Quantum Optimization for Scheduling

Project on class [MATH303] Quantum Computation

Thomas Lagkalis

Electrical and Computer Engineering
Technical University of Crete

February, 2024

Table of Contents

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Definition of the scheduling problem

A generic and very abstract definition for the scheduling problem is the following:

definition

the challenge of efficiently allocating and arranging resources over time to optimize specific objectives.

In this project we will describe two specific instances of the scheduling problem:

- Job Shop Scheduling Problem (JSSP)
- Nurse Scheduling Problem (NSP)

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Algorithms/Approaches

There are many approaches/algorithms to solve scheduling problems. Two of them will be discussed in this project:

- Quantum Approximation Optimization Algorithm (QAOA) applied in JSSP.
- Quantum Annealing (QA) applied in NSP.

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Quantum Optimization Approximation Algorithm (QAOA)

Mathematical optimization deals with finding the best solution to a problem (according to some criteria) from a set of possible solutions.

QAOA is a hybrid algorithm that combines classical and quantum computing resources to find approximate solutions to combinatorial optimization problems.

QAOA relies on the use of two unitary operators dependent on angles. In each iteration we apply the the operators on the qubits, measure the state, estimate the objective function and find a set of angles which optimizes the objective function.

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Algorithm description

Pseudo-code of the quantum subroutine of the algorithm:

Input

- Number of rounds of optimization p
- Two size p array of angles, γ and β.
- Hamiltonians C_a corresponding to the clauses of the optimization problem.

Output

- An approximation to the solution of problem

Procedure

Step 1. Construct the n-qubit uniform superposition state.
for $i = 1$ to p **do**

Step 2a. Apply $U(C, \gamma[i])$

Step 2b. Apply $U(B, \beta[i])$

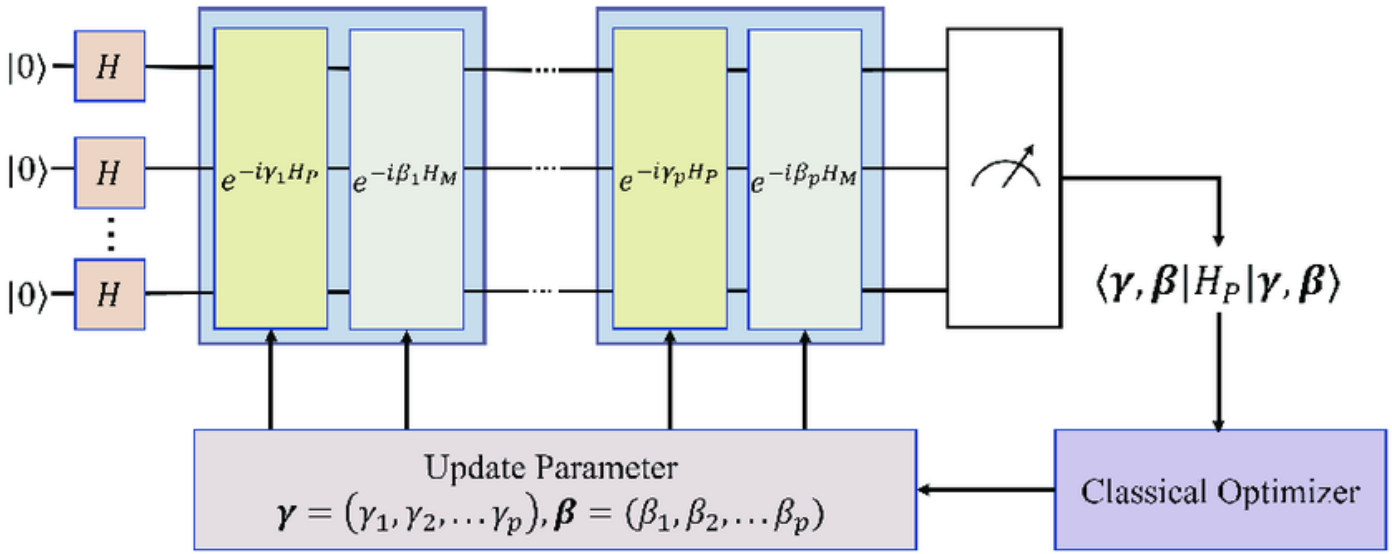
end for

Step 3. We call the state so constructed $|\beta, \gamma\rangle$. **return** $|\beta, \gamma\rangle$

