SAND2019-1238C

Quantum Approximate Optimization Algorithm (QAOA) on Constrained Optimization Problems

Jaimie S. Stephens[1, 2, 3], Ciarán Ryan-Anderson[4], William Bolden[1, 5], and Ojas Parekh[1]

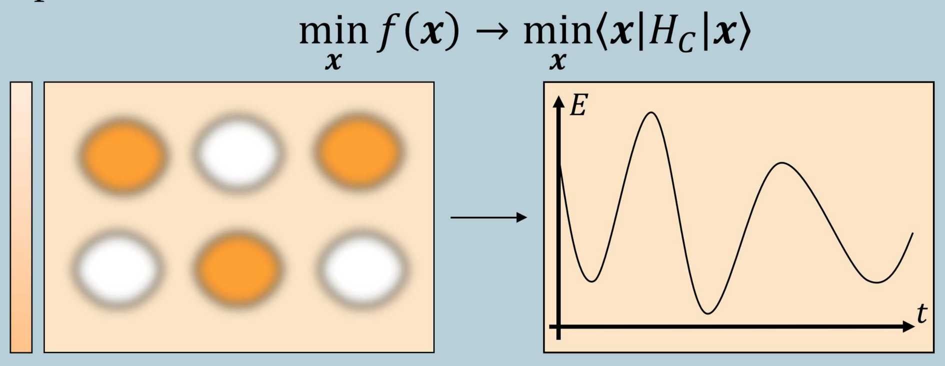
[1]*Advanced Device Technologies, Sandia National Laboratories, Albuquerque, NM, 87185, USA;*
[2]*Center for Quantum Information and Control, University of New Mexico, Albuquerque, NM, 87131, USA;*
[3]*Department of Physics and Astronomy, University of New Mexico, Albuquerque, NM, 87131, USA;*
[4]*Department of Physics, College of Science, Swansea University, Singleton Park, Swansea - SA2 8PP, United Kingdom;*
[5]*Department of Computer Science and Engineering, University of California Santa Cruz, Santa Cruz, CA 95064, USA*

QAOA, first proposed by Farhi *et. al.* in 2014, is a quantum algorithm that approximates hard optimization problems

Given $f:\{0,1\}^n \to \mathbb{R}$, find $x \in \{0,1\}^n$ s.t. $f(x)$ is a minimum (maximum).

Now let's change the problem.

$$\min_{x} f(\boldsymbol{x}) \to \min_{x} \langle \boldsymbol{x}|H_C|\boldsymbol{x}\rangle$$



Instead of minimizing $f(x)$, we find the ground state of a cost Hamiltonian, H_C, where $H_C|x\rangle = f(x)|x\rangle$.

QAOA was inspired by Trotterization of Adiabatic Quantum Computing.

Start in the equal super position of all states,

$$|s\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle .$$

- **In AQC**, we would evolve under $H(t) = s(t)H_C + (1 - s(t))H_D$ where $s = s(t)$ is a smooth function of t, $s(t=0) = 0$, and $s(t = \mathrm{T}) = 1$
- **In Trotterization of AQC**, we approximate

$$U(T,0) = \prod_{k=1}^{N} U(k\Delta t, (k-1)\Delta t)$$

where $\Delta t = {}^{T}/_{N}$ and,

$$U(t_2, t_1) = \mathcal{T}\exp\left[-i\int_{t_1}^{t_2} H(t)dt\right]$$

as

$$U(T,0) \approx \prod_{k=1}^{N} e^{-i\Delta t H(k\Delta t)}$$

$$\approx \prod_{k=0}^{N} \left(e^{-i\Delta t(1-s(k\Delta t))H_D}\right)\left(e^{-i\Delta t s(k\Delta t)H_C}\right).$$

- **In QAOA**, we evolve under

$$Q_m(\boldsymbol{\beta},\boldsymbol{\gamma}) = \prod_{k=0}^{m} \left(e^{-i\beta_k H_D}\right)\left(e^{-i\gamma_k H_C}\right).$$

We make m small, but $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ and $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_m)$ are free variables. From Trotterization, as $m \to \infty$ we get the exact solution.

Three ways of approximating Constrained Optimization Problems w QAOA variations

Method	H_C	H_D	Initial State
Original QAOA Farhi *et. al.* (2014)	$\sum_{x=0}^{2^n-1} f(x)\|x\rangle\langle x\|$	$-\sum_{j=0}^{2^n} X_j$	$\|+\rangle^{\otimes n}$
QAOA++ Hadfield *et. al.* (2017)	$\sum_{x=0}^{2^n-1} f(x)\|x\rangle\langle x\|$	$H_D\|x\rangle = \sum_j c_j\|x_j\rangle \in \mathcal{F}\ \forall\ \|x\rangle \in \mathcal{F}$	$\|\psi\rangle \in \mathcal{F}$
Constrained Quantum Annealing Hen and Spedalieri (2016)	$\sum_{x=0}^{2^n-1} f(x)\|x\rangle\langle x\|$	$[H_D, H_F] = 0$ where for the computational basis $\{\|x_j\rangle\}$, $H_F\|x_j\rangle = \begin{cases} 0 & \text{if } \|x_j\rangle \in \mathcal{F} \\ \lambda_j\|x_j\rangle \text{ s.t. } \lambda_j \gg 0 & \text{if } \|x_j\rangle \notin \mathcal{F} \end{cases}.$	$\|\psi\rangle \in \mathcal{F}$
QAOA with Penalties and Guaranteed Feasibility Stephens *et. al.*	$\sum_{x=0}^{2^n-1} f(x)\|x\rangle\langle x\| + H_P$	$-\sum_{j=0}^{2^n} X_j$	$\|+\rangle^{\otimes n}$

