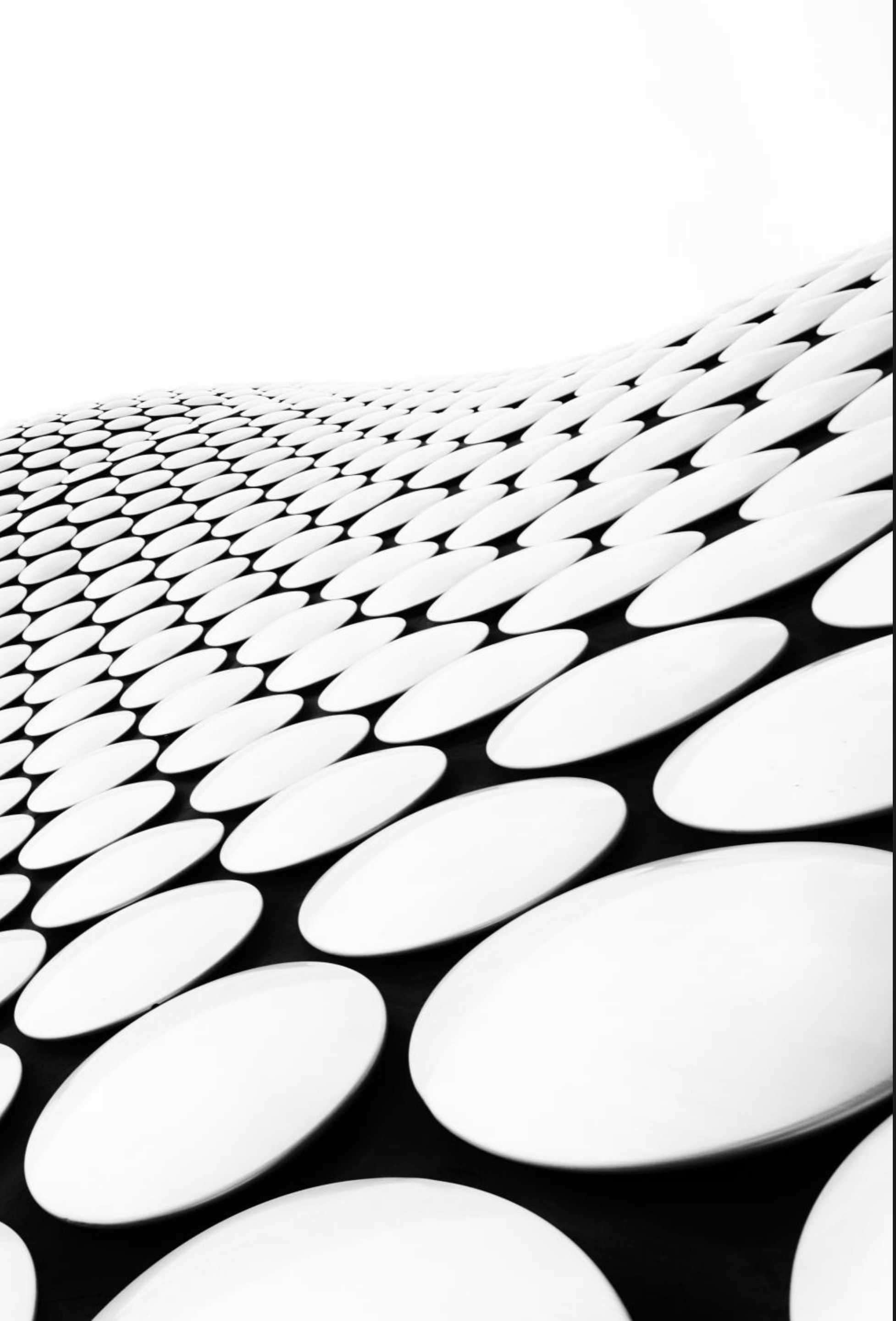

CLOUD COMPUTING FOR QUANTUM COMPUTER,

QUANTUM COMPUTER FOR CLOUD COMPUTING

QHack 2023 Open Hackathon

Amazon Bracket Challenge

Team Qloud

Table of Content

Introduction

Quantum computer and Cloud computing

- Cloud based quantum computing
  - Quantum processor through cloud
  - Many industry starts to develop system
    - Ex) Amazon Braket, Xanadu Quantum Cloud
  - Cloud computing helps quantum computing

- Is there a method for quantum computer to help cloud computing?
  - Maybe solving Important problem for cloud service
    - Which is hard to solve in classical computer

