Solving quadratic optimization problems with bosonic quantum computer simulator

Report I

Bálint Hantos
supervisor: Péter Rakyta

Motivation

The field of quantum computing is a promising subject because of the powerful applications in solving hard optimization problems.

Quadratic uncostrained binary optimization (QUBO)[4]

In many applications we face optimization problems, where one has to fit parameters of a given model. Mostly, those problems are about finding continous parameters, for which there are powerful tools eg. gradient descent algorithm. However, this requires convexity which means the
$f : \mathbb{R}^m \to \mathbb{R}$ objective function to be twice continously differentiable.

In our case, what we're trying to do is to find an optimal set of parameters from a finite set of discrete values. This implies that the domain of the objective function is not continous, therefore convexity is not satisfied. In most of the problems brute-force search is not feasible due to the large number of potential variations. Instead, there is a large amount of literature on polynomial-time algorithms for certain classes of such problems, which either approximate the optimal solution or rule out a part of the search space.

However there are still a number of combinatorial optimization problems, which have no determenistic polinomial-time algorithm yet. One of these problems is *quadratic unconstrained binary optimization* (QUBO).

Finding the optimal solution to a QUBO is equivalent to minimizing the H Hamilton operator of the Ising-model. It has a wide range of applications apart from physics, for example in binary portfolio optimization, or various other problems in finance and mathematics.

We can formulate it as finding the minimum of a quadratic binary mapping $f_Q(x)$ of a fixed $m \times m$ symmetric Q matrix $f_Q(x) : \{0,1\}^M \to \mathbb{R}$ in the form of

$$f_Q(x) = \frac{1}{2}\langle\, x|\, Q\, |x\,\rangle = \frac{1}{2}\sum_{i,j=1}^{M} Q_{ij}x_i x_j. \tag{1}$$

The goal is to find $x^* = \text{argmin}(f(x))$ for the given $Q\,\epsilon\,\mathbb{R}^{m\times m}$ symmetric matrix.

If there is a linear term it can be incorporated into the matrix $\tilde{Q}$:

$$\begin{aligned} f_Q(x) &= \sum_{i,j=1}^{M} Q_{ij}x_i x_j + \sum_{i=1}^{M} b_i x_i = \\ &= \sum_{i,j=1}^{M} Q_{ij}x_i x_j + \sum_{i=1}^{M} b_i x_i x_i = \\ &= \sum_{i=1}^{M}(Q_{ii} + b_i)x_i x_i + \sum_{\substack{i,j=1 \\ i\neq j}}^{M} Q_{i,j}x_i x_j = \\ &= \sum_{i=1}^{M} \tilde{Q}_{ii}x_i x_i + \sum_{\substack{i,j=1 \\ i\neq j}}^{M} Q_{i,j}x_i x_j = \\ &= \sum_{i,j=1}^{M} \tilde{Q}_{ij}x_i x_j, \end{aligned} \tag{2}$$

where Q_{ii} is the m-dimensional vector made up from the main diagonal of Q. This result can be reached because it makes no difference if b_i is multiplied by x_i or $x_i x_i$, since $x_i\,\epsilon\,\{0,1\}$. So, the Q matrix is transformed in the way

$$\tilde{Q}_{ij} = \begin{cases} Q_{ii} + b_i & \text{if} \quad i = j \\ Q_{ij} & \text{if} \quad i \neq j. \end{cases} \tag{3}$$

In the worst case we can find the optimal $|x\rangle$ of a QUBO by brute-force search. For a $m \times m$ Q matrix, the size of the binary vector $|x\rangle$ is m. So the number of potential variations for $|x\rangle$ is 2^m.

Introduction and theoretical background

Boson sampling[1]

Boson sampling shows an example of a classically computationally expensive mathematical problem, which can be efficiently solved by a realizable physical experimental setup. The boson sampler consists of a linear optics network and a coincidence photodetector. The linear optic part consists of beam splitters and phase shifter, characterized by angles ϑ_i and ψ_i for each piece of equipment accordingly.

