Wind farm layout optimization using quantum computing

Gopal Ramesh Dahale

dahalegopal27@gmail.com,

Link to GitHub Repository

1 Introduction

Wind farm layout optimization (WFLO) involves determining the optimal positions of wind turbines within a wind farm to maximize energy production while minimizing adverse environmental impacts and economic costs. To address the WFLO challenge, we propose a hybrid quantum-classical optimization approach. Next sections outlines the problem formulation based on [Sen+22] and algorithmic description.

2 Problem Description

- **Inputs**: The inputs consists of the grid size n where n is the number of cells. m being the number of turbines to place. Number of wind regimes to consider. Currently we consider either only one wind regime (0 degree, 12 kmph) or 36 wind regimes as shown in the figure below.

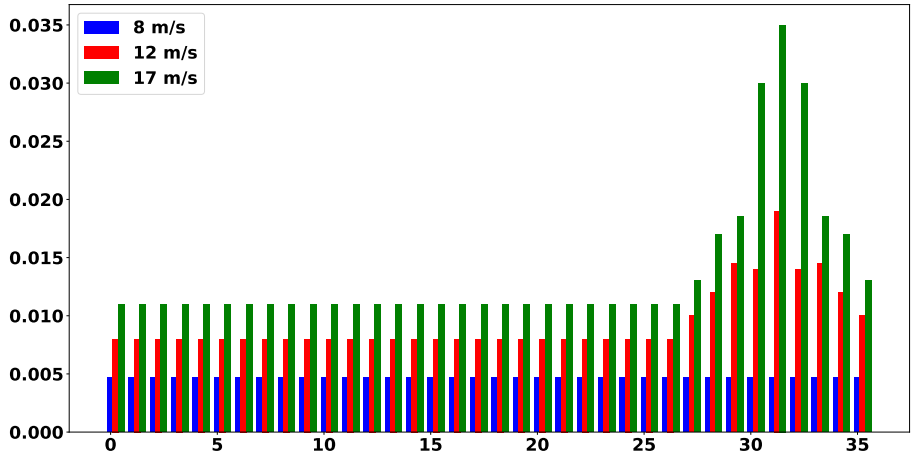

Figure 1: Wind regimes: x-axis is the angle (times 10 degrees), y-axis is the probability for wind regime, and color corresponds to free wind speed for the different wind directions [Sen+22]

- **Outputs**: The output consists of a binary 1d array (flattened 2d array representing the grid), where a 1 denotes placement of turbine. The output also show the final expected power of the LS model.

- **Decision variables** Here each cell in the grid represents a decision variable. A value of 1 means that the cell has a turbine. We have as many variables as the number of cells i.e. $n \times n$.

3 Problem Formulation

3.1 Representing wind farm as grid

The wind farm is represented as grid of size $n \times n$ where n is the number of square cells. Following the paper [Sen+22], the grid will be of size 2 km $\times$ 2 km and $n = 20$ which means that the size of each cell is 100 square meters. We also plan initially experiment with smaller grid sizes and later scale it.

3.2 Wakes

Wakes are considered as one constraint. The wind turbines influence each other in a farm. Wind passed from a upstream turbine change the airflow dynamics of a downstream turbine and this is known as wake interference.

Let $\mathcal{D}$ represent the set of wind regimes. The probability of a wind regime $d \in \mathcal{D}$ is denoted by p_d, where $\sum_{d\in\mathcal{D}} p_d = 1$.