Note: This is a demonstration of the quantum subroutine of QAOA. After Step 3 a classical optimization algorithm is used (e.g gradient descend) on variational parameters α and β to maximize $F_p(\gamma, \beta) = \langle \gamma, \beta | C | \gamma, \beta \rangle$

Quantum Circuit of QAOA

This algorithm could be described with the following circuit:



Job Shop Scheduling Problem (JSSP)

Abstract definition

The problem of allocating machines to a set of jobs over a specified time period.

Formal definition

We are given J jobs $(j_1, j_2, ..., j_J)$ of varying processing times, which need to be scheduled on M machines $(m_1, m_2, ..., m_M)$ with varying processing power, while trying to minimize the makespan – the total length of the schedule (that is, when all the jobs have finished processing). Each job consists of a set of O_j operations (tasks) $(O_{j1} \rightarrow O_{j2} \rightarrow ... \rightarrow O_{jo_j})$ which need to be processed in a specific order on a specific machine.

The binary variable of the problem:

$$x_{k,t} = \begin{cases} 1 & \text{if operation } o_k \text{ starts at time } t \leq T. \\ 0 & \text{otherwise} \end{cases}$$

Job Shop Scheduling Problem (JSSP)

There exist several constraints imposed on jobs and machines:

- Tasks are non-preemptive. Meaning running tasks cannot be interrupted by other tasks with the intention of resuming it at a later time.
- Tasks of different jobs are independent.
- Each task can be performed on one machine at a time.
- Each machine can process only one job at a time.

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Formulation of constraints

- Non preemption:

$$C_1(x) = \sum_k (\sum_t x_{k,t} - 1) = 0$$

- There can be only one operation running at a given machine at any time.

$$C_2(x) = \sum_m (\sum_{k,t,k',t' \in R_m} x_{k,t} x_{k',t'}) = 0$$

- the original order of the operations is kept for all the operations for every job:

$$C_3(x) = \sum_{n=1}^{J} (\sum_{\substack{k_{n-1} < k < k_n \\ t + l_k > t'}} x_{k,t} x_{k+1,t'}) = 0$$

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Formulation of constraints

All the above constraints are hard, i.e. they define a feasible solution. There can be one more soft constraint in order to minimize the total makespan:

- Penalty constraint:

$$C_4(x) = \sum_{n=1}^{J} (J+1)^{t_{k_n}}, \quad t_{k_n} \leq \tau$$

  where t_{k_n} is the completion time of last operation of job n and *tau* the completion time of an optimal schedule.

Now, by summing all the constraints we get the problem objection function:

$$C(\mathbf{x}) = \sum_{a=1}^{4} C_a(x)$$

Formulation of cost Hamiltonians

In order to get the cost Hamiltonian we have to:

1. Replace bits $x_i \in \{0, 1\}$ with spin variables $z_i \in \{-1, 1\}$ using the linear transformation $x_i = (z_i + 1)/2$
2. In $C(z) = \sum_m C_a(z_i)$ replace constants 1s with identity operator $I^{\otimes n}$ and z_i with Pauli Z-operator
   $Z_i = I^1 \otimes ... \otimes I^{i-1} \otimes \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes ... \otimes I^n$ acting on qubit i.

