BERKELEY LAB

U.S. DEPARTMENT OF ENERGY

UNIVERSITY OF CALIFORNIA

A Quantum Approximate Optimization Algorithm for Track Reconstruction

Eric Rohm, Heather Gray

Physics Division
HEP.QPR Project

October 30, 2019

Charged Particle Track Reconstruction

- Particles passing through detectors can deposit energy (called “hits”) throughout their path
- In tracking, these hits are considered in order to reconstruct the trajectory of the original particle
    - Through pattern recognition, hits are assigned to specific particles in clusters which form “track candidates”
- In track fitting, the physical properties of the particles are determined from their trajectories

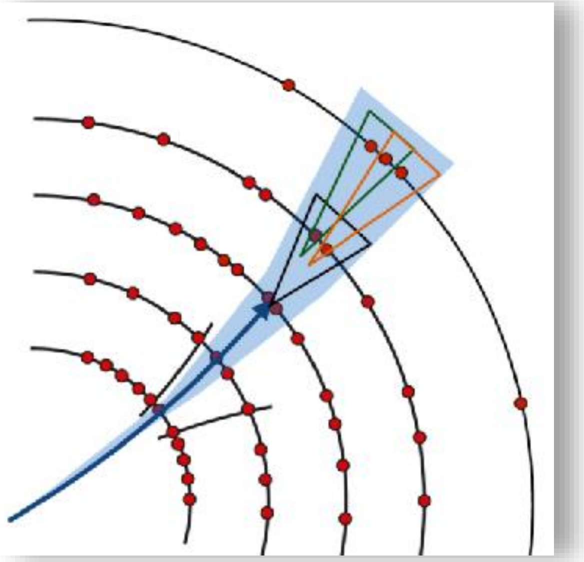

Figure 1. An illustration of an event. A charged particle’s trajectory through detector layers is reconstructed (Boser et. al. 2018).

Quantum Computing for Track Reconstruction: D-Wave

- Qallse
  - Developed as part of HEP.QPR project
  - Encodes pattern recognition problem form TrackML challenge into a QUBO and solves it using quantum annealing on D-Wave in addition to classical methods
  - Performance of D-Wave solver was not substantially better than classical solving methods

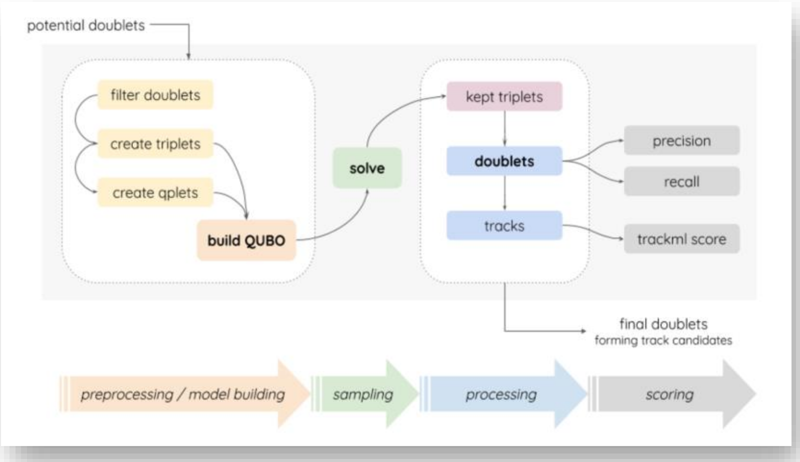

Figure 2. The process of formulating and solving QUBO models from TrackML datasets with Qallse (Linder 2019).

Quantum Approximate Optimization Algorithm (QAOA) Overview

- Hybrid classical-quantum algorithm that combines quantum circuits and classical optimization of those circuits
- Has only been used on gate-based machines
- Our project aims to convert track reconstruction problems to a form solvable by QAOA on gate-based machines

QAOA: Cost

- Primary goal is to minimize a set of two n-length parameters, **γ** and **β** which together parameterize a cost function
- Parameters are iteratively varied
- Cost function is evaluated in a quantum circuit to produce a cost value (the expectation value of cost function)
- Cost value then classically optimized to indicate if the varying is in a direction of greater or lesser cost
- Classical optimizer then returns updated parameters to the circuit for re-evaluation each iteration