Let u_{id} represent the wind velocity at turbine $i = 1, \ldots, m$ for wind regime $d \in \mathcal{D}$. Given the interference of the wakes, the velocity of turbine i is not equivalent to the free wind speed of d. Instead, u_{id} can be computed using the following equation:

$$u_{id} = u_{id\infty}\left[1 - \sqrt{\sum_{j\in\mathcal{U}_{id}}\left(1 - \frac{u_{ijd}}{u_{id\infty}}\right)^2}\right], \tag{1}$$

Here, $u_{id\infty}$ represents the free wind speed, $\mathcal{U}id$ corresponds to the set of turbines that are upstream to i under wind regime d, and $uijd$ is the wake-reduced speed at i due to upstream turbine j. The set of upstream turbines, $\mathcal{U}_{id}$ is computed by rotating the grid with respect to the current wind regime $d \in \mathcal{D}$ and identifying the turbines, j, located upstream of turbine i given d. The expression in Eq. (1) is referred to as the sum-of-squares (SS) model for the total expected power in the presence of wakes. We can now express the sum-of-squares (SS) expected power of a wind farm with m turbines as:

$$E_{SS} = \sum_{i=1}^{m}\sum_{d\in\mathcal{D}} \frac{1}{3} u_{id}^3 p_d. \tag{2}$$

Eq. (2) is considered the most accurate analytical total power expression that accounts for multiple wakes. However, this expression is challenging for exact mathematical optimization techniques due to its complexity, as it involves the cube of a square root.

We will consider the linear superposition equation given by:

$$E_{LS} = \sum_{i=1}^{m}\sum_{d\in\mathcal{D}}\left(\frac{1}{3}u_{id\infty}^3 - \sum_{j\in\mathcal{U}_{id}}\frac{1}{3}(u_{id\infty}^3 - u_{ijd}^3)\right). \tag{3}$$

The LS model can be formulated as a QUBO problem.

3.3 Proximity constraints

Turbines should be situated with a minimum separation of at least five rotor diameters. This constraint is incorporated this constraint into the optimization model depending on

the cell size. In below figure from [Sen+22], the shaded area surrounding the turbine signifies the restricted cells.

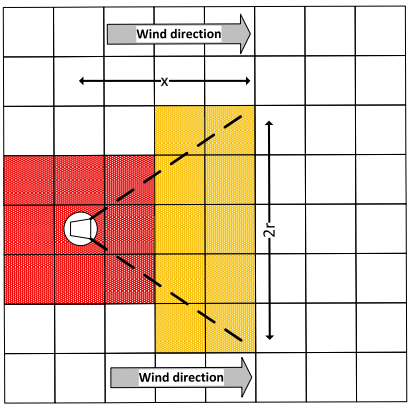

Figure 2: Wind farm as a grid where the red cells shows the proximity contrainsts and yellow cells shows the wake effect.

3.4 QUBO Formulation

Define binary decision variables, denoted as x_i, which indicate whether a turbine is located at position $i \in \mathcal{N}$. Here, $\mathcal{N}$ represents the set of $k = n^2$ possible turbine locations within the grid. To simplify notation, we use x to represent the vector $(x_1, \ldots, x_k)$.

Additionally, we establish a set $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$, consisting of pairs of locations where turbines cannot be simultaneously placed due to proximity constraints.

We introduce two wind speed variables, u_{id} and $u_{id\infty}$, corresponding to turbine location $i \in \mathcal{N}$ under wind regime $d \in \mathcal{D}$, with and without the influence of other turbines causing wake effects. It's important to note that in this context, we refer to i as a potential turbine location, as opposed to an indexed turbine. Furthermore, we define $\mathcal{U}_{id} \subseteq \mathcal{N}$ as the set of upstream turbine locations that influence a turbine placed at cell i for wind regime d. u_{ijd} represents the wind speed at location j due to a single wake generated by an upstream turbine at location i ($j \in \mathcal{U}_{id}$).