So from the objective funciton we get the cost Hamiltonian:

$$C(\mathbf{Z}) = \sum_{a=1}^{4} C_a(\mathbf{Z})$$

The mixing Hamiltonian is easy to prepare:

$$B = \sum_{j}^{n} X_j$$

Unitary operators

Now we have the cost and the mixing Hamiltonian we can define the two unitary operators needed for the problem:

- $U(C, \gamma)$ depends on an angle $\gamma \in [0, 2\pi)$

$$U(C, \gamma) = e^{-i\gamma C} = \prod_{a=1}^{m} e^{-i\gamma C_a} \tag{1}$$

- $U(B, \beta)$ depends on an angle $\beta \in [0, \pi)$

$$U(B, \beta) = e^{-i\beta B} = \prod_{j=1}^{n} e^{-i\gamma X_j} \tag{2}$$

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Quantum Annealing (QA)

Definition (Wikipedia)

Quantum annealing (QA) is an optimization process for finding the global minimum of a given objective function over a given set of candidate solutions (candidate states), by a process using quantum fluctuations.

Quantum annealing characteristics:

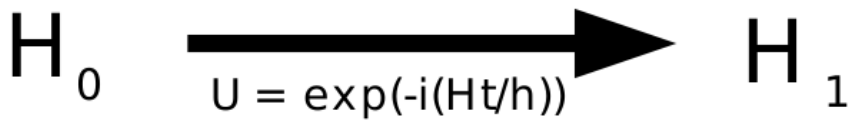
- Used mainly for problems where the search space is discrete (combinatorial optimization problems) with many local minima.
- QA is used in D-Wave systems.
- Leverages adiabatic evolution to find the ground state of cost Hamiltonian (that corresponds to the objective function).

Adiabatic Evolution

Quantum annealing uses adiabatic evolution to find the ground state of the problem (cost) Hamiltonian. The main idea is that we start from H_0, , whose ground state can be prepared easily and we evolve the Hamiltonian **slowly enough** using the Unitary operator $U = exp(-i\frac{Ht}{h})$ to reach the ground state of the problem Hamiltonian H_1.

$$H(t) = A(t)H_0 + B(t)H_1, \qquad t \in [0, T]$$

Where A(t) and B(t) are two monotonic functions which satisfy, $A(0) = 1$, $B(0) = 0$ and $A(T) = 0$, $B(T) = 1$.



ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Nurse Scheduling Problem (NSP)

Definition

Create a rotating roster of nurses in a hospital respecting a set of constraints.

We consider N nurses n = 1,...,N and a schedule consisting of D days d = 1,...,D.A Binary variable $x_{n,d} \in \{0, 1\}$ where $x_{n,d} = 1$ specifies the assignment of nurse n to day d.

Before we apply adiabatic evolution to solve the problem we have to map it to a Quadratic Unconstrained Binary Optimization (**QUBO**) problem.

$$f(\mathbf{x}) = \mathbf{x}^T Q \mathbf{x} = \sum_{i,j} Q_{i,j} x_i x_j$$

Where, **x** is a binary vector of length n ¿ 0, $x \in \{0, 1\}$ and a matrix $Q \in R^{N \times N}$ whose entries Q_{ij} define a weight for each pair of indices.

NSP constraints

We consider the following constraints:

- Hard shift constraint: at least 1 nurse is assigned each working day.
- Hard nurse constraint: no nurse works two or more consecutive days.
- Soft nurse constraint: all nurses should have approximately even work schedules.

All the above constraints can be summarized in the following QUBO formulation:

$$f(\mathbf{x}) = \underbrace{\sum_{i,j} J_{i,j} x_i x_j}_{\text{Hard nurse constraint}} + \underbrace{\lambda \sum_{d} \left(\sum_{n} E(n) x_i - W(d) \right)^2}_{\text{Hard shift constraint}}$$

$$+ \underbrace{\gamma \sum_{n} \left(\sum_{d} G(n,d) x_i - F(n) \right)^2}_{\text{Soft nurse constraint}}$$

NSP constraints

Where:

W(d): the required workforce W(d) needed on each day d

E(n): the amount of effort each nurse has available.

i(n, d) and j(n, d): composite indices as functions of the nurse n and the day d.