QAOA++ ⇔ CQA

Proof:

($\Longrightarrow$) Suppose $\forall\ |f\rangle \in \mathcal{F}, H_D|f\rangle = \sum_j c_j\,|f_j\rangle$. Define

$$H_F = \sum_{x=0}^{2^n-1} c_x|x\rangle\langle x| \text{ where } c_x = \begin{cases} 0 \; if \; |x\rangle \in \mathcal{F} \\ 1 \quad\quad o.w. \end{cases}.$$

If $\langle x|H_D H_F|y\rangle = \langle x|H_F H_D|y\rangle\ \forall$ basis vectors $|x\rangle$ and $|y\rangle$ of the full space then $[H_D, H_F] = 0$. There are three cases:

(Case 1) $|x\rangle \in \mathcal{F}$ and $|y\rangle \in \mathcal{F}$. $H_F|x\rangle = H_F|y\rangle = 0 \Rightarrow \langle x|H_D H_F|y\rangle = \langle x|H_F H_D|y\rangle = 0.$

(Case 2) Either $|x\rangle$ or $|y\rangle \in \mathcal{F}$, but not both. WLOG let $|x\rangle \in \mathcal{F}$. Then $H_D|x\rangle \in \mathcal{F} \Longrightarrow H_F|x\rangle = H_F(H_D|x\rangle) = 0$. Thus $\langle x|H_D H_F|y\rangle = \langle x|H_F H_D|y\rangle = 0$.

(Case 3) $|x\rangle \notin \mathcal{F}$ and $|y\rangle \notin \mathcal{F}$. $H_F|x\rangle = |x\rangle$ and $H_F|y\rangle = |y\rangle \Longrightarrow \langle x|H_D H_F|y\rangle = \langle x|H_D|y\rangle = \langle x|H_F H_D|y\rangle$.

($\Longleftarrow$) Suppose $[H_D, H_F] = 0$, and let the feasible subspace be the ground state of the feasible Hamiltonian $H_F|x\rangle = 0$ iff $|x\rangle \in \mathcal{F}$. Let $|x\rangle \in \mathcal{F}$. Then $H_F(H_D|x\rangle) = H_D H_F|x\rangle = 0$. Thus $H_D|x\rangle \in \mathcal{F}\ \forall\ |x\rangle \in \mathcal{F}$.

Therefore if you have a driving Hamiltonian for QAOA++ you can use it for CQA and if you have a driving Hamiltonian for CQA you can use it for QAOA++. ∎

Finding Penalties that Guarantee Feasibility

We choose the penalty Hamiltonian,

$$H_P = \sum_{x=0}^{2^n-1} p_x|x\rangle\langle x| ,$$

to be obtained from a classical oracle. The penalties, p_x, are found by,

$$p_x = \langle x|P^\dagger H_C P|x\rangle - \langle x|H_C|x\rangle$$

where, P, is a classical algorithm that maps an infeasible state to a feasible one

$$P|x\rangle = \begin{cases} |x\rangle & \text{if } |x\rangle \in \mathcal{F} \\ |y\rangle \in \mathcal{F} & \text{o.w.} \end{cases}.$$

This method can be used for any constrained optimization problem to produce a feasible output.

- **MinVertexCover Example** (unweighted case)

Given a graph, $G = (V, E)$, the cost Hamiltonian is

$$H_C = \sum_{\mu \in V} W_\mu$$

where $W_\mu = |1\rangle\langle 1|_\mu$. Note, the expectation of H_C for some basis state, $|x\rangle$, is the number of vertices in the subset $V' \subseteq V$ that is represented by $|x\rangle$. Then we define

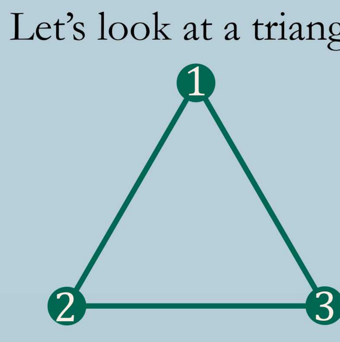
$$P_{minVC} = \sum_{(\mu,\nu)\in E} \left(I - W_\mu - \mathrm{W}_\nu + W_\mu W_\nu\right).$$

This is the operator equivalent of

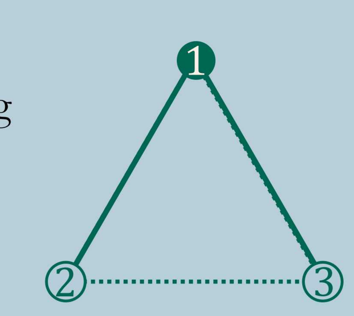
$$\sum_{(\mu,\nu)\in E} (1 - x_\mu - x_\nu + x_\mu x_\nu),$$

for x_μ. It is easy to see that this is a classical oracle for counting the number of edges not covered.

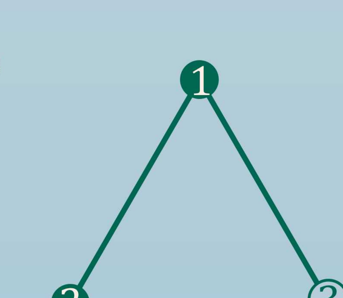
Let's look at a triangle:



Then consider getting the following state after running QAOA, $|100\rangle$. The cost of this solution is 1and the penalty of not covering $(2, 3)$ is 1.



Now, we choose the solution that costs the least. The cost of adding vertex 2 is equal to the penalty of not covering $(2, 3)$, so we choose the feasible solution. This classical correction will take $O(|E|)$.



Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.
SAND No. _____________

CQuIC Center for Quantum Information and Control

The University of New Mexico

Sandia National Laboratories