Five Starter Problems: Solving Quadratic Unconstrained Binary Optimization Models on Quantum Computers

https://github.com/arulrhikm/Solving-QUBOs-on-Quantum-Computers

Arul Rhik Mazumder[1] Sridhar Tayur[2]

[1]School of Computer Science, Carnegie Mellon University
[2]Tepper School of Business, Carnegie Mellon University

October 1, 2025

Overview: Table of Contents[1]

[1] The animations on slides 19, 28, 33 were created using the `animate` package. It is only visible in PDF viewers that support animated PDF features, such as Adobe Acrobat Reader.

Section 1

Quadratic Unconstrained Binary Optimization

The QUBO Model

The Universal Language for Optimization

QUBO Model: Definition and Universality

- **Definition:** Quadratic Unconstrained Binary Optimization (QUBO) models problems in operations research, finance, and physics.

QUBO Model: Definition and Universality

- **Definition:** Quadratic Unconstrained Binary Optimization (QUBO) models problems in operations research, finance, and physics.
- **Mathematical Form:** Minimize a quadratic function of binary variables $\mathbf{x} \in \{0,1\}^n$:

$$\min_{\mathbf{x}\in\{0,1\}^n} [\mathbf{x}^T \mathbf{Q} \mathbf{x} + c]$$

  - $\mathbf{Q}$ is the QUBO matrix.
  - The constant c is irrelevant to the optimal solution.

QUBO Model: Definition and Universality

- **Definition:** Quadratic Unconstrained Binary Optimization (QUBO) models problems in operations research, finance, and physics.
- **Mathematical Form:** Minimize a quadratic function of binary variables $\mathbf{x} \in \{0,1\}^n$:

$$\min_{\mathbf{x}\in\{0,1\}^n} [\mathbf{x}^T \mathbf{Q} \mathbf{x} + c]$$

  - $\mathbf{Q}$ is the QUBO matrix.
  - The constant c is irrelevant to the optimal solution.
- **Expanded/Triangular Form:** Since $x_i^2 = x_i$ for binary variables:

$$\min_{x_i\in\{0,1\}} \left[\sum_{i<j} Q_{ij} x_i x_j + \sum_i Q_{ii} x_i + c \right]$$

QUBO Model: Definition and Universality

- **Definition:** Quadratic Unconstrained Binary Optimization (QUBO) models problems in operations research, finance, and physics.
- **Mathematical Form:** Minimize a quadratic function of binary variables $\mathbf{x} \in \{0,1\}^n$:

$$\min_{\mathbf{x} \in \{0,1\}^n} [\mathbf{x}^T \mathbf{Q} \mathbf{x} + c]$$

  - $\mathbf{Q}$ is the QUBO matrix.
  - The constant c is irrelevant to the optimal solution.
- **Expanded/Triangular Form:** Since $x_i^2 = x_i$ for binary variables:

$$\min_{x_i \in \{0,1\}} \left[\sum_{i<j} Q_{ij} x_i x_j + \sum_i Q_{ii} x_i + c \right]$$

- **Universality:** QUBO provides a unified framework for representing combinatorial optimization, including many NP-hard problems.

Classical Approaches for QUBO

- **Exact Algorithms:**

Classical Approaches for QUBO

- **Exact Algorithms:**
  - Methods like branch-and-bound and semidefinite optimization are used, but their runtime is limited by the NP-Hard nature of the problems.

Classical Approaches for QUBO

- **Exact Algorithms:**
  - Methods like branch-and-bound and semidefinite optimization are used, but their runtime is limited by the NP-Hard nature of the problems.
- **Heuristic and Metaheuristic Algorithms:**

Classical Approaches for QUBO

- **Exact Algorithms:**
  - Methods like branch-and-bound and semidefinite optimization are used, but their runtime is limited by the NP-Hard nature of the problems.
- **Heuristic and Metaheuristic Algorithms:**
  - These general-purpose techniques are often applied to find high-quality, near-optimal solutions quickly.

Classical Approaches for QUBO

- **Exact Algorithms:**
  - Methods like branch-and-bound and semidefinite optimization are used, but their runtime is limited by the NP-Hard nature of the problems.
- **Heuristic and Metaheuristic Algorithms:**
  - These general-purpose techniques are often applied to find high-quality, near-optimal solutions quickly.
  - **Key Examples:**
    1. **Simulated Annealing (SA)**: A metaheuristic that uses a "temperature" to explore the solution space and escape local minima.
    2. Genetic Algorithms.
    3. Tabu Search.

Classical Approaches for QUBO

- **Exact Algorithms:**
  - Methods like branch-and-bound and semidefinite optimization are used, but their runtime is limited by the NP-Hard nature of the problems.
- **Heuristic and Metaheuristic Algorithms:**
  - These general-purpose techniques are often applied to find high-quality, near-optimal solutions quickly.
  - **Key Examples:**
    1. **Simulated Annealing (SA)**: A metaheuristic that uses a "temperature" to explore the solution space and escape local minima.
    2. Genetic Algorithms.
    3. Tabu Search.
  - These methods offer competitive performance against specialized algorithms in practice.

The Quantum Landscape for QUBO

- **Quantum Relevance:** QUBOs are mathematically equivalent to the Ising Model, making them central to quantum optimization.

The Quantum Landscape for QUBO

- **Quantum Relevance:** QUBOs are mathematically equivalent to the Ising Model, making them central to quantum optimization.
- **1. Quantum Annealing (QA):**
  - **Method:** Finds the global minimum by utilizing quantum fluctuations, particularly suited for dedicated hardware (e.g., D-Wave).

The Quantum Landscape for QUBO

- **Quantum Relevance:** QUBOs are mathematically equivalent to the Ising Model, making them central to quantum optimization.
- **1. Quantum Annealing (QA):**
  - **Method:** Finds the global minimum by utilizing quantum fluctuations, particularly suited for dedicated hardware (e.g., D-Wave).
- **2. Gate-Based Quantum Computing:**
  - **Key Algorithm:** Quantum Approximate Optimization Algorithm (QAOA), a hybrid classical-quantum approach using quantum gates to find approximate solutions.

The Quantum Landscape for QUBO

- **Quantum Relevance:** QUBOs are mathematically equivalent to the Ising Model, making them central to quantum optimization.
- **1. Quantum Annealing (QA):**
  - **Method:** Finds the global minimum by utilizing quantum fluctuations, particularly suited for dedicated hardware (e.g., D-Wave).
- **2. Gate-Based Quantum Computing:**
  - **Key Algorithm:** Quantum Approximate Optimization Algorithm (QAOA), a hybrid classical-quantum approach using quantum gates to find approximate solutions.
- **3. Hybrid/Variational Quantum-Classical Heuristics:**
  - Methods that combine quantum subroutines with classical optimization, such as Quantum-Assisted Genetic Algorithms (QAGA).

The Quantum Landscape for QUBO

- **Quantum Relevance:** QUBOs are mathematically equivalent to the Ising Model, making them central to quantum optimization.
- **1. Quantum Annealing (QA):**
  - **Method:** Finds the global minimum by utilizing quantum fluctuations, particularly suited for dedicated hardware (e.g., D-Wave).
- **2. Gate-Based Quantum Computing:**
  - **Key Algorithm:** Quantum Approximate Optimization Algorithm (QAOA), a hybrid classical-quantum approach using quantum gates to find approximate solutions.
- **3. Hybrid/Variational Quantum-Classical Heuristics:**
  - Methods that combine quantum subroutines with classical optimization, such as Quantum-Assisted Genetic Algorithms (QAGA).
- **4. Quantum-Inspired Algorithms:**
  - Classical algorithms (like Quantum Particle Swarm Optimization) that incorporate principles from quantum mechanics to enhance performance without using quantum hardware.

Section 2

Canonical QUBO Formulation

The Number Partitioning Problem

Balancing the Binary Partition

Canonical Problem: Number Partitioning (NP)

- **Problem Definition (NP-Hard):** Given a set S of positive integers $\{s_1, s_2, \ldots, s_n\}$, partition S into two subsets, A and $S \setminus A$.

Canonical Problem: Number Partitioning (NP)

- **Problem Definition (NP-Hard):** Given a set S of positive integers $\{s_1, s_2, \ldots, s_n\}$, partition S into two subsets, A and $S \setminus A$.
- **Objective:** Minimize the absolute difference (d) between the sum of elements in A and the sum of elements in $S \setminus A$.

$$d = \left| \sum_{s_i \in A} s_i - \sum_{s_j \in S \setminus A} s_j \right|$$

Canonical Problem: Number Partitioning (NP)

- **Problem Definition (NP-Hard):** Given a set S of positive integers $\{s_1, s_2, \ldots, s_n\}$, partition S into two subsets, A and $S \setminus A$.
- **Objective:** Minimize the absolute difference (d) between the sum of elements in A and the sum of elements in $S \setminus A$.

$$d = \left| \sum_{s_i \in A} s_i - \sum_{s_j \in S \setminus A} s_j \right|$$

- **Goal:** Make the sums of the two subsets as close as possible.

Canonical Problem: Number Partitioning (NP)

- **Problem Definition (NP-Hard):** Given a set S of positive integers $\{s_1, s_2, \ldots, s_n\}$, partition S into two subsets, A and $S \setminus A$.
- **Objective:** Minimize the absolute difference (d) between the sum of elements in A and the sum of elements in $S \setminus A$.

$$d = \left| \sum_{s_i \in A} s_i - \sum_{s_j \in S \setminus A} s_j \right|$$

- **Goal:** Make the sums of the two subsets as close as possible.

Modeling with Binary Variables

Decision Variable $x_i \in \{0, 1\}$:

- $x_i = 1 \implies s_i$ belongs to set A.
- $x_i = 0 \implies s_i$ belongs to set $S \setminus A$.

Let c be the total sum of all elements in S.

