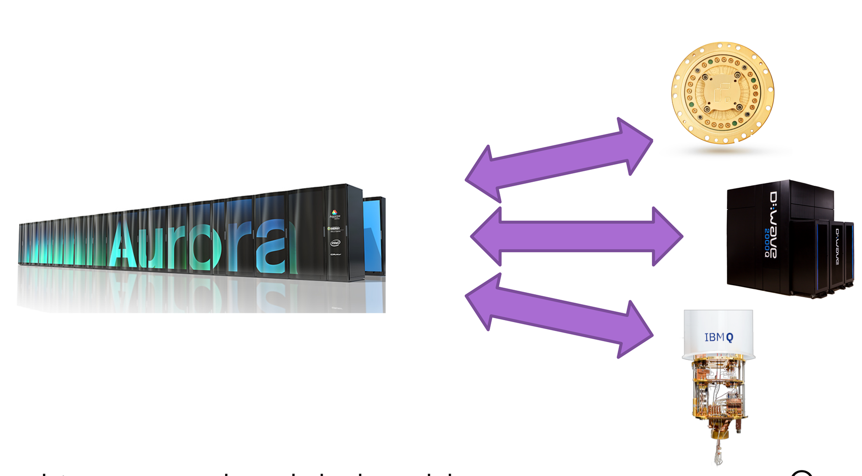
Example
9



Quantum Local Search
3

Our approach

- Local search
  - Start with some initial solution
  - Search its neighborhood on a NISQ device
  - If a better solution is found, update the current solution
- Neighborhood search (subproblem) can be encapsulated, making the framework architecture-agnostic and extendable to new architectures as they become available
- We implement subproblem solvers using IBM Q and D-Wave backends
- Provides a path to integrating heterogenous NISQ devices into HPC environments



Classical machine stores the global problem and orchestrates local search by sending small subproblems to quantum solvers

Quantum computers solve small subproblems

Quantum Local Search: in Detail
4

Algorithm 1 Community Detection

```
solution = initial_guess(G)
while not converged do
    X = populate_subset(G)
    // using IBM UQC or D-Wave QA
    candidate = solve_subproblem(G, X)
    if candidate > solution then
        solution = candidate
```

Subset selection

- Modularity (global problem):

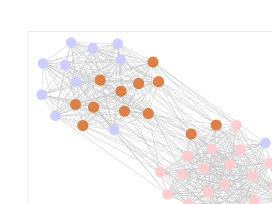
$$\underset{s}{\text{maximize}} \quad \frac{1}{4|E|}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m})s_i s_j = \sum_{ij} B_{ij} s_i s_j$$

- Gain from moving a vertex from one community to another can be easily computed and depends only on neighbors
- Approach: at each step take highest gain vertices

Subproblem (neighborhood search)

- Local subproblem: fix assignment of vertices not in the subset and encode as boundary condition

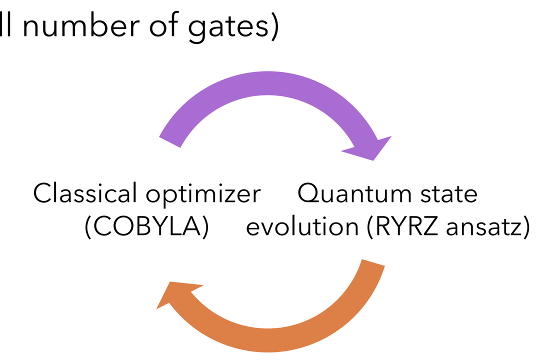
$$\begin{aligned} &\underset{s}{\text{maximize}} && \sum_{i>j|i,j\in X} 2B_{ij}s_i s_j + \boxed{\sum_{i\in X} C_i s_i} \\ &\text{where} && X \text{ is the subset we are optimizing} \end{aligned}$$



Quantum Approximate Optimization Algorithm (QAOA)
5

Local (subproblem) solver

- A heuristic that can be run on any gate-model (universal) quantum computer
- The problem is encoded as an objective Hamiltonian and solved by performing a quantum evolution
- Evolution is parametrized by variational parameters
- Classical optimizer finds optimal variational parameters
- Can be run on a NISQ computer (only requires small number of gates)
- Provides a path to quantum advantage [1]
- Allow COBYLA 100 iterations to train (optimize variational parameters) using simulator, after that run on quantum device



[1] E.Farhi, A. Harrow "Quantum Supremacy through the Quantum Approximate Optimization Algorithm" arXiv:1602.07674

Quantum Annealing (QA)
6

Local (subproblem) solver

- A heuristic
- Solves an optimization problem by encoding it as an Ising model Hamiltonian, with the ground state of that Hamiltonian corresponding to the global solution of the optimization problem

$$Q_s = \sum_{i>j|i,j\in X} 2B_{ij}s_i s_j + \sum_{i\in X} C_i s_i \leftarrow \text{already in Ising form!}$$