The total expected energy of a wind farm based on placement decisions x is expressed by the following function:

$$f(x) = \sum_{i \in \mathcal{N}} \sum_{d \in \mathcal{D}} p_d \left(\frac{1}{3}\, u_{id,\infty}^3 x_i - \sum_{j \in \mathcal{U}_{id}} \frac{1}{3} (u_{id,\infty}^3 - u_{ijd}^3) x_i x_j \right). \tag{4}$$

We need to maximize $f(x)$ such that

$$\sum_{i \in \mathcal{N}} x_i = m, \tag{5}$$

$$x_i + x_j \leq 1, \forall (i,j) \in \mathcal{E}, \tag{6}$$

$$x_i \in \{0,1\}, \forall i \in \mathcal{N}. \tag{7}$$

To convert this constrained quadratic problem into an unconstrained quadratic model, we introduce penalty terms for the constraints.

$$\max_x f(x) - \lambda_1 \left(\sum_{i \in \mathcal{N}} x_i - m \right)^2 - \lambda_2 \sum_{(i,j) \in \mathcal{E}} x_i x_j, \tag{8}$$

The values of the two penalty terms, denoted as λ_1 and λ_2, need to be sufficiently substantial to guarantee the feasibility of solutions with respect to the problem constraints.

4 Hybrid Quantum-Classical Algorithm

4.1 Small grid size

For an initial prototype, we start with smaller grid sizes which can fit on current Pasqal devices. This involves using the QAOA and/or QAA to solve the QUBO formulation [Pulb]. The QUBO problem is embedded onto atomic register followed by preparing the following ising hamiltonian H_Q

$$H_Q = \sum_{i=1}^{N} \frac{\hbar\Omega}{2}\sigma_i^x - \sum_{i=1}^{N} \frac{\hbar\delta}{2}\sigma_i^z + \sum_{j<i} \frac{C_6}{|\mathbf{r}_i - \mathbf{r}_j|^6} n_i n_j \tag{9}$$

4.2 Scaling grid size

For large grid size, which cannot fit on the current Pasqal devices, we use decomposition techniques as presented in [BRR17]. The approach is to select subQUBOs of variables which contribute maximally to the problem energy. We solve the subQUBOs using Pasqal's neutral atom devices. The algorithm makes a set of calls to a subQUBO solver for global minimization and a call to classical search (Tabu [GL98]) for local minimization. Algorithm 1 shows the psuedocode.

The primary data structure underpinning this algorithm is the index vector. This vector arranges the variables in descending order of their influence on the solution's value. The influence of a variable is quantified as the change in the objective function value, where a higher impact leads to a less optimal result when the variable is negated in the current solution.

Once an initial value (best energy), solution (best solution), and index are determined, the main loop iterates through two phases of the computation. In the first phase, the input QUBO instance is divided into subQUBOs that fit within a given Pasqal device's capacity. The constant "fraction" specifies the portion of the QUBO instance partitioned into subQUBOs, typically falling within the range of 0.05 to 0.15. To optimize a set of variables S, the variables outside of S are held constant at their values in the current best solution. The subQUBO on S is defined as

$$f_S(x_S) = \sum_{i \in S} (Q_{ii} + d_i) x_i + \sum_{i,j \in S, i<j} Q_{ij} x_i x_j \tag{10}$$

where

$$d_i = \sum_{j \notin S} (Q_{ij} + Q_{ji}) x_j^* \tag{11}$$

Here $x*$ denotes the current best solution. Once the subQUBO is solved, we update the current solution by incorporating the relevant bits from the subQUBO solution vector. The objective here is to construct a fresh potential solution, which takes into account both the previous candidate solution and the updates obtained from the subQUBO results. The aim is to escape from a local minimum. In the event that none of the subQUBO updates produce any changes, we randomly assign values of 0 or 1 to those bits of the solution. This new solution is then fed into a tabu search process to seek out a new local minimum. When a superior solution is discovered, the passCount iteration counter within the main loop is reset. The algorithm will iterate up to numRepeats times in an effort to uncover a new best solution.

The hybrid design takes advantage of both classical search and the QAOA/QAA. While the quantum computer is highly effective at exploring various parts of the state space it can get trapped in local minima. On the other hand, the tabu search can locate an exact minimum inside a neighbourhood very rapidly, but it occasionally has trouble leaving it. The method can be thought of as a large-neighborhood local search, where each iteration produces tabu improvements.