Number Partitioning QUBO

- **Sums of the Two Partitions:**

$$\text{Sum}(A) = \sum_{i=1}^{n} s_i x_i \qquad \text{Sum}(S \setminus A) = c - \sum_{i=1}^{n} s_i x_i$$

Number Partitioning QUBO

- **Sums of the Two Partitions:**

$$\text{Sum}(A) = \sum_{i=1}^{n} s_i x_i \qquad \text{Sum}(S \setminus A) = c - \sum_{i=1}^{n} s_i x_i$$

- **The Difference (d):** The difference d between these two sums:

$$d = \left(\sum_i s_i x_i \right) - \left(c - \sum_i s_i x_i \right) = 2 \sum_{i=1}^{n} s_i x_i - c$$

Number Partitioning QUBO

- **Sums of the Two Partitions:**

$$\text{Sum}(A) = \sum_{i=1}^{n} s_i x_i \qquad \text{Sum}(S \setminus A) = c - \sum_{i=1}^{n} s_i x_i$$

- **The Difference (d):** The difference d between these two sums:

$$d = \left(\sum_i s_i x_i \right) - \left(c - \sum_i s_i x_i \right) = 2 \sum_{i=1}^{n} s_i x_i - c$$

- **QUBO Objective:** Since we want to minimize the absolute difference $|d|$, the equivalent unconstrained binary optimization is to minimize the square of the difference:

$$\min_{\mathbf{x} \in \{0,1\}^n} d^2 = \left(2 \sum_{i=1}^{n} s_i x_i - c \right)^2$$

Number Partitioning QUBO (continued)

- **Goal:** Express the squared difference as the QUBO quadratic form, $\min \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x}$ (ignoring the constant term c^2 from expansion).

$$\left(2\sum_{i=1}^{n} s_i x_i - c\right)^2 \propto \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x}$$

Number Partitioning QUBO (continued)

- **Goal:** Express the squared difference as the QUBO quadratic form, $\min \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x}$ (ignoring the constant term c^2 from expansion).

$$\left(2\sum_{i=1}^{n} s_i x_i - c\right)^2 \propto \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x}$$

- **QUBO Matrix Coefficients (q_{ij}):** The coefficients are derived from the squared objective function, where $\mathbf{Q}$ is a symmetric matrix.

$$q_{ij} = \begin{cases} \mathbf{s_i(s_i - c)} & \text{if } i = j \quad \text{(Diagonal, linear term in } x_i\text{)} \\ \mathbf{2s_is_j} & \text{if } i \neq j \quad \text{(Off-diagonal, quadratic term } x_i x_j\text{)} \end{cases}$$

Number Partitioning QUBO (continued)

- **Goal:** Express the squared difference as the QUBO quadratic form, $\min \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x}$ (ignoring the constant term c^2 from expansion).

$$\left(2\sum_{i=1}^{n} s_i x_i - c\right)^2 \propto \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{x}$$

- **QUBO Matrix Coefficients (q_{ij}):** The coefficients are derived from the squared objective function, where $\mathbf{Q}$ is a symmetric matrix.

$$q_{ij} = \begin{cases} \mathbf{s_i(s_i - c)} & \text{if } i = j \quad \text{(Diagonal, linear term in } x_i\text{)} \\ \mathbf{2s_is_j} & \text{if } i \neq j \quad \text{(Off-diagonal, quadratic term } x_i x_j\text{)} \end{cases}$$

- **Significance:** This matrix $\mathbf{Q}$ is the input for all subsequent algorithms (SA, QA, QAOA).

Section 3

Practical QUBO Formulation

Cancer Genomics Pathways

Identifying Driver Mutations from TCGA Data

Practical Problem: Cancer Genomics (TCGA)

- **Goal:** The *de novo* identification of altered cancer pathways from gene mutation data (e.g., The Cancer Genome Atlas - TCGA).

Practical Problem: Cancer Genomics (TCGA)

- **Goal:** The *de novo* identification of altered cancer pathways from gene mutation data (e.g., The Cancer Genome Atlas - TCGA).
- **Problem Type:** This complex practical problem non-trivially reduces to the Independent Set problem, meaning it is NP-Complete and a suitable candidate for quantum optimization.

Practical Problem: Cancer Genomics (TCGA)

- **Goal:** The *de novo* identification of altered cancer pathways from gene mutation data (e.g., The Cancer Genome Atlas - TCGA).
- **Problem Type:** This complex practical problem non-trivially reduces to the Independent Set problem, meaning it is NP-Complete and a suitable candidate for quantum optimization.
- **Data Modeling: Hypergraph**
  - Genes (g_i) are the vertices.
  - Patients (P_j) are the hyperedges (groups of mutated genes).
  - Modeled by the **Incidence Matrix (B)** where $b_{ij} = 1$ if gene i is mutated in patient j.

Practical Problem: Cancer Genomics (TCGA)

- **Goal:** The *de novo* identification of altered cancer pathways from gene mutation data (e.g., The Cancer Genome Atlas - TCGA).
- **Problem Type:** This complex practical problem non-trivially reduces to the Independent Set problem, meaning it is NP-Complete and a suitable candidate for quantum optimization.
- **Data Modeling: Hypergraph**
  - Genes (g_i) are the vertices.
  - Patients (P_j) are the hyperedges (groups of mutated genes).
  - Modeled by the **Incidence Matrix (B)** where $b_{ij} = 1$ if gene i is mutated in patient j.
- **Graph Laplacian:** The gene-gene correlation matrix is derived:

$$\mathbf{L}^{+} = \mathbf{B}\mathbf{B}^{T} = \mathbf{D} + \mathbf{A}$$

- The Graph Laplacian is decomposed into two matrices corresponding to two key combinatorial criteria for identifying "driver" mutations:

Cancer Genomics: Criteria for Driver Genes

- The Graph Laplacian is decomposed into two matrices corresponding to two key combinatorial criteria for identifying "driver" mutations:
- **1. Coverage (Maximize $\mathbf{x}^T\mathbf{D}\mathbf{x}$):**
  - We seek genes that are prevalent across a large patient cohort.
  - Modeled by the **Degree Matrix (D)**: A diagonal matrix where d_{ii} is the number of patients affected by gene i.

Cancer Genomics: Criteria for Driver Genes

- The Graph Laplacian is decomposed into two matrices corresponding to two key combinatorial criteria for identifying "driver" mutations:
- **1. Coverage (Maximize $x^T Dx$):**
  - We seek genes that are prevalent across a large patient cohort.
  - Modeled by the **Degree Matrix (D)**: A diagonal matrix where d_{ii} is the number of patients affected by gene i.
- **2. Exclusivity (Minimize $x^T Ax$):**
  - Multiple mutations are unlikely in a single patient for the same pathway.
  - Modeled by the **Adjacency Matrix (A)**: a_{ij} is the number of patients affected by both gene i and gene j.

Cancer Genomics: The QUBO Formulation

- **Decision Vector x:** $x_i = 1$ if gene i in the pathway; $x_i = 0$ otherwise.

Cancer Genomics: The QUBO Formulation

- **Decision Vector x:** $x_i = 1$ if gene i in the pathway; $x_i = 0$ otherwise.
- **Combined Objective:** We must find a pathway that maximizes coverage and minimizes exclusivity. This is formulated as:

$$\min_{\mathbf{x}} \left[(\text{Exclusivity Term}) - \alpha (\text{Coverage Term}) \right]$$

Cancer Genomics: The QUBO Formulation

- **Decision Vector x:** $x_i = 1$ if gene i in the pathway; $x_i = 0$ otherwise.
- **Combined Objective:** We must find a pathway that maximizes coverage and minimizes exclusivity. This is formulated as:

$$\min_{\mathbf{x}} [(\text{Exclusivity Term}) - \alpha(\text{Coverage Term})]$$

- **The Final QUBO Objective:** $\min_{\mathbf{x}} \left[\mathbf{x}^T \mathbf{A}\mathbf{x} - \alpha \mathbf{x}^T \mathbf{D}\mathbf{x}\right]$

Cancer Genomics: The QUBO Formulation

- **Decision Vector x:** $x_i = 1$ if gene i in the pathway; $x_i = 0$ otherwise.
- **Combined Objective:** We must find a pathway that maximizes coverage and minimizes exclusivity. This is formulated as:

$$\min_{\mathbf{x}} [(\text{Exclusivity Term}) - \alpha(\text{Coverage Term})]$$

- **The Final QUBO Objective:** $\min_{\mathbf{x}} \left[\mathbf{x}^T \mathbf{A}\mathbf{x} - \alpha \mathbf{x}^T \mathbf{D}\mathbf{x}\right]$
- **Expanded QUBO Form:** $\min_{\mathbf{x}} \left[\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j - \alpha \sum_{i=1}^{n} d_{ii} x_i\right]$

Cancer Genomics: The QUBO Formulation

- **Decision Vector x:** $x_i = 1$ if gene i in the pathway; $x_i = 0$ otherwise.
- **Combined Objective:** We must find a pathway that maximizes coverage and minimizes exclusivity. This is formulated as:

$$\min_{\mathbf{x}} \left[(\text{Exclusivity Term}) - \alpha(\text{Coverage Term}) \right]$$

- **The Final QUBO Objective:** $\min_{\mathbf{x}} \left[\mathbf{x}^T \mathbf{A} \mathbf{x} - \alpha \mathbf{x}^T \mathbf{D} \mathbf{x} \right]$
- **Expanded QUBO Form:** $\min_{\mathbf{x}} \left[\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j - \alpha \sum_{i=1}^{n} d_{ii} x_i \right]$
- **Penalty Factor α:** The weight $\alpha \geq 1$ reflects that the coverage criterion (the linear term) is more important than exclusivity.

Section 4

Building Blocks of Quantum Computation

Circuits and the Ising Model

Bridging QUBO to Quantum Hardware

Qubits and Superposition

- **Qubits (Quantum Bits):** The fundamental unit of quantum information, analogous to a classical bit. They have two basis states,

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

.

Qubits and Superposition

- **Qubits (Quantum Bits):** The fundamental unit of quantum information, analogous to a classical bit. They have two basis states,

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

- **Superposition:** Unlike classical bits (restricted to 0 or 1), a qubit can exist in a superposition of both states simultaneously:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

α and β are complex probability amplitudes.

Qubits and Superposition

- **Qubits (Quantum Bits):** The fundamental unit of quantum information, analogous to a classical bit. They have two basis states,

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ and } |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

- **Superposition:** Unlike classical bits (restricted to 0 or 1), a qubit can exist in a superposition of both states simultaneously:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

  α and β are complex probability amplitudes.

- **Born's Rule:** Measurement forces the qubit to collapse to a basis state ($|0\rangle$ or $|1\rangle$) with probabilities:

$$P(0) = |\alpha|^2, \quad P(1) = |\beta|^2, \quad \text{where } |\alpha|^2 + |\beta|^2 = 1$$

Visualizing Qubit States (Bloch Sphere)

- **Bloch Sphere:** A geometrical representation of a pure single-qubit state, where the surface represents all possible states.

Visualizing Qubit States (Bloch Sphere)

- **Bloch Sphere:** A geometrical representation of a pure single-qubit state, where the surface represents all possible states.
- The poles correspond to the computational basis states.

Visualizing Qubit States (Bloch Sphere)

- **Bloch Sphere:** A geometrical representation of a pure single-qubit state, where the surface represents all possible states.
- The poles correspond to the computational basis states.
- Any point on the surface is a superposition state $|\psi\rangle$.

Visualizing Qubit States (Bloch Sphere)

- **Bloch Sphere:** A geometrical representation of a pure single-qubit state, where the surface represents all possible states.
- The poles correspond to the computational basis states.
- Any point on the surface is a superposition state $|\psi\rangle$.

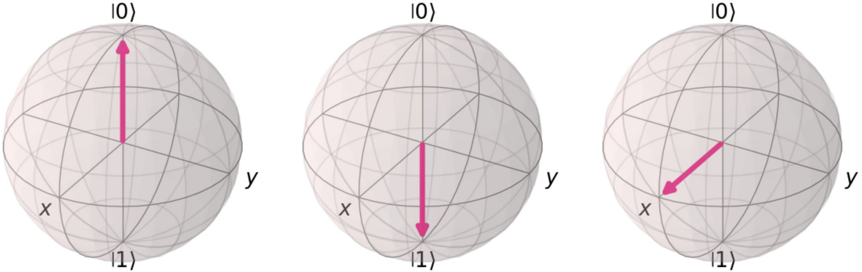


Figure: Bloch Sphere representations of $|0\rangle, |1\rangle$, and $|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$

Single-Qubit Gates

- **Quantum Gates:** These are **Unitary Operators** ($\mathbf{U}$) that act as rotations and reflections on the single-qubit state vector.