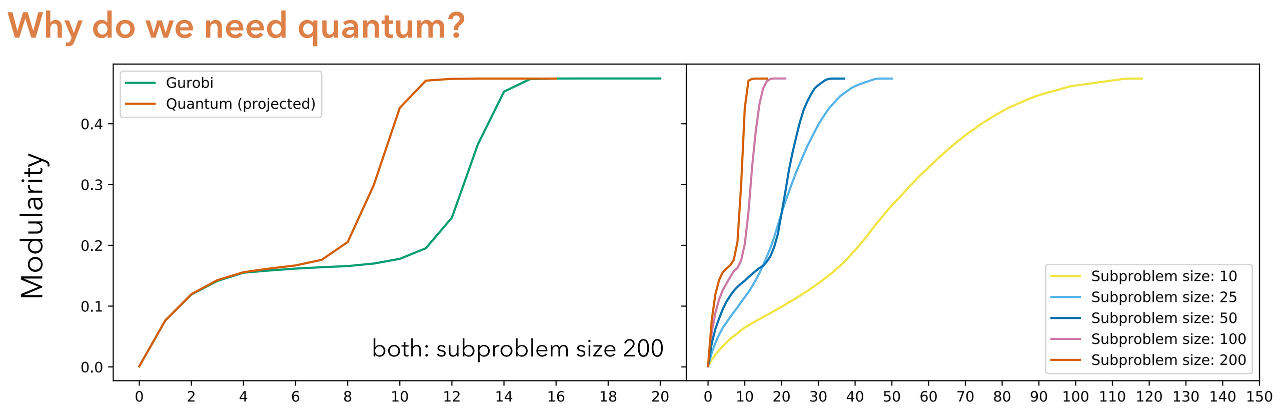
- QA finds the ground state of the objective Hamiltonian by performing a quantum evolution

$$\begin{aligned} H(t) &= (1-\tfrac{t}{T})H_F + \tfrac{t}{T}Q_S \\ H_F & \text{ - transverse field Hamiltonian} \\ Q_S & \text{ - problem Hamiltonian} \\ 0 \le & \; t \; \le T \end{aligned}$$

- We used D-Wave 2000Q (~2000 qubits) provided through Los Alamos National Lab

Motivation
7

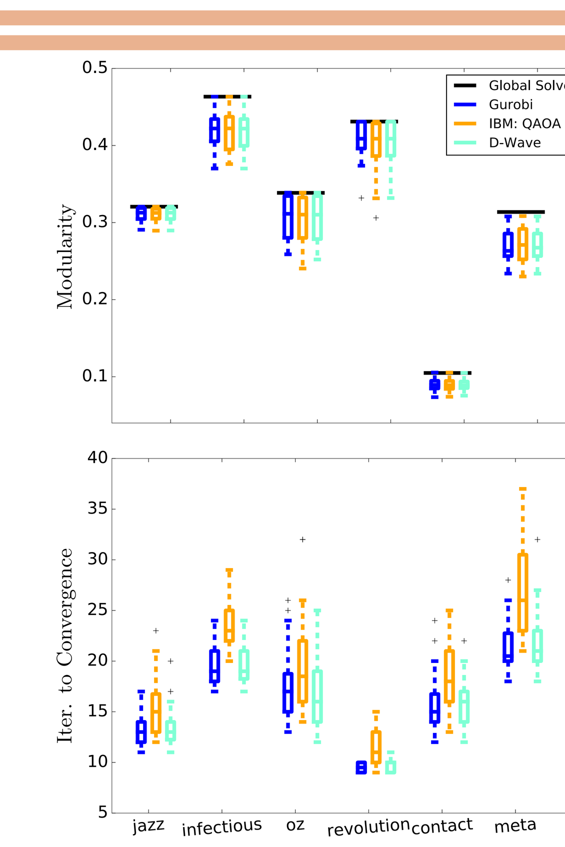
Why do we need quantum?



- We project the performance of QLS by using a classical optimization solver (Gurobi)
- State-of-the-art classical optimization solvers (Gurobi / CPLEX) **cannot provide the solution of desired quality quickly enough** even for subproblems small enough to potentially fit on NISQ-era quantum devices

Results
9

- Implemented QLS in Python, available on GitHub at http://bit.ly/QLSCommunity
- 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
- Dataset available online http://bit.ly/QLSdata
- QLS solves practically important problems of **up to 400 nodes** using only **16 qubits**
- All three methods demonstrate similar performance
- Quantum algorithms achieve results close to state-of-the-art



Full paper: arXiv:1810.12484