Quantum Optimization Modeler

nacedob

Consider the following binary optimization problem:

$$\begin{aligned} \underset{\boldsymbol{x}\in\{0,1\}^N}{\operatorname{argmin}} \quad & \boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{l}^T \boldsymbol{x}, \\ \text{s.t.} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & G\boldsymbol{x} \leq \boldsymbol{h} \end{aligned} \tag{1}$$

The aim of this repository is to translate such problems into a representation suitable for quantum hardware. In particular, we target the Ising Hamiltonian form, which is commonly used in quantum optimization. Additionally, we provide a transformation into the Quadratic Unconstrained Binary Optimization (QUBO) format, as many classical and quantum solvers are designed to work directly with this formulation.

1 From Quadratically Constrained Binary Optimization to QUBO

Unifying the Cost Function

Let $\mathcal{D}$ denote the operator that maps a vector $v \in \mathbb{R}^d$ to a diagonal matrix $V \in \mathbb{R}^{d\times d}$ whose diagonal entries are precisely the elements of v.

Since each variable x_i is binary, it satisfies $x_i^2 = x_i$. Therefore, the linear term can be rewritten using this identity: $l_i x_i = x_i l_i x_i$. This implies that the linear part can be expressed as a quadratic form:

$$\boldsymbol{l}^T \boldsymbol{x} = \boldsymbol{x}^T \mathcal{D}(\boldsymbol{l}) \boldsymbol{x}$$

Substituting this into the original objective function, problem (1) becomes:

$$\begin{aligned} \underset{\boldsymbol{x}\in\{0,1\}^N}{\operatorname{argmin}} \quad & \boldsymbol{x}^T (Q + \mathcal{D}(\boldsymbol{l})) \boldsymbol{x}, \\ \text{s.t.} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & G\boldsymbol{x} \leq \boldsymbol{h} \end{aligned} \tag{2}$$

Integer variables

To represent an integer variable $z \in [a, b]$ using exactly n binary variables, it is necessary that:

$$2^n - 1 \geq b - a.$$

We define a shifted version of the variable as $z' = z - a$, which ensures $z' \in [0, b - a]$. This can be encoded in binary as:

$$z' = \sum_{i=1}^{n} x_i \cdot 2^{i-1}, \quad x_i \in \{0, 1\}.$$

From this encoding, the original variable z is recovered as:

$$z = \sum_{i=1}^{n} x_i \cdot 2^{i-1} + a.$$

The constraint $0 \le z' \le b - a$ ensures that the domain of z is respected. This fixed-length binary encoding is commonly used in mixed-integer and combinatorial optimization problems.

Equality constraints

To transform the problem into a QUBO (Quadratic Unconstrained Binary Optimization) form, equality constraints must be incorporated as penalty terms in the objective function. This is done by adding the following penalty:

$$\lambda \sum_{i=1}^{m} \left(\sum_{j=1}^{n} A_{ij} x_j - b_i \right)^2 .$$

Expanding the squared terms and neglecting constants that do not affect the optimization, we obtain:

$$\lambda \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{n} A_{ij} A_{ik} x_j x_k - 2\lambda \sum_{i=1}^{m} b_i \sum_{j=1}^{n} A_{ij} x_j = \lambda x^T A^T A x - 2\lambda \boldsymbol{b}^T A x. \tag{3}$$

By choosing an appropriate value for the penalty parameter λ, the problem (2) becomes equivalent to:

$$\begin{aligned} \underset{\boldsymbol{x} \in \{0,1\}^N}{\operatorname{argmin}} \quad & \boldsymbol{x}^T Q_{\text{eq}} \boldsymbol{x}, \\ \text{s.t.} \quad & G\boldsymbol{x} \le \boldsymbol{h} \\ & Q_{\text{eq}} = Q + \lambda A^T A + \mathcal{D}(l - 2\lambda b^T A) \end{aligned} \tag{4}$$

Inequality constraints

Now consider an inequality constraint:

$$Gx \le h,$$

To reformulate this as an equality, we introduce *slack variables* $s \in \mathbb{R}^M$ such that:

$$Gx + s = h, \quad s \ge 0.$$

Each slack variable s_i is expressed as a sum of binary variables:

$$s_i = \sum_{j=0}^{K_i - 1} 2^j s_{ij}, \quad s_{ij} \in \{0, 1\}.$$

where:

- s_{ij} are binary variables encoding the value of s_i,
- K_i is the number of bits used for the encoding of slack s_i.