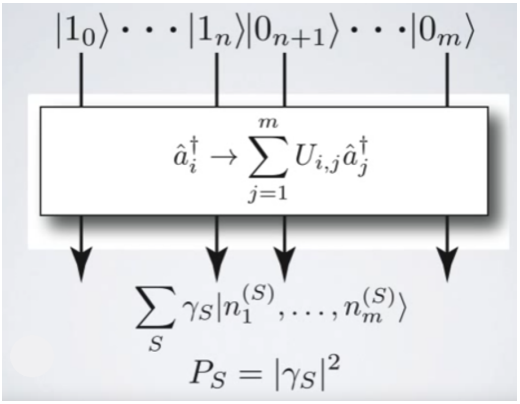

Figure 1: Schema of a boson sampler.
On top, the single photon states and vacuum states, propagated through a linear optics network U. At the bottom the superpositions of the multiphoton optical states. γ_S is the amplitude of the configuration S (instead of S we use $|\mathbf{n}\rangle$ in this document), P_S is the probability of measuring the configuration S, which we try to reconstruct with repeated photon coincidence measurements. Image credits to [1].

The boson sampler is an M-mode linear interferometer, where the inputs are single photons, and the outputs are multi photon states. The input photons are indistinguishable from each other, so we are not concerned about which input photon end up in which output state, we consider only the output configuration

$$|\xi(\theta,\psi)\rangle = \sum_{|\mathbf{n}\rangle} \alpha_{|\mathbf{n}\rangle} |n_1, n_2, ...n_M\rangle. \tag{4}$$

This output superposition of multiple states is measured so $|\xi\rangle$ collapses to a single $|\mathbf{n}\rangle = |n_1, n_2, ..., n_M\rangle$ multiphoton Fock-vector with probability $\alpha^2_{|\mathbf{n}\rangle}$, where n_i are the number of photons in the i state.

Simulating such an equipment on a classical computer, that samples bosons in this way is pretty hard because the probabilities (and amplitudes) associated with the optical states involves calculating the permanent of a complex-valued matrix, which is #P-complete [2]:

$$P(|\mathbf{n}\rangle) = \frac{|\mathrm{Per}(A_{|\mathbf{n}\rangle})|^2}{n_1! \cdot n_2! \cdot ... \cdot n_M!}, \tag{5}$$

where $A_{|\mathbf{n}\rangle}$ is a unitary matrix describing both the unitary matrices of the linear optics network configuration and the $|\mathbf{n}\rangle$ output configuration. The n_i in the denominator are the number of photons in state i for the given configuration $|\mathbf{n}\rangle$.

Variational bosonic solver[3]

Most problems in physics and other fields of study have no exact solutions. The variational approach is a way of obtaining a numerical solution to such problems. It approaches the exact solution (if there is any) as close as it is needed. We generally think of energy as an objective function, and the goal is to minimize it by finding the optimal values for the free parameters.

In order to interpret the measurement in a qubit basis the Fock-vector is mapped onto a M component $|\mathbf{b^{(j)}}\rangle$ binary vector with the $\wp_j$ parity function:

$$\wp_j : b_i^{(j)} = (n_i \bmod 2) \oplus j. \tag{6}$$

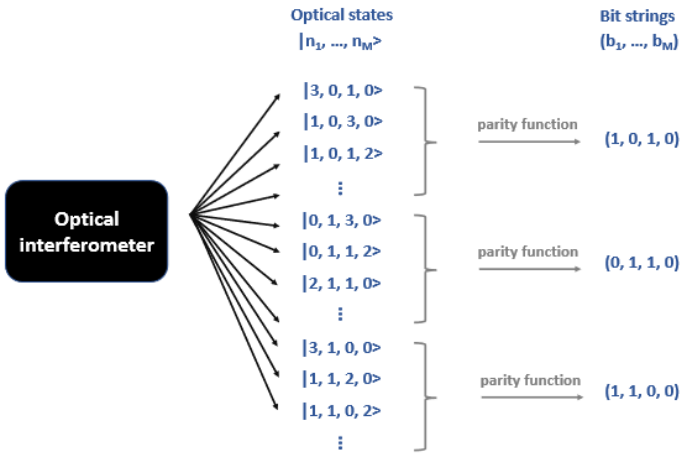

Figure 2: Examples of mapping multiple optical states the same bitstring. Image credits to [3].