QAOA: Solution

- Each iteration increases total number of gates and parameters, so accuracy improves with each iteration
- Once iterated the maximum number of times, the parameters (which have now been optimized) are used to define a solution

QAOA Formulation

$$C(z) = \sum_{\alpha=1}^{m} C_\alpha(z)$$

- *C(z)* is the objective function that contains the number of satisfied clauses *α* over *m* total clauses and *n* bit strings each of $z = z_1 z_2 \ldots z_n$
- Optimization seeks a bit string which satisfies every clause
- QAOA uses *C(z)* as a diagonalized operator in the computational basis state
- A state dependent upon the varying parameters is given by

$$|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle = U(B, \beta_p)\, U(C, \gamma_p) \cdots U(B, \beta_1)\, U(C, \gamma_1)\ |s\rangle$$

- Where *U(B,**β**)* and *U(C,**γ**)* are the products of commuting one-bit operators dependent on their respective parameters ***β*** and ***γ***

QAOA Formulation

$$F_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \langle \boldsymbol{\gamma}, \boldsymbol{\beta} | \, C \, | \boldsymbol{\gamma}, \boldsymbol{\beta} \rangle$$

- The cost value $F_p(\boldsymbol{\gamma},\boldsymbol{\beta})$ is the expectation value of C in the state $|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle$
- In QAOA, a number of iterations p is chosen along with an initial set of parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ which maximize F_p
- The quantum machine is used to prepare the actual state $|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle$
- The optimization of parameters is done classically