Single-Qubit Gates

- **Quantum Gates:** These are **Unitary Operators** (**U**) that act as rotations and reflections on the single-qubit state vector.
- **Key Single-Qubit Gates:**
  - **Hadamard (*H*):** Creates a uniform superposition from a basis state. It is crucial for initial state preparation.

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$$

Single-Qubit Gates

- **Quantum Gates:** These are **Unitary Operators (U)** that act as rotations and reflections on the single-qubit state vector.
- **Key Single-Qubit Gates:**
  - **Hadamard (H):** Creates a uniform superposition from a basis state. It is crucial for initial state preparation.

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$$

  - **Pauli Gates ($\sigma_X, \sigma_Y, \sigma_Z$):** Implement 180° rotations around the X, Y, and Z axes on the Bloch sphere.

Single-Qubit Gates

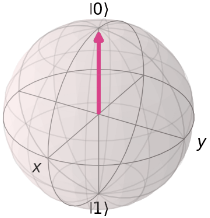
- **Quantum Gates:** These are **Unitary Operators** (**U**) that act as rotations and reflections on the single-qubit state vector.
- **Key Single-Qubit Gates:**
  - **Hadamard (H):** Creates a uniform superposition from a basis state. It is crucial for initial state preparation.

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle$$

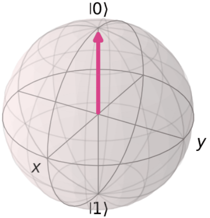
  - **Pauli Gates ($\sigma_X, \sigma_Y, \sigma_Z$):** Implement 180° rotations around the X, Y, and Z axes on the Bloch sphere.
  - **Rotation Gates ($R_X(\theta), R_Y(\theta), R_Z(\theta)$):** Implement arbitrary **parameterized rotations** around the axes.
    - ★ These parameterized gates are the core components optimized by the classical loop in Variational Quantum Algorithms (VQAs) like QAOA.

Visualizing Single-Qubit Gates

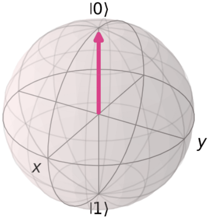
Animations of key single-qubit gates acting on the Bloch sphere



Hadamard (H)



Pauli-X (σ_X)



Rotation $R_X(\pi/2)$

Naturally all gates are reversible (except measurement!).

Multi-Qubit Gates

- **Multi-Qubit Gates:** Operators that act on two or more qubits simultaneously, creating correlations between them.

Multi-Qubit Gates

- **Multi-Qubit Gates:** Operators that act on two or more qubits simultaneously, creating correlations between them.
- **CNOT (CX):** The fundamental gate for creating **entanglement** (a non-classical correlation).

Multi-Qubit Gates

- **Multi-Qubit Gates:** Operators that act on two or more qubits simultaneously, creating correlations between them.
- **CNOT (CX):** The fundamental gate for creating **entanglement** (a non-classical correlation).
  - The state of the target qubit is flipped only if the control qubit is $|1\rangle$.

Multi-Qubit Gates

- **Multi-Qubit Gates:** Operators that act on two or more qubits simultaneously, creating correlations between them.
- **CNOT (CX):** The fundamental gate for creating **entanglement** (a non-classical correlation).
  - The state of the target qubit is flipped only if the control qubit is $|1\rangle$.
  - **Controlled-Z (CZ):** Flips the phase of the target qubit only if the control qubit is $|1\rangle$.

Multi-Qubit Gates

- **Multi-Qubit Gates:** Operators that act on two or more qubits simultaneously, creating correlations between them.
- **CNOT (CX):** The fundamental gate for creating **entanglement** (a non-classical correlation).
  - The state of the target qubit is flipped only if the control qubit is $|1\rangle$.
  - **Controlled-Z (CZ):** Flips the phase of the target qubit only if the control qubit is $|1\rangle$.
- **QUBO Interaction Gate ($R_{ZZ}(\theta)$):**
  - This gate applies a phase shift based on the correlation (or alignment) of the two qubits' Z-states.
  - It is essential for implementing the Cost Hamiltonian in quantum optimization algorithms, as it directly models the two-body interaction terms ($x_i x_j$) present in QUBOs.

Quantum Circuits

- **Definition:** A quantum circuit is a conceptual model representing a sequence of quantum gates applied to an initial state of qubits.

Quantum Circuits

- **Definition:** A quantum circuit is a conceptual model representing a sequence of quantum gates applied to an initial state of qubits.
- **Execution Order:**
  - Circuits are typically read and drawn **left-to-right** (time evolution).

Quantum Circuits

- **Definition:** A quantum circuit is a conceptual model representing a sequence of quantum gates applied to an initial state of qubits.
- **Execution Order:**
  - Circuits are typically read and drawn **left-to-right** (time evolution).
  - Mathematically, the corresponding unitary operators ($\mathbf{U}_i$) are multiplied in the reverse order (**right-to-left**) due to matrix multiplication:

$$|\psi_{\text{out}}\rangle = \mathbf{U}_L \mathbf{U}_{L-1} \cdots \mathbf{U}_1 |\psi_{\text{in}}\rangle$$

Quantum Circuits

- **Definition:** A quantum circuit is a conceptual model representing a sequence of quantum gates applied to an initial state of qubits.
- **Execution Order:**
  - Circuits are typically read and drawn **left-to-right** (time evolution).
  - Mathematically, the corresponding unitary operators ($\mathbf{U}_i$) are multiplied in the reverse order (**right-to-left**) due to matrix multiplication:

$$|\psi_{\text{out}}\rangle = \mathbf{U}_L \mathbf{U}_{L-1} \cdots \mathbf{U}_1 |\psi_{\text{in}}\rangle$$

- **Tensor Product of States:** When multiple quantum states (each in different Hilbert spaces $\mathcal{H}_i$) or registers are combined, such as $|\psi_1\rangle \in \mathcal{H}_1$, $|\psi_2\rangle \in \mathcal{H}_2$, ..., the joint system is described by:

$$|\psi_1\rangle \otimes |\psi_2\rangle \otimes \cdots \otimes |\psi_n\rangle = |\psi_1 \psi_2 \cdots \psi_n\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \cdots \otimes \mathcal{H}_n$$

$$|01\rangle = |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ 0 \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Quantum Circuits Visualization

- Consider the quantum expression:

$$[(\mathrm{CX}) \cdot (Z \otimes Z) \cdot (X \otimes H)] \left|00\right\rangle$$

- Consider the quantum expression:

$$[(\mathrm{CX}) \cdot (Z \otimes Z) \cdot (X \otimes H)] \, |00\rangle$$

- Step-by-step gate application (right-to-left):
  1. Apply X to $q[0]$ and H to $q[1]$
  2. Then apply Z to both qubits
  3. Finally apply a **CX** gate with control $q[1]$ and target $q[0]$

Quantum Circuits Visualization

- Consider the quantum expression:

$$[(\text{CX}) \cdot (Z \otimes Z) \cdot (X \otimes H)]\,|00\rangle$$

- Step-by-step gate application (right-to-left):
  1. Apply X to $q[0]$ and H to $q[1]$
  2. Then apply Z to both qubits
  3. Finally apply a **CX** gate with control $q[1]$ and target $q[0]$
- This circuit transforms $|00\rangle$ into a specific entangled state.

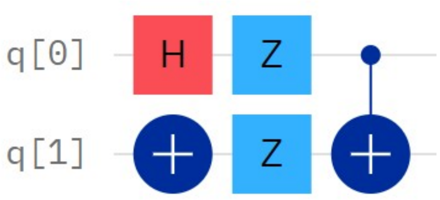


Figure: Quantum circuit represents: $[(\text{CX}) \times (Z \otimes Z) \times (X \otimes H)]|00\rangle$.

Hamiltonian and Quantum Evolution

- **The Hamiltonian (H):**
  - A Hermitian operator representing the energy of a quantum system.
  - Its eigenvalues correspond to the possible **energy levels** of the system.
  - The corresponding eigenvectors are the quantum states.

Hamiltonian and Quantum Evolution

- **The Hamiltonian (H):**
  - A Hermitian operator representing the energy of a quantum system.
  - Its eigenvalues correspond to the possible **energy levels** of the system.
  - The corresponding eigenvectors are the quantum states.
- **The Optimization Goal:** We seek the lowest energy state, known as the **ground state**, which corresponds to the optimal solution.

Hamiltonian and Quantum Evolution

- **The Hamiltonian (H):**
  - A Hermitian operator representing the energy of a quantum system.
  - Its eigenvalues correspond to the possible **energy levels** of the system.
  - The corresponding eigenvectors are the quantum states.
- **The Optimization Goal:** We seek the lowest energy state, known as the **ground state**, which corresponds to the optimal solution.
- **Time Evolution (Schrödinger Equation):** The Hamiltonian governs how a quantum state $|\psi(t)\rangle$ changes over time:

$$i\hbar \frac{\partial}{\partial t}|\psi(t)\rangle = H|\psi(t)\rangle$$

Hamiltonian and Quantum Evolution

- **The Hamiltonian (H):**
  - A Hermitian operator representing the energy of a quantum system.
  - Its eigenvalues correspond to the possible **energy levels** of the system.
  - The corresponding eigenvectors are the quantum states.
- **The Optimization Goal:** We seek the lowest energy state, known as the **ground state**, which corresponds to the optimal solution.
- **Time Evolution (Schrödinger Equation):** The Hamiltonian governs how a quantum state $|\psi(t)\rangle$ changes over time:

$$i\hbar \frac{\partial}{\partial t}|\psi(t)\rangle = H|\psi(t)\rangle$$

- **Relevance to QA:** This continuous time evolution is the basis for Quantum Annealing (QA), where the system is slowly steered from a known initial state to the problem's ground state.

Ising Model and QUBO

- **Ising Model:** A mathematical tool from physics (ferromagnetism) that models systems with interacting "spins" ($\sigma_i \in \{-1, 1\}$).

Ising Model and QUBO

- **Ising Model:** A mathematical tool from physics (ferromagnetism) that models systems with interacting "spins" ($\sigma_i \in \{-1, 1\}$).
- **Ising Hamiltonian (H_{Ising}):** The energy function is minimized when solving the Ising problem:

$$H(\sigma) = -\sum_{i<j} J_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i$$

  - The first term represents two-body interactions (J_{ij}).
  - The second term represents external biases (h_i).

Ising Model and QUBO

- **Ising Model:** A mathematical tool from physics (ferromagnetism) that models systems with interacting "spins" ($\sigma_i \in \{-1, 1\}$).
- **Ising Hamiltonian (H_{Ising}):** The energy function is minimized when solving the Ising problem:

$$H(\sigma) = -\sum_{i<j} J_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i$$

  - The first term represents two-body interactions (J_{ij}).
  - The second term represents external biases (h_i).
- **The Critical Link: QUBO-Ising Equivalence**
  - QUBO (binary variables $x_i \in \{0, 1\}$) is directly convertible to the Ising Model (spin variables $\sigma_i \in \{-1, 1\}$).
  - The substitution is: $x_i = \frac{1+\sigma_i}{2}$.

