QUBO and Ising Models

Translate a constrained binary objective into QUBO and Ising form without losing its meaning.

Undergraduate 22 min lesson Dr. Charith Jayasekara

By the end, you'll be able to

- Expand a binary penalty using $x^2 = x$.
- Convert binary variables to the stated spin convention.
- Verify energies on every small-instance assignment.

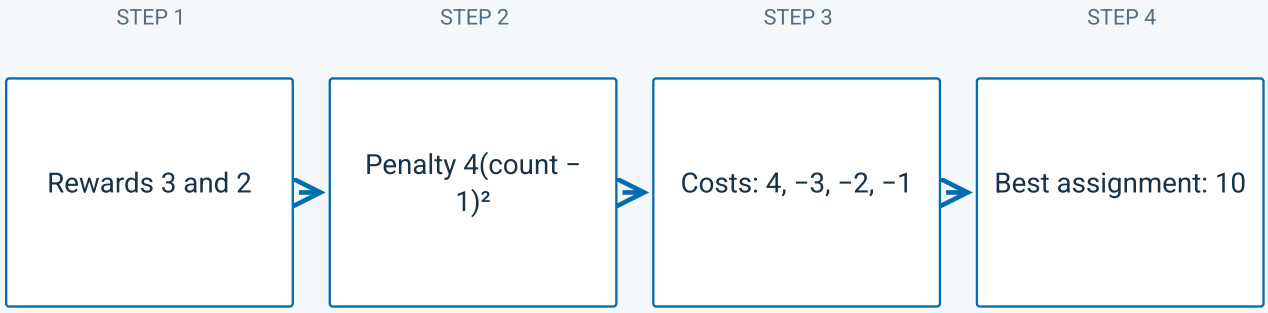
Two languages for the same binary problem

A QUBO expresses a quadratic objective in zero-or-one variables. An Ising model uses variables taking values positive or negative one. Converting between them is ordinary algebra. Quantum optimisation methods often use the Ising form because the Pauli Z operator has those two eigenvalues, while application software often starts with binary decisions.

Prerequisites: tensor-product ordering, Pauli gates and the preceding optimisation lesson. Read the first example as a table if algebra is unfamiliar, then verify the expansion term by term.

WORKED DIAGRAM · A17

Choose one item



In order 00, 10, 01, 11, the costs are 4, −3, −2 and −1. Spin values use $x = (1 - z)/2$.

State the objective convention

Write a QUBO as a constant plus linear terms and one coefficient for each unordered pair. This avoids ambiguity about whether off-diagonal entries in a matrix expression are counted twice. If you use the matrix notation x-transpose Q x, state whether Q is symmetric or upper triangular. Two files with identical-looking numbers can represent different objectives under different conventions.

For the Ising conversion used here, binary value x corresponds to spin z through x equal to one minus z, divided by two. Thus binary zero maps to spin positive one, matching the Z eigenvalue of the zero basis state. Binary one maps to negative one. Another sign convention is valid if it is applied consistently to coefficients and decoding.

Convert a binary decision into the corresponding spin value before substituting into the objective.

$$z_i = 1 - 2x_i \in \{+1, -1\}, \qquad x_i = \frac{1 - z_i}{2}, \qquad \widehat{x}_i = \frac{I - Z_i}{2}.$$

Binary zero corresponds to spin $+1$ and binary one to spin -1. The operator $\widehat{x}_i$ has those binary values on computational-basis states, so the worked scalar cost becomes a diagonal Hamiltonian with the same table of costs.

Worked example: choose exactly one item

There are two items with rewards three and two. Choose exactly one. Minimise the negative reward plus a penalty of four times the square of the selection-count error:

$$C(x_0, x_1) = -3x_0 - 2x_1 + 4(x_0 + x_1 - 1)^2.$$

Because a binary variable satisfies its square equal to itself, expansion gives a constant four, linear coefficients negative seven and negative six, and pair coefficient eight. Evaluate all four candidates. The costs of 00, 10, 01 and 11 are respectively four, negative three, negative two and negative one. The best candidate is 10, meaning choose the higher-reward item.

Now substitute each binary variable with its spin expression. The Ising form is

$$C = -\frac{1}{2} + \frac{3}{2}z_0 + z_1 + 2z_0z_1.$$

Check 10: its spin values are negative one and positive one, giving negative one half minus one and a half plus one minus two, which is negative three. Check 00: both spins are positive one, giving four. A complete four-row comparison is an inexpensive way to catch coefficient or sign errors.