5 Feasibility as opposed to present technical capacities

The proposed hybrid quantum-classical optimization approach is well-suited for implementation on Pasqal's neutral atom quantum processors. The QUBO formulation of the WFLO problem is compatible with Pasqal's architecture, and the problem decomposition strategies enable efficient utilization of the available qubits.

The scalability of the proposed approach is further enhanced by the use of problem decomposition strategies. As Pasqal's quantum processors continue to scale, larger and more complex WFLO problems can be tackled by partitioning them into smaller subQUBOs and solving them efficiently. Also, since QAA allows for controlled time evolution, the optimization process can be tailored, ensuring that time does not become a bottleneck.

6 Evidence for Analog implementation

The proposed hybrid quantum-classical optimization approach uses QAOA/QAA for solving the subQUBOs on neutral atoms which is supported by evidence from previous studies and successful implementation in other domains [Dal21] [CDH22]. The compatibility of QUBO with Pasqal's neutral atom architecture further supports the feasibility of the proposed approach.

7 Results and Interpretation

Table 1 contains the results for the classical as well as quantum samplers compared to the best results from [Sen+22] for the 12 WFLO instances. The runtime for Tabu Sampler was set to be always less than 50 seconds after which a timeout occurs while for QAA a mean runtime of 37.40 seconds was observed. The configuration can be found in the appendix.

Table 1: Total Expected Power (kW) after for the Classical Tabu Sampler and for QAA on Pasqal Devices (Simulation)

Wind Directions	n	m	Classical Tabu	QAA	Best from [Sen+22]
WR1	10	20	11185.40	11185.40	11185.41
		30	15739.57	15738.45	15742.93
		40	19265.21	19265.21	19265.21
	20	20	11404.79	11404.79	11404.80
		30	16774.37	16774.37	16774.37
		40	21947.00	21940.37	21973.80
WR36	10	20	19221.44	19221.44	19221.44
		30	27443.33	27443.33	27443.34
		40	34900.45	34900.45	35409.58
	20	20	19437.51	19437.51	19437.49
		30	28023.58	28023.58	27939.08
		40	35693.35	35693.35	35623.11

Algorithm 1 QbSolv algorithm for neutral atoms based on [BRR17]

Input: QUBO instance
best energy is the lowest value found to date
best solution is the solution bit vector corresponding to the lowest value so far
index is the indices of the bits in the solution, sorted from
most to least impact on value

Get initial estimate of minimum value and backbone
solution ← random 0/1 vector
(best energy, best solution) ← *TabuSearch*(QUBO, solution)
index ← *OrderByImpact*(QUBO, best solution)
passCount ← 0
solution ← best solution
while passCount < numRepeats **do**
    change ← false
    for i = 0; i < fraction ∗ Size(QUBO); i += subQUBOSize **do**
        # select subQUBO with other variables clamped
        sub index ← i : i+subQUBOSize−1
        subQUBO ← *Clamp*(QUBO, solution, index[sub index])
        (sub energy, sub solution) ← ***PasqalQAA***(subQUBO)
        # project onto the full solution
        if solution[sub index] ≠ sub solution **then**
            solution[sub index] ← sub solution
            change ← true
    if not change **then**
        Randomize(solution[0 : i − 1])
    (energy, solution) ← *TabuSearch*(QUBO, solution)
    if energy < best energy **then**
        best energy ← energy
        best solution ← solution
        passCount ← 0
    else
        passCount ++
    index ← *OrderByImpact*(QUBO, solution)
Output: best energy, best solution

We notice that the solution based on the QAA yields results similar to those obtained by Classical Tabu and those from [Sen+22] and for the last instance (36 wind regime, $n = 20, m = 40$) was able to surpass the best result from [Sen+22].