Section 5

Optimization Landscape

Algorithms: Simulated Annealing (SA) - Classical

- **Analogy:** SA is a heuristic inspired by the physical process of **annealing** (gradual cooling to reach a stable, low-energy state).

Algorithms: Simulated Annealing (SA) - Classical

- **Analogy:** SA is a heuristic inspired by the physical process of **annealing** (gradual cooling to reach a stable, low-energy state).
- **Goal:** Find a high-quality heuristic solution (low-cost, or low-energy) to an optimization problem, such as a QUBO.

Algorithms: Simulated Annealing (SA) - Classical

- **Analogy:** SA is a heuristic inspired by the physical process of **annealing** (gradual cooling to reach a stable, low-energy state).
- **Goal:** Find a high-quality heuristic solution (low-cost, or low-energy) to an optimization problem, such as a QUBO.
- **Process Overview:**
  1. Map the target problem to a **cost function** $f(\mathbf{x})$ (energy).
  2. Initialize with a random solution $\mathbf{x}$ and a high temperature T.
  3. Iteratively generate a **neighboring solution** $\mathbf{x}'$ by applying small perturbations.

Algorithms: Simulated Annealing (SA) - Classical

- **Analogy:** SA is a heuristic inspired by the physical process of **annealing** (gradual cooling to reach a stable, low-energy state).
- **Goal:** Find a high-quality heuristic solution (low-cost, or low-energy) to an optimization problem, such as a QUBO.
- **Process Overview:**
  1. Map the target problem to a **cost function** $f(\mathbf{x})$ (energy).
  2. Initialize with a random solution $\mathbf{x}$ and a high temperature T.
  3. Iteratively generate a **neighboring solution** $\mathbf{x}'$ by applying small perturbations.
- **Key Feature: Escaping Local Minima**
  - If the neighbor is better ($\Delta E < 0$), accept it deterministically.
  - If the neighbor is worse ($\Delta E > 0$), accept it **probabilistically**. This resistance to sticking to local minima is what sets SA apart.

Simulated Annealing: Temperature Schedule and Cooling

- **Acceptance Probability:** The likelihood of accepting a worse solution $\mathbf{x}'$ ($\Delta E = f(\mathbf{x}') - f(\mathbf{x}) > 0$) is given by the formula derived from thermal annealing:

$$p_{\text{accept}} = \exp\left(-\frac{\Delta E}{T}\right)$$

Simulated Annealing: Temperature Schedule and Cooling

- **Acceptance Probability:** The likelihood of accepting a worse solution $\mathbf{x}'$ ($\Delta E = f(\mathbf{x}') - f(\mathbf{x}) > 0$) is given by the formula derived from thermal annealing:

$$p_{\text{accept}} = \exp\left(-\frac{\Delta E}{T}\right)$$

- **Exploration vs. Exploitation:**
  - **High T (Start):** p_{accept} is high. The algorithm explores widely, accepting many worse moves.
  - **Low T (End):** p_{accept} is low. The algorithm mostly accepts better moves, exploiting the best solution found so far.

Simulated Annealing: Temperature Schedule and Cooling

- **Acceptance Probability:** The likelihood of accepting a worse solution $\mathbf{x}'$ ($\Delta E = f(\mathbf{x}') - f(\mathbf{x}) > 0$) is given by the formula derived from thermal annealing:
$$p_{\text{accept}} = \exp\left(-\frac{\Delta E}{T}\right)$$
- **Exploration vs. Exploitation:**
  - **High T (Start):** p_{accept} is high. The algorithm explores widely, accepting many worse moves.
  - **Low T (End):** p_{accept} is low. The algorithm mostly accepts better moves, exploiting the best solution found so far.
- **Cooling Process:** T is gradually reduced at each step using a **cooling factor** α ($0 < \alpha < 1$): $T_{\text{new}} = \alpha \cdot T_{\text{old}}$

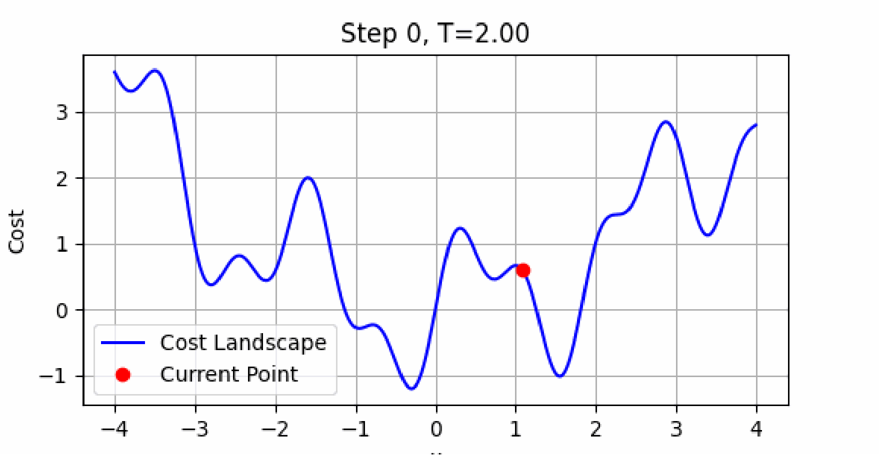
Simulated Annealing: Temperature Schedule and Cooling

- **Acceptance Probability:** The likelihood of accepting a worse solution $\mathbf{x}'$ ($\Delta E = f(\mathbf{x}') - f(\mathbf{x}) > 0$) is given by the formula derived from thermal annealing:

$$p_{\text{accept}} = \exp\left(-\frac{\Delta E}{T}\right)$$

- **Exploration vs. Exploitation:**
  - **High T (Start):** p_{accept} is high. The algorithm explores widely, accepting many worse moves.
  - **Low T (End):** p_{accept} is low. The algorithm mostly accepts better moves, exploiting the best solution found so far.
- **Cooling Process:** T is gradually reduced at each step using a **cooling factor** α ($0 < \alpha < 1$): $T_{\text{new}} = \alpha \cdot T_{\text{old}}$
- **Trade-offs:**
  - **Advantage:** Simple and effective at escaping local minima.
  - **Limitation:** Performance is highly dependent on careful tuning of parameters (initial T, α, iterations per temperature).

Simulated Annealing Visualization



Algorithms: Quantum Annealing (QA) - Quantum

- **Type:** A example of **Adiabatic Quantum Computing (AQC)**.

Algorithms: Quantum Annealing (QA) - Quantum

- **Type:** A example of **Adiabatic Quantum Computing (AQC)**.
- **Goal:** Find the ground state of a system, which corresponds to the optimal solution of a QUBO/Ising problem.

Algorithms: Quantum Annealing (QA) - Quantum

- **Type:** A example of **Adiabatic Quantum Computing (AQC)**.
- **Goal:** Find the ground state of a system, which corresponds to the optimal solution of a QUBO/Ising problem.
- **Mechanism:** Leverages Adiabatic Theorem and Quantum Tunneling.

Algorithms: Quantum Annealing (QA) - Quantum

- **Type:** A example of **Adiabatic Quantum Computing (AQC)**.
- **Goal:** Find the ground state of a system, which corresponds to the optimal solution of a QUBO/Ising problem.
- **Mechanism:** Leverages Adiabatic Theorem and Quantum Tunneling.
- **Hardware:** Highly specialized for optimization (D-Wave QPUs).

Algorithms: Quantum Annealing (QA) - Quantum

- **Type:** A example of **Adiabatic Quantum Computing (AQC)**.
- **Goal:** Find the ground state of a system, which corresponds to the optimal solution of a QUBO/Ising problem.
- **Mechanism:** Leverages Adiabatic Theorem and Quantum Tunneling.
- **Hardware:** Highly specialized for optimization (D-Wave QPUs).

QA vs. SA: The Quantum Advantage

- Simulated Annealing (SA) must climb energy barriers.
- QA uses **quantum tunneling** to bypass energy barriers, potentially reaching the global optimum more efficiently.

Algorithms: Quantum Annealing (QA) - Quantum

- **Type:** A example of **Adiabatic Quantum Computing (AQC)**.
- **Goal:** Find the ground state of a system, which corresponds to the optimal solution of a QUBO/Ising problem.
- **Mechanism:** Leverages Adiabatic Theorem and Quantum Tunneling.
- **Hardware:** Highly specialized for optimization (D-Wave QPUs).

QA vs. SA: The Quantum Advantage

- Simulated Annealing (SA) must climb energy barriers.
- QA uses **quantum tunneling** to bypass energy barriers, potentially reaching the global optimum more efficiently.

Mapping the Problem

The QUBO problem $\mathbf{x}^T\mathbf{Q}\mathbf{x}$ is mapped to the equivalent **Ising Hamiltonian** H_C, where the variables are quantum spins $\sigma_i \in \{-1, 1\}$.

Quantum Annealing: The Adiabatic Theorem

- **Core Principle:** A quantum system initially in the ground state of a time-dependent Hamiltonian $\mathbf{H}(t)$ will **remain in its instantaneous ground state** throughout the evolution, provided the evolution is **sufficiently slow** (adiabaticity).

Quantum Annealing: The Adiabatic Theorem

- **Core Principle:** A quantum system initially in the ground state of a time-dependent Hamiltonian $\mathbf{H}(t)$ will **remain in its instantaneous ground state** throughout the evolution, provided the evolution is **sufficiently slow** (adiabaticity).
- **Time-Dependent Hamiltonian:** The process is governed by two combined Hamiltonians:

$$\mathbf{H}(s(t)) = (1 - s(t))\mathbf{H}_D + s(t)\mathbf{H}_C$$

  - $s(t) \in [0, 1]$ is the monotonic scheduling function.
  - At $t = 0$ $(s = 0)$, $\mathbf{H} = \mathbf{H}_D$.
  - At $t = T$ $(s = 1)$, $\mathbf{H} = \mathbf{H}_C$.

Quantum Annealing: The Adiabatic Theorem

- **Core Principle:** A quantum system initially in the ground state of a time-dependent Hamiltonian $\mathbf{H}(t)$ will **remain in its instantaneous ground state** throughout the evolution, provided the evolution is **sufficiently slow** (adiabaticity).
- **Time-Dependent Hamiltonian:** The process is governed by two combined Hamiltonians:

$$\mathbf{H}(s(t)) = (1 - s(t))\mathbf{H}_D + s(t)\mathbf{H}_C$$

  - $s(t) \in [0, 1]$ is the monotonic scheduling function.
  - At $t = 0$ $(s = 0)$, $\mathbf{H} = \mathbf{H}_D$.
  - At $t = T$ $(s = 1)$, $\mathbf{H} = \mathbf{H}_C$.
- **Driver Hamiltonian ($\mathbf{H}_D$):** A simple Hamiltonian whose ground state is easy to prepare (usually uniform superposition):

$$\mathbf{H}_D = -\sum_i \sigma_x^{(i)}$$

Quantum Annealing: The Adiabatic Theorem

- **Core Principle:** A quantum system initially in the ground state of a time-dependent Hamiltonian $\mathbf{H}(t)$ will **remain in its instantaneous ground state** throughout the evolution, provided the evolution is **sufficiently slow** (adiabaticity).
- **Time-Dependent Hamiltonian:** The process is governed by two combined Hamiltonians:

$$\mathbf{H}(s(t)) = (1 - s(t))\mathbf{H}_D + s(t)\mathbf{H}_C$$

  - $s(t) \in [0, 1]$ is the monotonic scheduling function.
  - At $t = 0$ $(s = 0)$, $\mathbf{H} = \mathbf{H}_D$.
  - At $t = T$ $(s = 1)$, $\mathbf{H} = \mathbf{H}_C$.
- **Driver Hamiltonian ($\mathbf{H}_D$):** A simple Hamiltonian whose ground state is easy to prepare (usually uniform superposition):

$$\mathbf{H}_D = -\sum_i \sigma_x^{(i)}$$

- **Cost Hamiltonian ($\mathbf{H}_C$):** Encodes the optimization problem.