QAOA Formulation

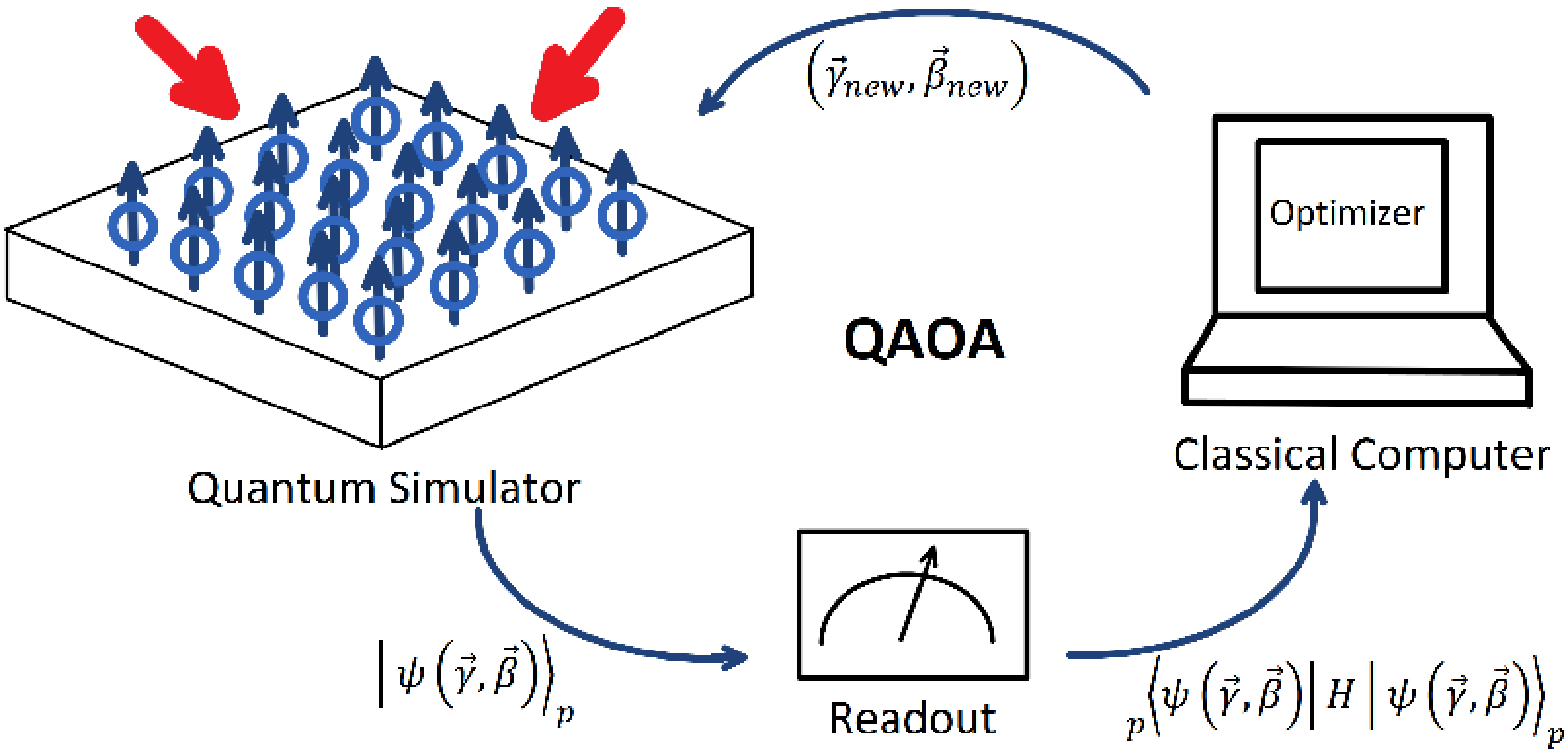

Figure 3. An illustration of the general process of QAOA. A superposition of bit strings is prepared on a quantum circuit, while each successive iteration of parameters is optimized classically then returned to the quantum machine (Ho, Hsieh 2018).

Potential of QAOA on Near-Term Devices

The Quantum Approximate Optimization Algorithm (QAOA) is designed to run on a gate model quantum computer and has shallow depth. It takes as input a combinatorial optimization problem and outputs a string that satisfies a high fraction of the maximum number of clauses that can be satisfied. For certain problems the lowest depth version of the QAOA has provable performance guarantees although there exist classical algorithms that have better guarantees. Here we argue that beyond its possible computational value the QAOA can exhibit a form of "Quantum Supremacy" in that, based on reasonable complexity theoretic assumptions, the output distribution of even the lowest depth version cannot be efficiently simulated on any classical device. We contrast this with the case of sampling from the output of a quantum computer running the Quantum Adiabatic Algorithm (QADI) with the restriction that the Hamiltonian that governs the evolution is gapped and stoquastic. Here we show that there is an oracle that would allow sampling from the QADI but even with this oracle, if one could efficiently classically sample from the output of the QAOA, the Polynomial Hierarchy would collapse. This suggests that the QAOA is an excellent candidate to run on near term quantum computers not only because it may be of use for optimization but also because of its potential as a route to establishing Quantum Supremacy.

(Farhi et al. 2016)

QAOA Software Compatible with Rigetti

- **[Grove]**
  - Includes pyQAOA, a Python module for running QAOA on Rigetti QVM
  - Separate modules for using pyQAOA on specific problem instances
  - No module for QAOA on QUBO
- **[EntropicaQAOA]**
  - Fully native support for Rigetti QVM and QPU
  - Apparently more versatile and customizable
  - Contains modules with use QAOA directly on QUBOs and Ising model

EntropicaQAOA

- Released only days ago (October 2019)
- Will be used in this project due to its built-in QUBO solving
- Allows for more customization and direct parameterization of Hamiltonians and cost functions

ENTROPICA LABS

Using QAOA on QUBO

- In EntropicaQAOA, a QUBO variable x_i is assigned with an i^{th} qubit
- For our purposes, each triplet in the QUBO will be assigned a qubit
- Hamiltonian is defined by coupling strengths of these triplets
- QAOA approximates the ground state of this Hamiltonian iteratively

Qaoala

- Codebase for this is on GitHub:
  - https://github.com/erohm/qaoala
- Qaoala aims to integrate the QUBO model building capabilities of Qallse with the QAOA Hamiltonian minimization of EntropicaQAOA
- A work in progress!

Qaoala: Proposed Process

1. TrackML problem → QUBO
   - Built via Qallse

2. QUBO → QAOA solver on Rigetti QVM
   - Using EntropicaQAOA
   - Requires some transformation from Qallse output format to Entropica recognized form

3. QUBO → QAOA solver on Rigetti QPU

Current Progress and Issues

Track

↓

QUBO

↓

QAOA Solver (QVM)

↓

QAOA Solver (QPU)

✓

✕

?