After introducing the slack variables, the inequality constraint becomes an equality, and the formulation in (3) can be applied analogously.

To express this compactly, define $y = (x, s) \in \mathbb{R}^{N+\sum_i K_i}$ and consider the problem:

$$
\begin{aligned}
&\underset{y_i \in \{0,1\}}{\operatorname{argmin}}\ y^T \tilde{Q} y \\
&\quad \tilde{G} y = h \\
&\text{where: } \tilde{Q} = \begin{pmatrix} Q + \lambda A^T A + \mathcal{D}(l - 2\lambda b^T A) & \mathbf{0}_{N \times \sum_i K_i} \\ \mathbf{0}_{\sum_i K_i \times N} & \mathbf{0}_{\sum_i K_i \times \sum_i K_i} \end{pmatrix} \\
&\quad \tilde{G} = \begin{pmatrix} G_{0:} & 2_{K_0} & 0_{K_1} & \dots & 0_{K_{M-1}} \\ G_{1:} & 0_{K_0} & 2_{K_1} & \dots & 0_{K_{M-1}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ G_{n:} & 0_{K_0} & 0_{K_1} & \dots & 2_{K_{n-1}} \end{pmatrix} \\
&\quad 2_m = \begin{pmatrix} 2^0 & 2^1 & \dots & 2^{m-1} \end{pmatrix} \in \mathbb{R}^m \\
&\quad 0_m = \begin{pmatrix} 0 & 0 & \dots & 0 \end{pmatrix} \in \mathbb{R}^m \\
&\quad \mathbf{0}_{m \times n} = \begin{pmatrix} 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix} \in \mathbb{R}^{m \times n} \\
&\quad G_{m:} \in \mathbb{R}^{1 \times N} \text{ is the } m^{\text{th}} \text{ row of } G
\end{aligned} \tag{5}
$$

By incorporating the equality constraint $\tilde{G}y = h$ as a penalty, following the method used in (4), we arrive at the final QUBO formulation:

$$
\begin{aligned}
\underset{\boldsymbol{y} \in \{0,1\}^{N+\sum_i K_i}}{\operatorname{argmin}} \quad & \boldsymbol{y}^T \tilde{Q}_{\text{eq}} \boldsymbol{y}, \\
& \tilde{Q}_{\text{eq}} = \tilde{Q} + \lambda(\tilde{G}^T \tilde{G} - 2\mathcal{D}(h^T \tilde{G}))
\end{aligned} \tag{6}
$$

The optimal solution to the original problem (1) can be recovered by selecting the first N components of the solution vector y.

2 From QUBO to Ising model

To convert the QUBO into an Ising formulation, each binary variable $x_i \in \{0, 1\}$ is mapped to an Ising spin variable $z_i \in \{-1, 1\}$ via:

$$x_i = \frac{1 - z_i}{2}.$$

This mapping ensures:

$$x_i = 0 \iff z_i = 1, \quad x_i = 1 \iff z_i = -1.$$

By substituting this transformation into the final QUBO form, one obtains an Ising objective. As described in [1], this can be expressed as:

$$\underset{\boldsymbol{x} \in \{0,1\}^N}{\operatorname{argmin}} \quad \boldsymbol{x}^T \boldsymbol{J} \boldsymbol{x} + \boldsymbol{h}^T \boldsymbol{x},$$

where:

$$
\begin{aligned}
J &= \tilde{Q}_{\text{eq}}, \\
h &= -(J + J^T) \cdot \vec{1}_N \qquad \left(h_i = -\left(\sum_j J_{ij} + \sum_j J_{ji} \right) \right).
\end{aligned}
$$

References

[1] IBM Quantum Learning. *Solve utility-scale quantum optimization problems*. https://learning.quantum.ibm.com/tutorial/quantum-approximate-optimization-algorithm. Accedido: 3 de febrero de 2025.