Quantum Annealing: Spectral Gap and Speed

- **The Spectral Gap ($\Delta(s)$):** The energy difference of the ground state (E_0) and first excited state (E_1) of the Hamiltonian $\mathbf{H}(s)$.

$$\Delta(s) = E_1(s) - E_0(s)$$

Quantum Annealing: Spectral Gap and Speed

- **The Spectral Gap ($\Delta(s)$):** The energy difference of the ground state (E_0) and first excited state (E_1) of the Hamiltonian $\mathbf{H}(s)$.

$$\Delta(s) = E_1(s) - E_0(s)$$

- **Minimum Gap (Δ_{min}):** The min gap over the entire annealing path.
  - The minimum gap often occurs where the problem is hardest (a "quantum critical point").

Quantum Annealing: Spectral Gap and Speed

- **The Spectral Gap ($\Delta(s)$):** The energy difference of the ground state (E_0) and first excited state (E_1) of the Hamiltonian $\mathbf{H}(s)$.

$$\Delta(s) = E_1(s) - E_0(s)$$

- **Minimum Gap (Δ_{min}):** The min gap over the entire annealing path.
    - The minimum gap often occurs where the problem is hardest (a "quantum critical point").
- **Adiabatic Condition:** To ensure the system remains in the ground state (and thus finds the optimal solution), the total annealing time T must be long enough:

$$T \gg \frac{1}{\Delta_{\text{min}}^2}$$

Quantum Annealing: Spectral Gap and Speed

- **The Spectral Gap ($\Delta(s)$):** The energy difference of the ground state (E_0) and first excited state (E_1) of the Hamiltonian $\mathbf{H}(s)$.

$$\Delta(s) = E_1(s) - E_0(s)$$

- **Minimum Gap (Δ_{min}):** The min gap over the entire annealing path.
  - The minimum gap often occurs where the problem is hardest (a "quantum critical point").
- **Adiabatic Condition:** To ensure the system remains in the ground state (and thus finds the optimal solution), the total annealing time T must be long enough:

$$T \gg \frac{1}{\Delta_{\text{min}}^2}$$

- **Implication:** A smaller min gap requires a much longer annealing time T to avoid exciting the system into a non-optimal state.

Quantum Annealing: Workflow (D-Wave Example)

- **Initial State ($s = 0$):** The Hamiltonian is dominated by $\mathbf{H}_D$ (Pauli-X terms), forcing all qubits into a uniform superposition.

Quantum Annealing: Workflow (D-Wave Example)

- **Initial State ($s = 0$):** The Hamiltonian is dominated by $\mathbf{H}_D$ (Pauli-X terms), forcing all qubits into a uniform superposition.
- **Annealing Process ($0 < s < 1$):** The coefficients $A(s)$ (Driver) decrease and $B(s)$ (Cost) increase. The energy landscape gradually deforms from a simple, flat landscape to the complex, spiked landscape defined by $\mathbf{H}_C$.

Quantum Annealing: Workflow (D-Wave Example)

- **Initial State ($s = 0$):** The Hamiltonian is dominated by $\mathbf{H}_D$ (Pauli-X terms), forcing all qubits into a uniform superposition.
- **Annealing Process ($0 < s < 1$):** The coefficients $A(s)$ (Driver) decrease and $B(s)$ (Cost) increase. The energy landscape gradually deforms from a simple, flat landscape to the complex, spiked landscape defined by $\mathbf{H}_C$.
- **The $\mathbf{H}_{\text{Ising}}$ Combination:**

$$H_{\text{Ising}} = \underbrace{-\frac{A(s)}{2}\sum_i \sigma_X^{(i)}}_{\text{Initial Driver Hamiltonian}} + \underbrace{\frac{B(s)}{2}\left(\sum_i h_i \sigma_Z^{(i)} + \sum_{i<j} J_{ij}\sigma_Z^{(i)}\sigma_Z^{(j)}\right)}_{\text{Final Cost Hamiltonian}}$$

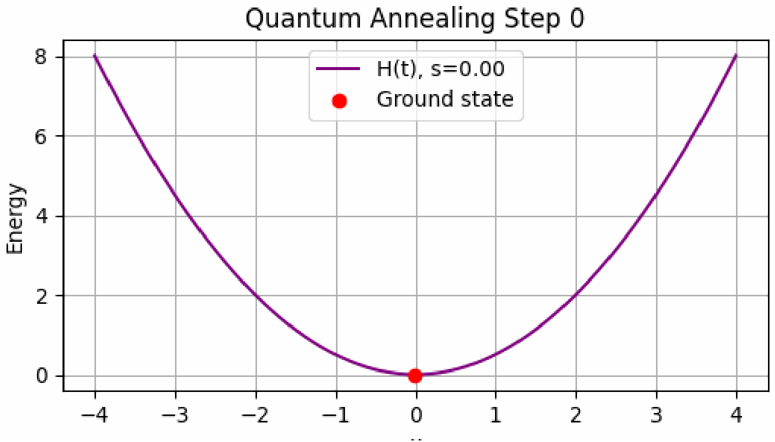
Quantum Annealing: Workflow (D-Wave Example)

- **Initial State ($s = 0$):** The Hamiltonian is dominated by $\mathbf{H}_D$ (Pauli-X terms), forcing all qubits into a uniform superposition.
- **Annealing Process ($0 < s < 1$):** The coefficients $A(s)$ (Driver) decrease and $B(s)$ (Cost) increase. The energy landscape gradually deforms from a simple, flat landscape to the complex, spiked landscape defined by $\mathbf{H}_C$.
- **The $\mathbf{H}_{\text{Ising}}$ Combination:**

$$H_{\text{Ising}} = \underbrace{-\frac{A(s)}{2}\sum_i \sigma_X^{(i)}}_{\text{Initial Driver Hamiltonian}} + \underbrace{\frac{B(s)}{2}\left(\sum_i h_i \sigma_Z^{(i)} + \sum_{i<j} J_{ij}\sigma_Z^{(i)}\sigma_Z^{(j)}\right)}_{\text{Final Cost Hamiltonian}}$$

- **Final State ($s = 1$):** The Hamiltonian is dominated by $\mathbf{H}_C$ (Pauli-Z terms). The qubits collapse to the configuration that minimizes this energy, yielding the optimal QUBO solution.

Quantum Annealing Visualization



Algorithms: Quantum Approximate Optimization Algorithm (QAOA) - Quantum

- **Type:** A Quantum-Classical Hybrid Algorithm designed for combinatorial optimization problems (e.g., QUBOs).

Algorithms: Quantum Approximate Optimization Algorithm (QAOA) - Quantum

- **Type:** A Quantum-Classical Hybrid Algorithm designed for combinatorial optimization problems (e.g., QUBOs).
- **NISQ Era Algorithm:** It has a relatively low circuit depth, making it more resilient to **decoherence** on current Noisy Intermediate-Scale Quantum (NISQ) devices.

Algorithms: Quantum Approximate Optimization Algorithm (QAOA) - Quantum

- **Type:** A Quantum-Classical Hybrid Algorithm designed for combinatorial optimization problems (e.g., QUBOs).
- **NISQ Era Algorithm:** It has a relatively low circuit depth, making it more resilient to **decoherence** on current Noisy Intermediate-Scale Quantum (NISQ) devices.
- **Motivation: Discretizing Quantum Annealing (QA)**
  - QA relies on continuous-time evolution, which isn't natively digital.
  - QAOA uses **Trotterization** to simulate this continuous evolution using alternating, repeated gate sequences (ansatz).

Algorithms: Quantum Approximate Optimization Algorithm (QAOA) - Quantum

- **Type:** A Quantum-Classical Hybrid Algorithm designed for combinatorial optimization problems (e.g., QUBOs).
- **NISQ Era Algorithm:** It has a relatively low circuit depth, making it more resilient to **decoherence** on current Noisy Intermediate-Scale Quantum (NISQ) devices.
- **Motivation: Discretizing Quantum Annealing (QA)**
  - QA relies on continuous-time evolution, which isn't natively digital.
  - QAOA uses **Trotterization** to simulate this continuous evolution using alternating, repeated gate sequences (ansatz).
- **Workflow:** A quantum circuit generates a state, and a optimizer tunes the circuit's parameters to minimize the expected cost.

QAOA: The Parametrized Quantum Circuit

- **Initial State Preparation:** The circuit begins by applying a Hadamard gate (H) to all n qubits, creating a uniform superposition of all 2^n possible solutions:

$$|\psi(0)\rangle = |+\rangle^{\otimes n}$$

QAOA: The Parametrized Quantum Circuit

- **Initial State Preparation:** The circuit begins by applying a Hadamard gate (H) to all n qubits, creating a uniform superposition of all 2^n possible solutions:

$$|\psi(0)\rangle = |+\rangle^{\otimes n}$$

- **Alternating Operators:** The core consists of p repeated layers of operators derived from the Hamiltonians used in Quantum Annealing:
  1. **Cost Operator ($e^{-i\gamma H_C}$):** Encodes the objective function.
  2. **Mixer Operator ($e^{-i\beta H_D}$):** Explores the solution space.

QAOA: The Parametrized Quantum Circuit

- **Initial State Preparation:** The circuit begins by applying a Hadamard gate (H) to all n qubits, creating a uniform superposition of all 2^n possible solutions:

$$|\psi(0)\rangle = |+\rangle^{\otimes n}$$

- **Alternating Operators:** The core consists of p repeated layers of operators derived from the Hamiltonians used in Quantum Annealing:
    1. **Cost Operator ($e^{-i\gamma H_C}$):** Encodes the objective function.
    2. **Mixer Operator ($e^{-i\beta H_D}$):** Explores the solution space.
- **The Ansatz (U):** The quantum state after p layers is:

$$|\psi(\vec{\beta}, \vec{\gamma})\rangle = \prod_{i=1}^{p} e^{-i\beta_i H_D} e^{-i\gamma_i H_C} |\psi(0)\rangle$$

- Parameterized by **2p** angles: $\vec{\beta} = \{\beta_1, \ldots, \beta_p\}$ and $\vec{\gamma} = \{\gamma_1, \ldots, \gamma_p\}$.

QAOA: Cost Hamiltonian ($\mathbf{H}_C$) - (Problem Encoding)

- **Purpose:** The Cost Hamiltonian ($\mathbf{H}_C$) **encodes the QUBO problem** (the objective function) into the quantum system's energy landscape.