Turn the scalar model into an operator

Replace each spin by the corresponding Z operator and each spin product by a tensor product of Z operators on those wires. The result is diagonal in the computational basis. Every bitstring is an eigenstate, and its eigenvalue equals the classical cost. The operator’s ground-state labels therefore encode the QUBO minimisers, provided the penalty model correctly represents the original constraints.

The constant shift changes all energies equally. It does not change the minimising bitstrings and contributes only a global phase in an isolated cost-evolution unitary. It can still matter when reporting absolute objective values or comparing against an application specification. Omitting a constant from a circuit does not authorise omitting it from a claimed numerical solution value.

Choose a penalty with a reason

In this two-item example, choosing both gains extra reward but violates the constraint. A penalty greater than two makes the both-chosen cost exceed the best feasible cost negative three. The chosen value four satisfies that requirement. A penalty of one would make 11 cost negative four and incorrectly reward an infeasible solution.

For larger problems, derive a bound from the maximum possible objective improvement gained by violating the constraint. Blindly setting an enormous penalty can exceed hardware coefficient ranges or suppress relevant objective

differences after rescaling. Penalty design is part of the model, and feasibility must be checked after decoding every returned sample.

Higher-order products are not QUBO terms. They may require auxiliary variables and additional penalties, increasing the model size and changing connectivity. Count those auxiliary variables when reporting qubit requirements. A compact application description can expand substantially before it becomes a hardware-compatible quadratic model.

A reproducible conversion procedure

Start with a classical evaluator for the original constrained problem. Expand the binary penalty using the binary-square identity. Collect constant, linear and pair coefficients. Apply the binary-to-spin substitution and collect again. Enumerate every assignment for a small test instance and compare original, QUBO and Ising energies. Finally map bit positions to wires explicitly before constructing a circuit.

When the model is too large for enumeration, keep small unit instances with known answers. They protect against convention changes in a solver interface. Independent verification should use the original application rules, not merely reevaluate the same potentially incorrect transformed coefficients.

Check the constant shift separately

Remove the constant negative one half from the worked Ising expression. Every reported energy increases by one half: the original costs four, negative three, negative two and negative one become four and a half, negative two and a half, negative one and a half, and negative one half. The minimising assignment is unchanged because all differences between candidates are preserved.

A phase-evolution circuit can ignore this constant when it only contributes a common phase. An application report cannot silently use the shifted energy as the original reward or penalty. Decode the bitstring and reevaluate the original objective to recover its meaningful value. This is especially important when comparing solvers that use different coefficient normalisations.

Multiplying the entire objective by a positive scale also preserves minimisers but changes phase angles and numerical precision requirements. Multiplying by a negative scale reverses minimisation and maximisation. Constants, positive scales and sign reversals therefore have different consequences, even though all can look like harmless algebraic simplifications.

Limits and primary context

This transformation preserves an objective; it does not solve it. QUBO models can be attacked by exact search, mixed-integer optimisation, local search, simulated annealing, quantum annealing or gate-based methods. An appropriate comparison uses the same decoded problem and target quality. Lucas provides primary Ising formulations for a collection of computational problems, while QAOA connects cost functions to phase-separation circuits. Lucas: Ising formulations, QAOA.

Exercises and worked answers

Exercise 1. Convert the one-variable cost negative three times x into the stated spin convention.

Check the worked answer

Substitution gives negative three halves plus three halves times z. The energies are zero for binary zero and negative three for binary one, so minimisation chooses one.

Exercise 2. In the worked example, use penalty one instead of four and evaluate 11. What failed?

Check the worked answer

The reward is negative five and the penalty is one, giving negative four. That beats the best feasible cost negative three, so the penalty does not enforce the intended constraint. A solver can minimise the transformed model perfectly while returning an invalid application answer.

Check your understanding

PAUSE & THINK

With rewards 3 and 2 and penalty 1 for choosing both, what is the cost of 11?

- ☐ −4
- ☐ −1
- ☐ 1

Answer: −4. The negative reward is −5 and the penalty is +1, giving −4. This incorrectly beats the best feasible cost −3.

Sources

- Lucas: Ising formulations
- QAOA