Qaoala: Possible Further Studies

- EntropicaQAOA has modules which can convert a QUBO to a MaxCut problem
  - Have not seen this before, only MaxCut to QUBO previously
  - Investigate if QUBO → MaxCut → QAOA solver is more efficient than current process
- Simulated QAOA on D-wave machine

Thank you!

Any Questions?

Quantum Computing: Some Current Prospects

- D-Wave
  - Current QPU (D-Wave 2000Q) has 2048 qubits and 6016 couplers
  - Operates by process of adiabatic quantum annealing
  - Provides a cloud platform (Ocean) for public use
- Rigetti
  - Circuit-based quantum computing
  - Provides a cloud platform (Forrest) for public use

BERKELEY LAB

QAOA

- QAOA prepares the state $|\gamma, \beta\rangle = V_p U_p \ldots V_1 U_1 |\psi\rangle$ where p is the number of iterations, $|\psi\rangle$ is the uniform superposition of all bit strings. The QAOA prepares this state then measures it in the computational basis.
- Each ith iteration, you apply a unitary cost function U_i (dependent on i and the cost Hamiltonian) for some angle γ_i followed by the driver unitary function V_i (dependent on i and the driver Hamiltonian) for some angle β_i . The driver Hamiltonian is the sum of all ith qubits σ_i

BERKELEY LAB

The Maximum Cut (MaxCut) Problem: Graph Theory Foundations

- In graph theory, a graph G = (V,E) has V vertices and E edges. A **cut** can be made somewhere on the graph to partition the vertices into two disjoint subsets.
- The **cut-set** is the set of edges which have endpoints in different subsets. The edges in this set connect the two subsets (Fig. 1). A cut-set C = (S,T) defines the two disjoint subsets S and T.

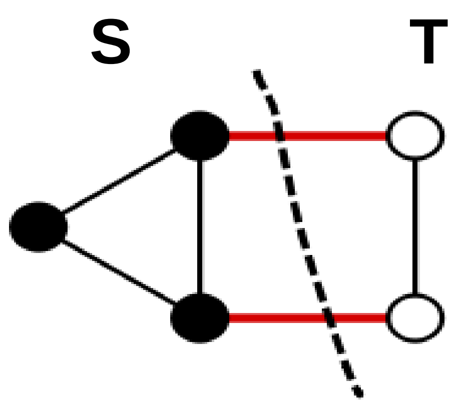

Figure 1. A graph has been cut into two disjoint subsets (filled, left and blank, right). The cut-set is depicted in red.

The Maximum Cut (MaxCut) Problem

- In the Maximum Cut problem (MaxCut), the goal is to find the cut for a graph which maximizes the number of edges in the cut-set.This is an NP hard problem.

Ising model → QAOA on QVM

- Apparent dependency issues
  - Qallse is used to formulate track problem to QUBO (later converted to Ising model), though certain QVM QAOA solvers fail to run on Ising model Hamiltonians which are originally sourced from Qallse
  - QAOA solvers work smoothly on unrelated Ising model problems from different sources
  - One likely solution is to somehow isolate Qallse from the other steps
    - (One approach proposed by Koji Terashi has Qallse output dumped into an external text file

BERKELEY LAB

Quantum Computing: In Theory

- With 2 classical bits, there exist 4 possible states: 00, 01, 10, 11. With 2 qubits, there exist 4 computational basis states: |00〉, |01〉, |10〉, |11〉 but the 2 qubits can exist in a superposition of these four states. The quantum state of this 2 qubit system includes a complex coefficient (amplitude) associated with each complex basis state.
- Because of this superposition, for a system of n qubits, the quantum state of the system has 2^n amplitudes.

Quantum Computing: In Theory

- Classical bits can take on a value of *0* or *1,* while quantum bits (qubits) take on a linear combination of states $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ where $|\alpha|^2$ and $|\beta|^2$ are the probabilities of measuring each respective state.

- $|0\rangle$ and $|1\rangle$ form an orthonormal basis for a vector space. A qubit's state is a unit vector in 2D complex vector space. [1]