QAOA: Cost Hamiltonian ($\mathbf{H}_C$) - (Problem Encoding)

- **Purpose:** The Cost Hamiltonian ($\mathbf{H}_C$) **encodes the QUBO problem** (the objective function) into the quantum system's energy landscape.
- **Structure:** It uses Pauli-Z operators (σ_Z), as the computational basis states $|0\rangle, |1\rangle$ are eigenstates of σ_Z.

$$\mathbf{H}_C = \sum_i h_i \sigma_Z^{(i)} + \sum_{i<j} J_{ij} \sigma_Z^{(i)} \sigma_Z^{(j)}$$

QAOA: Cost Hamiltonian ($\mathbf{H}_C$) - (Problem Encoding)

- **Purpose:** The Cost Hamiltonian ($\mathbf{H}_C$) **encodes the QUBO problem** (the objective function) into the quantum system's energy landscape.
- **Structure:** It uses Pauli-Z operators (σ_Z), as the computational basis states $|0\rangle, |1\rangle$ are eigenstates of σ_Z.

$$\mathbf{H}_C = \sum_i h_i \sigma_Z^{(i)} + \sum_{i<j} J_{ij} \sigma_Z^{(i)} \sigma_Z^{(j)}$$

- **Coefficients:** The h_i and J_{ij} terms are derived directly from the linear and quadratic coefficients of the QUBO matrix $\mathbf{Q}$.

QAOA: Cost Hamiltonian ($\mathbf{H}_C$) - (Problem Encoding)

- **Purpose:** The Cost Hamiltonian ($\mathbf{H}_C$) **encodes the QUBO problem** (the objective function) into the quantum system's energy landscape.
- **Structure:** It uses Pauli-Z operators (σ_Z), as the computational basis states $|0\rangle, |1\rangle$ are eigenstates of σ_Z.

$$\mathbf{H}_C = \sum_i h_i \sigma_Z^{(i)} + \sum_{i<j} J_{ij} \sigma_Z^{(i)} \sigma_Z^{(j)}$$

- **Coefficients:** The h_i and J_{ij} terms are derived directly from the linear and quadratic coefficients of the QUBO matrix $\mathbf{Q}$.
- **Cost Operator ($e^{-i\gamma H_C}$):** This unitary operator applies the phase encoding the cost, parameterized by γ.

QAOA: Cost Hamiltonian ($\mathbf{H}_C$) - (Problem Encoding)

- **Purpose:** The Cost Hamiltonian ($\mathbf{H}_C$) **encodes the QUBO problem** (the objective function) into the quantum system's energy landscape.
- **Structure:** It uses Pauli-Z operators (σ_Z), as the computational basis states $|0\rangle, |1\rangle$ are eigenstates of σ_Z.

$$\mathbf{H}_C = \sum_i h_i \sigma_Z^{(i)} + \sum_{i<j} J_{ij} \sigma_Z^{(i)} \sigma_Z^{(j)}$$

- **Coefficients:** The h_i and J_{ij} terms are derived directly from the linear and quadratic coefficients of the QUBO matrix $\mathbf{Q}$.
- **Cost Operator ($e^{-i\gamma H_C}$):** This unitary operator applies the phase encoding the cost, parameterized by γ.
- **Implementation:** It is approximated (via Trotterization) using a series of single-qubit Z-rotations and two-qubit U_{ZZ} gates:

$$e^{-i\gamma H_C} \approx \prod_{i<j} e^{-i\gamma J_{ij} \sigma_Z^{(i)} \sigma_Z^{(j)}} \prod_i e^{-i\gamma h_i \sigma_Z^{(i)}}$$

QAOA: Mixer Hamiltonian ($\mathbf{H}_D$) - Exploration

- **Purpose:** The Mixer Hamiltonian ($\mathbf{H}_D$) drives the **exploration** of the solution space, preventing from getting trapped in local minima.

QAOA: Mixer Hamiltonian ($\mathbf{H}_D$) - Exploration

- **Purpose:** The Mixer Hamiltonian ($\mathbf{H}_D$) drives the **exploration** of the solution space, preventing from getting trapped in local minima.
- **Structure:** It's typically a sum of single-qubit Pauli-X operators (σ_X), which cause transitions between $|0\rangle$ and $|1\rangle$:

$$\mathbf{H}_D = -\sum_i \sigma_X^{(i)}$$

QAOA: Mixer Hamiltonian ($\mathbf{H}_D$) - Exploration

- **Purpose:** The Mixer Hamiltonian ($\mathbf{H}_D$) drives the **exploration** of the solution space, preventing from getting trapped in local minima.
- **Structure:** It's typically a sum of single-qubit Pauli-X operators (σ_X), which cause transitions between $|0\rangle$ and $|1\rangle$:

$$\mathbf{H}_D = -\sum_i \sigma_X^{(i)}$$

- **Mixer Operator:** The parameterized unitary operator that executes the mixing step: $e^{-i\beta H_D}$.

QAOA: Mixer Hamiltonian ($\mathbf{H}_D$) - Exploration

- **Purpose:** The Mixer Hamiltonian ($\mathbf{H}_D$) drives the **exploration** of the solution space, preventing from getting trapped in local minima.
- **Structure:** It's typically a sum of single-qubit Pauli-X operators (σ_X), which cause transitions between $|0\rangle$ and $|1\rangle$:

$$\mathbf{H}_D = -\sum_i \sigma_X^{(i)}$$

- **Mixer Operator:** The parameterized unitary operator that executes the mixing step: $e^{-i\beta H_D}$.
- **Implementation:** The operator is implemented using single-qubit **Rotation Gates** around the X-axis (R_X):

$$e^{-i\beta H_D} \approx \prod_i e^{-i\beta\sigma_X^{(i)}}$$

QAOA: Mixer Hamiltonian ($\mathbf{H}_D$) - Exploration

- **Purpose:** The Mixer Hamiltonian ($\mathbf{H}_D$) drives the **exploration** of the solution space, preventing from getting trapped in local minima.
- **Structure:** It's typically a sum of single-qubit Pauli-X operators (σ_X), which cause transitions between $|0\rangle$ and $|1\rangle$:

$$\mathbf{H}_D = -\sum_i \sigma_X^{(i)}$$

- **Mixer Operator:** The parameterized unitary operator that executes the mixing step: $e^{-i\beta H_D}$.
- **Implementation:** The operator is implemented using single-qubit **Rotation Gates** around the X-axis (R_X):

$$e^{-i\beta H_D} \approx \prod_i e^{-i\beta\sigma_X^{(i)}}$$

- **Parameters:** The β angles are part of the $2p$ total parameters optimized by the classical algorithm.

QAOA: The Hybrid Optimization Loop

1. **Quantum Execution:**
   - The circuit $|\psi(\vec{\beta}, \vec{\gamma})\rangle$ is executed on a quantum computer.
   - Measure the state to estimate $\langle\psi|H_C|\psi\rangle$, indicating solution quality.

QAOA: The Hybrid Optimization Loop

1. **Quantum Execution:**
   - The circuit $|\psi(\vec{\beta}, \vec{\gamma})\rangle$ is executed on a quantum computer.
   - Measure the state to estimate $\langle\psi|H_C|\psi\rangle$, indicating solution quality.
2. **Classical Optimization:**
   - A classical optimizer (e.g., gradient descent or heuristic methods) receives the estimated expected value.
   - The optimizer adjusts the $2p$ parameters $(\vec{\beta}, \vec{\gamma})$ to minimize this expected cost.

QAOA: The Hybrid Optimization Loop

1. **Quantum Execution:**
   - The circuit $|\psi(\vec{\beta}, \vec{\gamma})\rangle$ is executed on a quantum computer.
   - Measure the state to estimate $\langle\psi|H_C|\psi\rangle$, indicating solution quality.
2. **Classical Optimization:**
   - A classical optimizer (e.g., gradient descent or heuristic methods) receives the estimated expected value.
   - The optimizer adjusts the $2p$ parameters $(\vec{\beta}, \vec{\gamma})$ to minimize this expected cost.
3. **Iteration:** Steps 1 and 2 are repeated until the parameters converge.

QAOA: The Hybrid Optimization Loop

1. **Quantum Execution:**
   - The circuit $|\psi(\vec{\beta}, \vec{\gamma})\rangle$ is executed on a quantum computer.
   - Measure the state to estimate $\langle\psi|H_C|\psi\rangle$, indicating solution quality.
2. **Classical Optimization:**
   - A classical optimizer (e.g., gradient descent or heuristic methods) receives the estimated expected value.
   - The optimizer adjusts the $2p$ parameters $(\vec{\beta}, \vec{\gamma})$ to minimize this expected cost.
3. **Iteration:** Steps 1 and 2 are repeated until the parameters converge.
4. **Output:** The final, optimized quantum state is measured to obtain the approximate binary solution to the QUBO problem.

QAOA Visualization

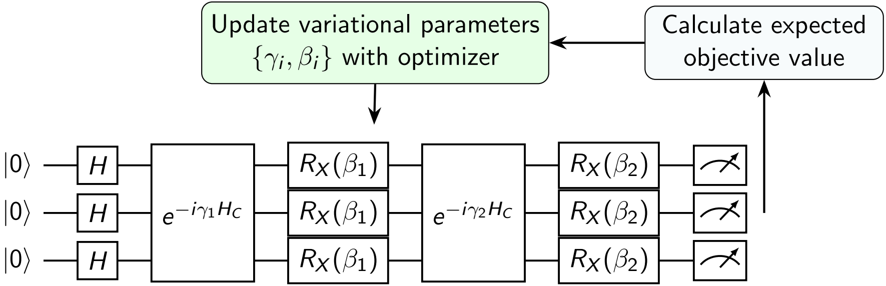


Figure: QAOA Circuit: Each layer alternates between a problem-specific cost unitary $e^{-i\gamma H_C}$ and a mixing unitary $R_X(\beta)$. The parameters $(\gamma_1, \beta_1), (\gamma_2, \beta_2)$ are optimized classically.

Section 6

Implementation and Workflow

Solving QUBOs in Code

From QUBO Matrix to Algorithm Output

Vanilla QAOA: Framework and Input

- **Framework:** Vanilla QAOA utilizes a standard quantum-classical hybrid approach (e.g., using Qiskit) for solving QUBOs.

Vanilla QAOA: Framework and Input

- **Framework:** Vanilla QAOA utilizes a standard quantum-classical hybrid approach (e.g., using Qiskit) for solving QUBOs.
- **Input:** The optimization problem must first be converted into the QUBO form, providing the:
  - Quadratic coefficients ($\mathbf{Q}_{ij}$).
  - Linear coefficients ($\mathbf{c}_i$).

Vanilla QAOA: Framework and Input

- **Framework:** Vanilla QAOA utilizes a standard quantum-classical hybrid approach (e.g., using Qiskit) for solving QUBOs.
- **Input:** The optimization problem must first be converted into the QUBO form, providing the:
  - Quadratic coefficients ($\mathbf{Q}_{ij}$).
  - Linear coefficients ($\mathbf{c}_i$).
- **Parameters:** The circuit's performance depends on $2p$ tunable angles: $\vec{\gamma}$ (Cost) and $\vec{\beta}$ (Mixer). These are determined by the optimizer.

Vanilla QAOA: Cost Operator ($\mathbf{H}_C$) Implementation

- **Goal:** Encode the QUBO objective function into the Cost Hamiltonian ($\mathbf{H}_C$) in the Pauli-Z basis (σ_i^Z).