Introduction

Importance of Bin Packing Problem (BPP)

- Cloud service with the multi-server
  - Important to make operation costs down
  - Have to use energy efficiently
- Solving the Bin packing problem[1]
  - jobs can be efficiently allocated to machines for cloud service
  - Bin packing problem is an Np-hard problem so it's difficult to solve classically
- Goal of this project
  - Solve bin packing problems using several quantum optimization algorithms for cloud computing
  - Compare results form algorithms

Introduction

Project Plan

- Solve BPP Problem
  - In classical way
    - Using Gourbi Optimizer
  - Quantum way
    - Quantum Approximate Optimization Algorithm
    - Quantum Alternating Operator Ansatz
    - Feedback-based Quantum Optimization
  - Compare Result

- Expected result
  - Classical algorithm will solve much faster because problem is very small
  - FALQON maybe efficient because no need of extra optimizer

Method

Problem Discerption BPP

- Bin Packing Problem (BPP)
  - Optimization problem
  - Pack (n) items with different size (w) to finite number (u) of bins size C

- Mathematical formulation of BPP [2]

$$\min \sum_{i=1}^{\mathrm{u}} y_i$$

$$subjet\ to \sum_{j=1}^{n} w_j x_{ij} < c y_i \ \ (i = 1 \dots u)$$

$$\sum_{i=1}^{u} x_{ij} = 1 \ (j = 1 \dots n)$$

$$x_{ij}, y_i \in \{0,1\}$$

Method

Classical algorithm

- BPP is Binary Linear Programming (NP-Hard)
- Change Problem to Quadratic Unconstraint Binary Optimization
  - Change inequality constraint to equality constraint using extra variable
  - Add constraint to cost function with penalty term
- Using Gurobi optimizer [3]
  - Cutting plane
  - Branch and Bound
  - Branch and cut
- Performance
  - Fast solve for small size problem
  - Hard to solve in big size

Method

Quantum Approximate Optimization Algorithm (QAOA)

- Quantum Algorithm for combinatorial optimization
  - Find state minimize $\langle\psi(\beta,\gamma)|H_C|\psi(\beta,\gamma)\rangle$
  - Make state in quantum circuit and optimize using classical algorithm
- Method (Ansatz) [4]
  - $|\psi(\beta,\gamma)\rangle = U_B(\beta_p)U_C(\gamma_p)\dots U_B(\beta_1)U_C(\gamma_1)|s\rangle$
    - $U_B(\beta_k) = e^{-i\beta_k H_B \Delta t}, U_C(\beta_p) = e^{-i\gamma H_C}$
    - H_B : Mixer Hamiltonian
    - H_C : Problem Hamiltonian express cost function
- Implementation to problem
  - Using problem as QUBO formulation
  - Change binary variable (0,1) to 1, -1 variable
  - Optimize parameter using classical ADAM optimizer

Method

Quantum Alternating Operator Ansatz

- More general class of quantum state
  - Can dealing with optimization problem with constraints
- Method (Ansatz) [5]
  - $|\psi(\beta,\gamma)\rangle = U_B(\beta_p)U_C(\gamma_p)\ldots U_B(\beta_1)U_C(\gamma_1)|s\rangle$
    - Almost same formula with QAOA
  - Mixer makes state only in feasible subspace
  - Initial state to single feasible solution (satisfy constraint)
- Impetration to problem
  - constraint ->mixer operator (Don’t have to convert in QUBO)
    - Ex) for constraint $x_a + x_b = 1$
      - $|1\rangle\langle 0|_a \otimes |0\rangle\langle 1|_b + |0\rangle\langle 1|_a \otimes |1\rangle\langle 0|_b$
      - Because one of x_a or x_b have to be 1
  - Train Variable and see

Method

Feedback-based Quantum Optimization (FALQON)

- Quantum Optimization algorithm using Quantum way optimization method
  - Don't need extra classical optimizer

- Method [6]
  - $|\psi(\beta)\rangle_k = U_B(\beta_k)U_C \ldots U_B(\beta_1)U_C|+\rangle$
  - $U_B(\beta_k) = e^{-i\beta_k H_B \Delta t}, U_C = e^{-iH_C \Delta t}$
  - Recursively calculating β_1 to β_l
    - $\beta_{k+1} = -\langle\psi_k|i[H_B, H_C]|\psi_k\rangle$

- Implementation to problem
  - Convert problem to QUBO formulation
  - Repeat iteration to find optimal solution

