Quantum Approximate Optimization Algorithm

Jamal Slim

DESY - FTX

jamal.slim@desy.de

Software weekly meetings
October 26, 2023

The Quantum Approximate Optimization Algorithm

- The Quantum Approximate Optimization Algorithm (QAOA): obtain approximate solutions of the problem of minimizing

$$C(x) = \sum_a w_a C_a(x)$$

    - where x is n-bit string, w_a are real weights and each C_a is a boolean function
    - E.g. Max-Cut problem, with every w_a equal to 1 and C_a takes the following form

$$x_i \oplus x_j$$

QAOA: Introduction

- For each boolean function C_a, a Hamiltonian H_a of the form can be constructed as

$$a_0 I + \sum_i a_i Z_i + \sum_{i<j} a_{ij} Z_i Z_j + \sum_{i<j<k} a_{ijk} Z_i Z_j Z_k$$

  - such that for every string x it holds that

$$C_a(x) = \langle x | H_a | x \rangle$$

- Then, minimizing $C(x)$ is equivalent to finding the ground state of

$$H_f = \sum_a w_a H_a$$

- given that H_f is diagonal and

$$C(x) = \langle x | H_f | x \rangle$$

QAOA: State preparation

- The time dependent Hamiltonian can also be written as

$$H(t) = (1 - \frac{t}{T})H_i + \frac{t}{T}H_f$$

  - with

$$H_i = -\sum_{j=1}^{n} X_j$$

- As an approximation of the evolution of the system, we consider parametrized states of the form state of

$$\prod(e^{i\Delta t(AH_i + BH_f)}) |\psi_0\rangle$$

Using Lie-Trotter formula

$$e^{i\Delta t(AH_i + BH_f)} \approx e^{i\Delta t(AH_i)} e^{i\Delta t(BH_f)}$$

QAOA: State preparation

- Using a quantum circuit to prepare a state of the form

$$|\beta, \gamma\rangle = e^{i\beta_p H_i} e^{i\gamma_p H_f} \dots e^{i\beta_1 H_i} e^{i\gamma_1 H_f} |\psi_0\rangle$$

  - for a fixed p, $\vec{\beta} = (\beta_1 \cdots \beta_p)$ and $\vec{\gamma} = (\gamma_1 \cdots \gamma_p)$ whole state $\langle\beta, \gamma\rangle$
- $|\psi_0\rangle = \sum_{i=0}^{2^n-1} |x\rangle$

QAOA: The Algorithm

- Choose a value for p and some initial angles β, γ
- Prepare the state $\langle\beta, \gamma\rangle$
- Estimate the energy $E(\beta, \gamma) = \langle\beta, \gamma|H_f|\beta, \gamma\rangle$ of $|\beta, \gamma\rangle$
- vary β and γ in order to minimize $E(\beta, \gamma)$
- If the stopping criterium is met, stop. Else, go to 2

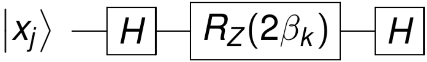
Step 2 is carried out on the quantum computer and steps 1, 3 and 4, on a classical one

Rotations

- $\psi_0 = \sum_{i=0}^{2^n-1} |x\rangle$ can be prepared with Hadamard gates
- Each $e^{-i\beta_k X_j}$ is a rotation$R_X(2\beta_k)$ or equivalently
- For $e^{-i\gamma H_f X_j}$
- Compute the parity of X_j with CNOT gates
- Apply $R_Z(2\gamma_k)$ on the qubit where we have computed the parity
- Uncompute the parity

Rotations

- $|\psi_0\rangle = \sum_{i=0}^{2^n-1} |\psi_0\rangle$ can be prepared using Hadamard gates
- Each $e^{-i\beta_k X_j}$ is a rotation$R_X(2\beta_k)$ or equivalently



- For $e^{-i\gamma H_f X_j}$ is a bit more complicated:
- Implementing $e^{-i\gamma H_f X_j}$ only requires $e^{-i\gamma Z_{i_1} \cdots Z_{i_j}}$
- $e^{-i\gamma Z_{i_1} \cdots Z_{i_j}}$ is diagonal in the computational basis
- $|x\rangle \mapsto e^{-i\gamma_k} |x\rangle$ if $x_{i_1} \oplus \cdots \oplus x_{i_j} = 0 \bmod 2$
- $|x\rangle \mapsto e^{i\gamma_k} |x\rangle$ if $x_{i_1} \oplus \cdots \oplus x_{i_j} = 1 \bmod 2$
- This is equivalent to the R_z rotation
- Compute the parity of X_j with CNOT gates
- Apply $R_Z(2\gamma_k)$ on the qubit where we have computed the parity
- Uncompute the parity