Vanilla QAOA: Cost Operator ($\mathbf{H}_C$) Implementation

- **Goal:** Encode the QUBO objective function into the Cost Hamiltonian ($\mathbf{H}_C$) in the Pauli-Z basis (σ_i^Z).
- **Mapping $\mathbf{H}_C$ (Pauli-Z basis):**

$$\mathbf{H}_C = \sum_{i,j} \frac{1}{4} Q_{ij} \sigma_i^Z \sigma_j^Z - \sum_i \frac{1}{2} \left(c_i + \sum_j Q_{ij} \right) \sigma_i^Z$$

Vanilla QAOA: Cost Operator ($\mathbf{H}_C$) Implementation

- **Goal:** Encode the QUBO objective function into the Cost Hamiltonian ($\mathbf{H}_C$) in the Pauli-Z basis (σ_i^Z).
- **Mapping $\mathbf{H}_C$ (Pauli-Z basis):**

$$\mathbf{H}_C = \sum_{i,j} \frac{1}{4} Q_{ij} \sigma_i^Z \sigma_j^Z - \sum_i \frac{1}{2} \left(c_i + \sum_j Q_{ij} \right) \sigma_i^Z$$

- **Cost Operator ($e^{-i\gamma \mathbf{H}_C}$):** This unitary applies the cost function's phase to the quantum state, parameterized by γ.

Vanilla QAOA: Cost Operator ($\mathbf{H}_C$) Implementation

- **Goal:** Encode the QUBO objective function into the Cost Hamiltonian ($\mathbf{H}_C$) in the Pauli-Z basis (σ_i^Z).
- **Mapping $\mathbf{H}_C$ (Pauli-Z basis):**

$$\mathbf{H}_C = \sum_{i,j} \frac{1}{4} Q_{ij} \sigma_i^Z \sigma_j^Z - \sum_i \frac{1}{2} \left(c_i + \sum_j Q_{ij} \right) \sigma_i^Z$$

- **Cost Operator ($e^{-i\gamma \mathbf{H}_C}$):** This unitary applies the cost function's phase to the quantum state, parameterized by γ.
- **Implementation:** The operator is constructed using standard quantum gates:
  - Single-qubit $\mathbf{R}_Z$ gates (to handle the linear σ_i^Z terms).
  - Two-qubit $\mathbf{R}_{ZZ}$ gates (to handle the quadratic $\sigma_i^Z \sigma_j^Z$ terms).

Vanilla QAOA: Cost Operator ($\mathbf{H}_C$) Implementation

- **Goal:** Encode the QUBO objective function into the Cost Hamiltonian ($\mathbf{H}_C$) in the Pauli-Z basis (σ_i^Z).
- **Mapping $\mathbf{H}_C$ (Pauli-Z basis):**

$$\mathbf{H}_C = \sum_{i,j} \frac{1}{4} Q_{ij} \sigma_i^Z \sigma_j^Z - \sum_i \frac{1}{2} \left(c_i + \sum_j Q_{ij} \right) \sigma_i^Z$$

- **Cost Operator ($e^{-i\gamma \mathbf{H}_C}$):** This unitary applies the cost function's phase to the quantum state, parameterized by γ.
- **Implementation:** The operator is constructed using standard quantum gates:
  - Single-qubit $\mathbf{R}_Z$ gates (to handle the linear σ_i^Z terms).
  - Two-qubit $\mathbf{R}_{ZZ}$ gates (to handle the quadratic $\sigma_i^Z \sigma_j^Z$ terms).

  The rotation angles are directly proportional to the γ parameter and the respective QUBO coefficients.

Vanilla QAOA: Mixer Operator ($\mathbf{H}_M$) Implementation

- **Goal:** Implement the Mixer Operator ($e^{-i\beta \mathbf{H}_M}$) to induce transitions between basis states, enabling **exploration**.

Vanilla QAOA: Mixer Operator ($\mathbf{H}_M$) Implementation

- **Goal:** Implement the Mixer Operator ($e^{-i\beta \mathbf{H}_M}$) to induce transitions between basis states, enabling **exploration**.
- **Mixer Hamiltonian ($\mathbf{H}_M$):** It is defined as a sum of Pauli-X operators:

$$\mathbf{H}_M = -\sum_{i=1}^{n} \sigma_X^{(i)}$$

Vanilla QAOA: Mixer Operator ($\mathbf{H}_M$) Implementation

- **Goal:** Implement the Mixer Operator ($e^{-i\beta \mathbf{H}_M}$) to induce transitions between basis states, enabling **exploration**.
- **Mixer Hamiltonian ($\mathbf{H}_M$):** It is defined as a sum of Pauli-X operators:

$$\mathbf{H}_M = -\sum_{i=1}^{n} \sigma_X^{(i)}$$

- **Implementation Strategy:** The operator is implemented using simple single-qubit rotations.

Vanilla QAOA: Mixer Operator ($\mathbf{H}_M$) Implementation

- **Goal:** Implement the Mixer Operator ($e^{-i\beta \mathbf{H}_M}$) to induce transitions between basis states, enabling **exploration**.
- **Mixer Hamiltonian ($\mathbf{H}_M$):** It is defined as a sum of Pauli-X operators:

$$\mathbf{H}_M = -\sum_{i=1}^{n} \sigma_X^{(i)}$$

- **Implementation Strategy:** The operator is implemented using simple single-qubit rotations.
- **Gate Used:** Single-qubit Rotation Gates around the X-axis ($\mathbf{R}_X$).
- **The Operator:**

$$e^{-i\beta \mathbf{H}_M} = \prod_{i=1}^{n} \mathbf{R}_X(2\beta)$$

Vanilla QAOA: Mixer Operator ($\mathbf{H}_M$) Implementation

- **Goal:** Implement the Mixer Operator ($e^{-i\beta \mathbf{H}_M}$) to induce transitions between basis states, enabling **exploration**.
- **Mixer Hamiltonian ($\mathbf{H}_M$):** It is defined as a sum of Pauli-X operators:

$$\mathbf{H}_M = -\sum_{i=1}^{n} \sigma_X^{(i)}$$

- **Implementation Strategy:** The operator is implemented using simple single-qubit rotations.
- **Gate Used:** Single-qubit Rotation Gates around the X-axis ($\mathbf{R}_X$).
- **The Operator:**

$$e^{-i\beta \mathbf{H}_M} = \prod_{i=1}^{n} \mathbf{R}_X(2\beta)$$

- **Parameters:** The β angle is one of the $2p$ parameters $(\vec{\gamma}, \vec{\beta})$ forming the parameter vector that is optimized by the classical algorithm.

Vanilla QAOA: Circuit Construction

- **Initialization:** Apply Hadamards (H) to all n qubits to create a uniform superposition ($|\psi(0)\rangle$).

Vanilla QAOA: Circuit Construction

- **Initialization:** Apply Hadamards (H) to all n qubits to create a uniform superposition ($|\psi(0)\rangle$).
- **Ansatz Layering (p-depth):** The core quantum circuit repeats an alternating sequence of parameterized operators p times:

$$|\psi(\vec{\beta}, \vec{\gamma})\rangle = \prod_{k=1}^{p} \left(e^{-i\gamma_k \mathbf{H}_C} e^{-i\beta_k \mathbf{H}_M} \right) |\psi(0)\rangle$$

Vanilla QAOA: Circuit Construction

- **Initialization:** Apply Hadamards (H) to all n qubits to create a uniform superposition ($|\psi(0)\rangle$).
- **Ansatz Layering (p-depth):** The core quantum circuit repeats an alternating sequence of parameterized operators p times:

$$|\psi(\vec{\beta}, \vec{\gamma})\rangle = \prod_{k=1}^{p} \left(e^{-i\gamma_k \mathbf{H}_C} e^{-i\beta_k \mathbf{H}_M} \right) |\psi(0)\rangle$$

- **Hybrid Core:** This sequence is the **ansatz**, parameterized by $2p$ angles, $\vec{\beta}$ (Mixer) and $\vec{\gamma}$ (Cost).

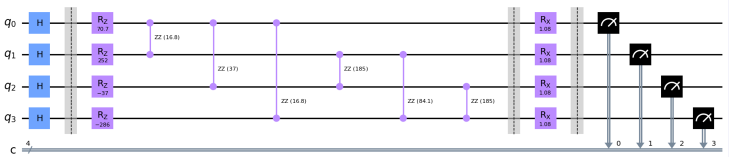
Vanilla QAOA: Circuit Construction

- **Initialization:** Apply Hadamards (H) to all n qubits to create a uniform superposition ($|\psi(0)\rangle$).
- **Ansatz Layering (p-depth):** The core quantum circuit repeats an alternating sequence of parameterized operators p times:

$$|\psi(\vec{\beta},\vec{\gamma})\rangle = \prod_{k=1}^{p} \left(e^{-i\gamma_k \mathbf{H}_C} e^{-i\beta_k \mathbf{H}_M} \right) |\psi(0)\rangle$$

- **Hybrid Core:** This sequence is the **ansatz**, parameterized by $2p$ angles, $\vec{\beta}$ (Mixer) and $\vec{\gamma}$ (Cost).

Qiskit Circuit (Example for $p = 1$)



Vanilla QAOA: Hybrid Optimization Workflow

- **Quantum Step:** Measure final state (e.g., 1000 shots) to estimate expected cost $\langle \psi | \mathbf{H}_C | \psi \rangle$.

Vanilla QAOA: Hybrid Optimization Workflow

- **Quantum Step:** Measure final state (e.g., 1000 shots) to estimate expected cost $\langle\psi|\mathbf{H}_C|\psi\rangle$.
- **Classical Step:** Optimizer (e.g., COBYLA) updates parameters $(\vec{\gamma}, \vec{\beta})$ to minimize cost.

Vanilla QAOA: Hybrid Optimization Workflow

- **Quantum Step:** Measure final state (e.g., 1000 shots) to estimate expected cost $\langle \psi | \mathbf{H}_C | \psi \rangle$.
- **Classical Step:** Optimizer (e.g., COBYLA) updates parameters $(\vec{\gamma}, \vec{\beta})$ to minimize cost.
- **Result:** Final sampling yields bitstring probabilities.
- **Solution:** Most frequent bitstring $\rightarrow$ approximate optimum.

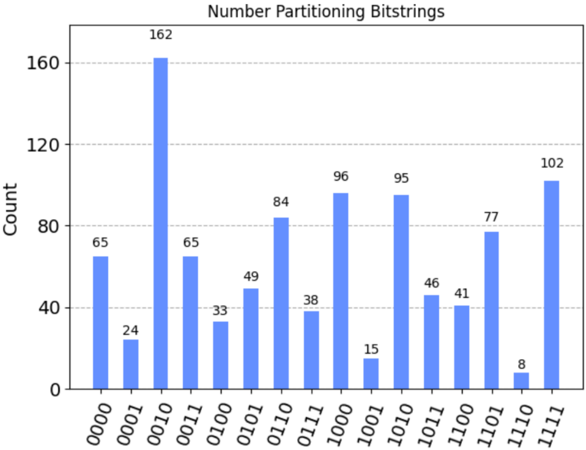
Vanilla QAOA: Hybrid Optimization Workflow

- **Quantum Step:** Measure final state (e.g., 1000 shots) to estimate expected cost $\langle\psi|\mathbf{H}_C|\psi\rangle$.
- **Classical Step:** Optimizer (e.g., COBYLA) updates parameters $(\vec{\gamma}, \vec{\beta})$ to minimize cost.
- **Result:** Final sampling yields bitstring probabilities.
- **Solution:** Most frequent bitstring $\rightarrow$ approximate optimum.