Simulation

Toy model

- Problem
  - 2 bin with size 3 each
  - Work a (size 1) + work b (size 2)
- Index formulation
  - 10 bit to express whole problem
  - 0,1 bit -> if each bin(1^{st} and 2^{nd}) are used
  - 2,3 bit -> if each work (1^{st} and 2^{nd}) are in 1^{st} bin
  - 4,5 bit -> if each work (1^{st} and 2^{nd}) are in 2^{nd} bin
  - 6,7,8,9 -> extra variable for inequality constraint
- Optimal solution
  - Work a and b into first or second bit (using 1 bin)
    - [1011000000] : All work in 1st bin
    - [0100110000] : All work in 2^{nd} bin

Simulation

Classical Result

- Simulation method
  - Use python for problem generation
  - Calculate using personal computer
  - Use Gurobi (gurobipy) for classical optimizer

- Gurobi result
  - [1,0,1,1,0,0,0,0,0] (optimal solution)
  - 0.01621 second
  - Get optimal solution directly in very small amount of time

Simulation

How to Compare Quantum Algorithms

- If using bigger depth or iteration, performance of optimization will also increase
  - QAOA : time = depth X Iteration
    - Assuming no time needed for classical optimization
  - Falcon : time = depth(depth+1)/2
    - Assuming time for unitary is same for all algorithm

- Hard to compare
  - Different optimization method
    - Classical for QAOA vs Quantum for FALQON
    - Same # of variable is X
- Select Parameter
  - QAOA : depth 10, Iteration 500 -> 5000
  - FALCON : depth 100 -> 5000

Simulation

Quantum Result

- Simulation Method
  - Use python for problem generation
  - Pennylane for circuit implementation
  - Use personal computer to train (same as classical)
    - To compare time of algorithms
  - Verify solution using Amazon Web Services – SV1
    - Not using real hardware because comparing performance of algorithm is goal of this project

- QAOA result
  - [0100110000] (Optimal1 and max prob): 0.0172
  - [1011000000] (Optimal2): 0.0159
  - Find optimal state with high probability
  - Time : 467 sec

Simulation

Quantum Result

- QAO Ansatz Result
  - [0110010101] (Max prob): 0.02647
  - [0100110000] (Optimal1): 0.01145
  - [1011000000] (Optimal2): 0.00776
  - Fail to find solution
  - Time for training : 430 sec

- FALQON Result
  - [000000000] (Max prob): 0.0148
  - [0100110000] (Optimal1): 0.01375
  - [1011000000] (Optimal2): 0.01375
  - Optimal1 + Optimal2 > Max
  - Get optimal state for high probability
  - Time : 128.5 sec

Conclusion

Discussion

- Comparison of Quantum Algorithm
  - Time need in classical simulator
    - QAOA > QAO Ansatz > FALCON
  - Performance
    - QAOA (find solution) > FALQON > QAO Ansatz
  - For small size Bpp problem, using QAOA is better than others
- Comparison of Classical and quantum
  - Classical algorithm find solution much faster
  - Classical algorithm is accurate (can find optimal solution)
- Reason of low performance in Quantum algorithm
  - Stop iteration to fast because of fair comparison
  - Need to tune Hyperparameter

Conclusion

Future work

- Try solving bigger problem
  - For toy problem, classical algorithm was better
  - Compare performance and find scalability
    - Maybe beneficial for quantum
- Try different quantum algorithms
  - Other Extensions of QAOAs
  - Durr – Hoyer algorithm (Quadratic speed up)
- Try to solve more realistic problem
  - For cloud computing, Bin Packing with Chance Constraints(BPCC) is more realistic[9]
    - Solve harder version of BPP using quantum

Reference

1. Cloud computing : Nursen Aydin et al, “Multi-objective temporal bin packing problem: An application in cloud computing
2. Bin-packing problem : Martello, “Bin-packing Problem”
3. Gurobi algorithm : Advanced Gurobi Algorithms
4. QAOA : Edward Farhi et al, “A Quantum Approximate Optimization Algorithm”
5. QAOA : Sturat Hadfield et al, “From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz”,
6. FALQON : Alicia B. Magann et al, “Feedback-based quantum optimization”
7. QAOA code : Intro to QAOA — PennyLane documentation
8. FALQON code : Feedback-Based Quantum Optimization (FALQON) — PennyLane documentation
9. Future Problem : Maxime C. Cohern et al, “Overcommitment in Cloud Services: Bin Packing with Chance Constraints”