J: matrix (symmetric) such that $J_{i(n,d),j(n,d+1)} = a$ and 0 otherwise.

F(n): the number of work days that each nurse wishes to be scheduled.

G(n, d)

λ, γ: weight factors which denote the significance of each term

NSP in D-Wave's hybrid quantum-classical computer

Implemented the Nurse Scheduling Problem using D-Wave's built in libraries which simplify the process of formulating and accessing quantum (hybrid) computers. Details about the implementation and the code can be found in the project report (see final slide for link).

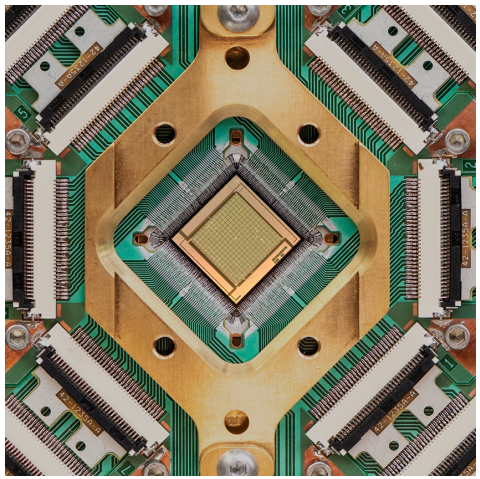


Figure: D-Wave's QPU

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Experimental results

Figure: Three experiments on D-Wave's QPU with different input sizes.

```
Problem Settings:
Number of nurses: 3
Number of days: 6

Sending problem to hybrid sampler...

Energy of solution: -18.0
Number of occurrences of solution: 1
Time of execution: 9.156771421432495 seconds

Optimal schedule calculated:

n/d| 1 | 2 | 3 | 4 | 5 | 6 |
 1 |   | X |   | X |   | X |
 2 | X |   |   |   | X |   |
 3 |   |   | X |   |   |   |
```

(a) 3 nurses 6 days

```
Problem Settings:
Number of nurses: 5
Number of days: 8

Sending problem to hybrid sampler...

Energy of solution: -8.0
Number of occurrences of solution: 1
Time of execution: 9.386857986450195 seconds

Optimal schedule calculated:

n/d| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
 1 |   |   | X |   |   |   | X |   |
 2 | X |   |   |   |   |   |   |   |
 3 |   |   |   | X |   | X |   |   |
 4 |   | X |   |   | X |   |   |   |
 5 |   |   |   |   |   |   |   |   |
```

(b) 5 nurses 8 days

```
Problem Settings:
Number of nurses: 10
Number of days: 14

Sending problem to hybrid sampler...

Energy of solution: -14.0
Number of occurrences of solution: 1
Time of execution: 9.875545978546143 seconds

Optimal schedule calculated:

n/d| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
 1 |   |   |   |   |   | X |   |   |   |    |    | X |    |    |
 2 |   |   |   |   | X |   |   |   |   |    |    |   |    |    |
 3 |   |   |   |   |   |   |   |   |   |    |    |   |    |    |
 4 |   |   |   |   |   |   |   |   | X |    |    |   |    |    |
 5 |   |   |   | X |   |   |   | X |   |    |    |   |    |    |
 6 |   |   |   |   |   |   |   |   |   | X  |    | X |    |    |
 7 |   | X |   |   |   |   |   |   |   |    |    |   |    |    |
 8 |   |   |   |   |   |   | X |   |   |    |    |   |    |    |
 9 | X |   |   |   |   |   |   |   |   |    |    |   |    |    |
10 |   |   | X |   |   |   |   |   |   |    |    |   |    |    |
```

(c) 10 nurses 14 days

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Thank you!

Code and report can be found at:
github.com/ThomasLagkalis/TUC-ARCHIVE → Quantum_Qomputing_MATH303

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