Workflow Summary

QAOA alternates between quantum measurements (to evaluate cost) and classical optimization (to improve parameters).

Vanilla QAOA: Result Example



Interpreting the Output

Most probable bitstring (e.g., 0010) $\Rightarrow$ Partition $\{1, 5, 5\}$ vs. $\{11\}$ in the number partitioning problem.

Quantum Annealing for Cancer Genomics

- **Platform:** Uses Quantum Annealers (e.g., D-Wave), which provide thousands of qubits to tackle larger, real-world QUBO instances.

Quantum Annealing for Cancer Genomics

- **Platform:** Uses Quantum Annealers (e.g., D-Wave), which provide thousands of qubits to tackle larger, real-world QUBO instances.
- **Application:** The Cancer Genomics pathway identification problem.

Quantum Annealing for Cancer Genomics

- **Platform:** Uses Quantum Annealers (e.g., D-Wave), which provide thousands of qubits to tackle larger, real-world QUBO instances.
- **Application:** The Cancer Genomics pathway identification problem.
- **QUBO Objective Recap:** The goal is to minimize exclusivity ($\mathbf{A}$) while maximizing coverage ($\mathbf{D}$):

$$\min_{\mathbf{x}} \left[\mathbf{x}^T \mathbf{A} \mathbf{x} - \alpha \mathbf{x}^T \mathbf{D} \mathbf{x} \right]$$

Quantum Annealing for Cancer Genomics

- **Platform:** Uses Quantum Annealers (e.g., D-Wave), which provide thousands of qubits to tackle larger, real-world QUBO instances.
- **Application:** The Cancer Genomics pathway identification problem.
- **QUBO Objective Recap:** The goal is to minimize exclusivity (**A**) while maximizing coverage (**D**):

$$\min_{\mathbf{x}} \left[\mathbf{x}^T \mathbf{A} \mathbf{x} - \alpha \mathbf{x}^T \mathbf{D} \mathbf{x} \right]$$

- **Data Preprocessing:** Requires significant effort to construct the QUBO coefficients from raw biological data (e.g., patient mutation lists from TCGA).

Quantum Annealing for Cancer Genomics

- **Platform:** Uses Quantum Annealers (e.g., D-Wave), which provide thousands of qubits to tackle larger, real-world QUBO instances.
- **Application:** The Cancer Genomics pathway identification problem.
- **QUBO Objective Recap:** The goal is to minimize exclusivity (**A**) while maximizing coverage (**D**):

$$\min_{\mathbf{x}} \left[\mathbf{x}^T \mathbf{A} \mathbf{x} - \alpha \mathbf{x}^T \mathbf{D} \mathbf{x} \right]$$

- **Data Preprocessing:** Requires significant effort to construct the QUBO coefficients from raw biological data (e.g., patient mutation lists from TCGA).

Input Data

Mutation data is sourced from databases like cBioPortal (TCGA AML study) to establish a Patient-Gene dictionary.

QA Preprocessing: Constructing **D** and **A**

- **1. Degree Matrix (D):** Defines the linear terms ($\mathbf{x}^T \mathbf{D} \mathbf{x}$).
  - **Role:** Measures Coverage (gene prevalence across patients).
  - **Construction:** **D** is diagonal; D_{ii} equals the number of patients affected by gene i.

QA Preprocessing: Constructing **D** and **A**

- **1. Degree Matrix (D):** Defines the linear terms ($\mathbf{x}^T\mathbf{D}\mathbf{x}$).
  - **Role:** Measures Coverage (gene prevalence across patients).
  - **Construction:** **D** is diagonal; D_{ii} equals the number of patients affected by gene i.
- **2. Adjacency Matrix (A):** Defines the quadratic terms ($\mathbf{x}^T\mathbf{A}\mathbf{x}$).
  - **Role:** Measures Exclusivity (gene-pair co-occurrence).
  - **Construction:** A_{ij} is the number of patients mutated by both gene i and gene j. Requires iterating over all gene pairs for each patient.

QA Preprocessing: Constructing **D** and **A**

- **1. Degree Matrix (D):** Defines the linear terms ($\mathbf{x}^T\mathbf{D}\mathbf{x}$).
  - **Role:** Measures Coverage (gene prevalence across patients).
  - **Construction:** **D** is diagonal; D_{ii} equals the number of patients affected by gene i.
- **2. Adjacency Matrix (A):** Defines the quadratic terms ($\mathbf{x}^T\mathbf{A}\mathbf{x}$).
  - **Role:** Measures Exclusivity (gene-pair co-occurrence).
  - **Construction:** A_{ij} is the number of patients mutated by both gene i and gene j. Requires iterating over all gene pairs for each patient.

```
Patient-Gene Dictionary:
TCGA-AB-2802
['IDH1', 'PTPN11', 'NPM1', 'MT-ND5', 'DNMT3A']
TCGA-AB-2804
['PHF6']
TCGA-AB-2805
['IDH2', 'RUNX1']
TCGA-AB-2806
['KDM6A', 'PLCE1', 'CROCC']
```

Figure 15: Sample of Patient-Gene Dictionary (Mapping patients to mutated gene lists)

QA Workflow: BQM and Embedding

- **BQM Construction:** The **A** and **D** matrices are compiled into the Binary Quadratic Model (BQM), which is the input format for the D-Wave system.

$$\mathbf{H} = \sum_{i,j} A_{ij} x_i x_j - \alpha \sum_{i} D_{ii} x_i$$

QA Workflow: BQM and Embedding

- **BQM Construction:** The **A** and **D** matrices are compiled into the Binary Quadratic Model (BQM), which is the input format for the D-Wave system.

$$\mathbf{H} = \sum_{i,j} A_{ij} x_i x_j - \alpha \sum_i D_{ii} x_i$$

- **Mapping Components:**
  - Linear terms ($-\alpha D_{ii}$) become **biases** on physical qubits.
  - Quadratic terms (A_{ij}) become **weights** on physical couplers.

QA Workflow: BQM and Embedding

- **BQM Construction:** The **A** and **D** matrices are compiled into the Binary Quadratic Model (BQM), which is the input format for the D-Wave system.

$$\mathbf{H} = \sum_{i,j} A_{ij} x_i x_j - \alpha \sum_{i} D_{ii} x_i$$

- **Mapping Components:**
  - Linear terms $(-\alpha D_{ii})$ become **biases** on physical qubits.
  - Quadratic terms (A_{ij}) become **weights** on physical couplers.
- **Embedding (Minor Embedding):** This is the crucial step where the abstract BQM graph is mapped onto the fixed physical topology of the Quantum Processing Unit (QPU).
  - D-Wave's `EmbeddingComposite` often handles this automatic placement and chaining of logical variables onto physical qubits.

QA Execution and Solution

1. **Sampling:** Submit BQM to D-Wave with multiple reads.
2. **Annealing:** System evolves toward the ground state.
3. **Results:** Returns bitstrings with associated energies.
4. **Selection:** Choose lowest-energy bitstring as optimal pathway.
5. **Mapping:** Convert binary solution to gene IDs.
6. **Validation:** Analyze pathway properties (e.g., coverage, exclusivity).

```
['ASXL1', 'BRINP3', 'DNMT3A']
coverage: 61.0
coverage/gene: 20.33
indep: 4.0
measure: 5.08
```

Example of a Discovered Cancer Gene Pathway

Conclusion: Synthesis of Problems and Solvers

- **QUBO as Interface:** The **Quadratic Unconstrained Binary Optimization (QUBO)** model serves as the universal language for expressing diverse NP-hard problems.

Conclusion: Synthesis of Problems and Solvers

- **QUBO as Interface:** The **Quadratic Unconstrained Binary Optimization (QUBO)** model serves as the universal language for expressing diverse NP-hard problems.
- **Scope:** We demonstrated QUBO formulation for both canonical (e.g., Number Partitioning) and practical (e.g., Cancer Genomics) problems.

Conclusion: Synthesis of Problems and Solvers

- **QUBO as Interface:** The **Quadratic Unconstrained Binary Optimization (QUBO)** model serves as the universal language for expressing diverse NP-hard problems.
- **Scope:** We demonstrated QUBO formulation for both canonical (e.g., Number Partitioning) and practical (e.g., Cancer Genomics) problems.
- **Algorithmic Synthesis:** QUBO links classical and quantum solvers by acting as the common input format:

Algorithm	**Platform**	**Mechanism**
Simulated Annealing (SA)	Classical	Thermal Fluctuation
Quantum Annealing (QA)	Quantum Hardware	Quantum Tunneling
QAOA	Hybrid/Gate Model	Parameterized Ansatz

Table: QUBO Solver Comparison

Future Directions: Bridging the Gap

The field needs focused research to move quantum algorithms toward practical advantage:

Future Directions: Bridging the Gap

The field needs focused research to move quantum algorithms toward practical advantage:

- **Scaling & Decomposition:** Developing techniques (e.g., circuit cutting) to partition large problems for limited NISQ hardware.

Future Directions: Bridging the Gap

The field needs focused research to move quantum algorithms toward practical advantage:

- **Scaling & Decomposition:** Developing techniques (e.g., circuit cutting) to partition large problems for limited NISQ hardware.
- **Error Mitigation:** Creating robust strategies to counteract high noise and decoherence in current quantum processors.

Future Directions: Bridging the Gap

The field needs focused research to move quantum algorithms toward practical advantage:

- **Scaling & Decomposition:** Developing techniques (e.g., circuit cutting) to partition large problems for limited NISQ hardware.
- **Error Mitigation:** Creating robust strategies to counteract high noise and decoherence in current quantum processors.
- **Hardware Optimization:** Tailoring circuits to specific device architectures for enhanced performance and reduced error rates.

Future Directions: Bridging the Gap

The field needs focused research to move quantum algorithms toward practical advantage:

- **Scaling & Decomposition:** Developing techniques (e.g., circuit cutting) to partition large problems for limited NISQ hardware.
- **Error Mitigation:** Creating robust strategies to counteract high noise and decoherence in current quantum processors.
- **Hardware Optimization:** Tailoring circuits to specific device architectures for enhanced performance and reduced error rates.
- **Algorithm Refinement:** Further scaling and refining hybrid methods (QAOA) and quantum machine learning models.

Future Directions: Bridging the Gap

The field needs focused research to move quantum algorithms toward practical advantage:

- **Scaling & Decomposition:** Developing techniques (e.g., circuit cutting) to partition large problems for limited NISQ hardware.
- **Error Mitigation:** Creating robust strategies to counteract high noise and decoherence in current quantum processors.
- **Hardware Optimization:** Tailoring circuits to specific device architectures for enhanced performance and reduced error rates.
- **Algorithm Refinement:** Further scaling and refining hybrid methods (QAOA) and quantum machine learning models.
- **Near-Term Solutions:** Continued development of **Quantum-Inspired** classical methods while fault-tolerant hardware matures.