Note that the different optical states can be mapped to the same bit strings, in that case their probabilities are summed. So if a $|\mathbf{n}\rangle$ and $|\mathbf{n}'\rangle$ with amplitudes α and α' are mapped to the same binary vector $|\mathbf{b^{(j)}}\rangle$ then the associated probability is $\beta = \alpha^2 + \alpha'^2$.

The $|\mathbf{b^{(j)}}\rangle$ bitstring has a corresponding amplitude $\beta_{|\mathbf{b^{(j)}}\rangle}$. Then the expected value of the E energy is calculated in the following way:

$$E_{|\mathbf{b^{(j)}}\rangle}(j; \vartheta, \psi) = \sum_{|\mathbf{b^{(j)}}\rangle} \beta_{|\mathbf{b^{(j)}}\rangle} \langle \mathbf{b^{(j)}}| \, H \, |\mathbf{b^{(j)}}\rangle. \tag{7}$$

The variational bosonic solver algorithm tries to find the E_{min} minimal energy and the corresponding configuration $|\mathbf{b}_{min}\rangle$. In each epoch the gradient descent step is made in the direction to the minimal energy value by adjusting ϑ_i and ψ_i angles of the beam splitters and phase-shifters.

Setup

Technical details

We're going to use the Picasso and Picasso Boost libraries as a bosonic quantum simulator, which are written in python and C++. In order to install them we needed to install a couple of dependencies on a Linux machine. For this task I decided to use a Docker Container built from an Ubuntu image, upon which I installed the Miniconda distribution of Anaconda to spare some disk space. With the help of my supervisor we installed the necessary packages and built the Piquasso Boost library. Then we launched a prewritten test of the package which turned out well, thus the installation was succesful.

Examining the code

After that I checked how the variational QUBO-solver worked with some examples. I used a general 3-by-3 symmetric matrix to calculate its expected values with all possible binary binary vectors

$$\mathbf{Q} = \begin{pmatrix} A & B & C \\ B & D & E \\ C & E & F \end{pmatrix}. \tag{8}$$

(1,0,0)	A
(1,1,0)	A + 2B + D
(1,1,1)	A + D + F + 2(B+ C + E)
(1,0,1)	A+2C+F
(0,1,1)	D+2E+F
(0,0,1)	F
(0,1,0)	D
(0,0,0)	0

Table 1: Expected value of $\mathbf{Q}$ with the possible binary vectors.

I found that the implementation of the variational algorithm works as intended. Although I found some corner cases, for example when the minimal expected values are degenerate then the optimal binary vector is ambiguous. In this case, the solver returns only one answer after multiple runs.

References

[1] Gard, Bryan T., et al. "An introduction to boson-sampling." *From atomic to mesoscale: The role of quantum coherence in systems of various complexities.* 2015

[2] Aaronson, Scott, and Alex Arkhipov. "The computational complexity of linear optics." Proceedings of the forty-third annual ACM symposium on Theory of computing. 2011. `https://arxiv.org/abs/1011.3245`

[3] Bradler, Kamil, and Hugo Wallner. "Certain properties and applications of shallow bosonic circuits." arXiv preprint arXiv:2112.09766 (2021). `https://arxiv.org/abs/2112.09766`

[4] Quadratic unconstrained binary optimization - Wikipedia, 2022 [Accessed 18 February 2022]