However, we are still not able to get the best out of the quantum computer. The next sections will highlight the problems associated with the current method and possible solutions.

8 Current Limitations

We are mostly focussed on QAA for this phase as it faster compared to the QAOA algorithm, however, the current embedding method lacks to capture the QUBO matrix. For the experiments, we use the embedding method mentioned in [Pulb] which tries to optimize the positions of the atoms using a classical optimizer to encode the off-diagonal terms of the QUBO matrix. This method is suitable for toy problems but does not perform well with the current problem. For example, the below QUBO matrix is for 36 wind regimes, 7 turbines on a gird on 3×3.

```
[[-137513.     31686.41   21079.11   31698.29   31622.6    21047.46   21085.72   21047.46   21039.96]
 [      0.   -137513.     31686.41   31708.06   31698.29   31622.6    21073.08   21085.72   21047.46]
 [      0.         0.   -137513.     21061.72   31708.06   31698.29   21074.16   21073.08   21085.72]
 [      0.         0.         0.   -137513.     31686.41   21079.11   31698.29   31622.6    21047.46]
 [      0.         0.         0.         0.   -137513.     31686.41   31708.06   31698.29   31622.6 ]
 [      0.         0.         0.         0.         0.   -137513.     21061.72   31708.06   31698.29]
 [      0.         0.         0.         0.         0.         0.   -137513.     31686.41   21079.11]
 [      0.         0.         0.         0.         0.         0.         0.   -137513.     31686.41]
 [      0.         0.         0.         0.         0.         0.         0.         0.   -137513.  ]]
```

One can notice that the values are too large and therefore we have to scale them to a smaller value to be able to fit them on the device. We divide them by the largest absolute value of the matrix and multiply by 15.7 (which is the maximum amplitude on chadoq2 device) that results into the following matrix.

```
[[-15.7   ,   3.6177,   2.4066,   3.619 ,   3.6104,   2.403 ,   2.4074,   2.403 ,   2.4022],
 [  0.    , -15.7   ,   3.6177,   3.6201,   3.619 ,   3.6104,   2.4059,   2.4074,   2.403 ],
 [  0.    ,   0.    , -15.7   ,   2.4046,   3.6201,   3.619 ,   2.4061,   2.4059,   2.4074],
 [  0.    ,   0.    ,   0.    , -15.7   ,   3.6177,   2.4066,   3.619 ,   3.6104,   2.403 ],
 [  0.    ,   0.    ,   0.    ,   0.    , -15.7   ,   3.6177,   3.6201,   3.619 ,   3.6104],
 [  0.    ,   0.    ,   0.    ,   0.    ,   0.    , -15.7   ,   2.4046,   3.6201,   3.619 ],
 [  0.    ,   0.    ,   0.    ,   0.    ,   0.    ,   0.    , -15.7   ,   3.6177,   2.4066],
 [  0.    ,   0.    ,   0.    ,   0.    ,   0.    ,   0.    ,   0.    , -15.7   ,   3.6177],
 [  0.    ,   0.    ,   0.    ,   0.    ,   0.    ,   0.    ,   0.    ,   0.    , -15.7   ]])
```

The issue is that we now need a higher precision while encoding the off-diagonal terms which the current method of embedding is not able to handle. Based on the above qubo matrix, we get the following layout of the atoms. We can see that, ideally we want and overlap between the 0^{th} atom and the atoms 1, 3, 4 because if we place a turbine at the 0^{th} location, due to proximity constraints, the neighbouring cells have to be excluded. This is however not captured by the below embedding and there we need sophisticated methods.

9 Arbitrary Connectivity

Based on the approach presented in [Ngu+23], we can perform an map a general QUBO instance to Maximum Weighted Independent Set (MWIS) problem on Unit Disk Graphs (UDG) which then can be implement on the Pasqal devices (even on the pre-calibrated

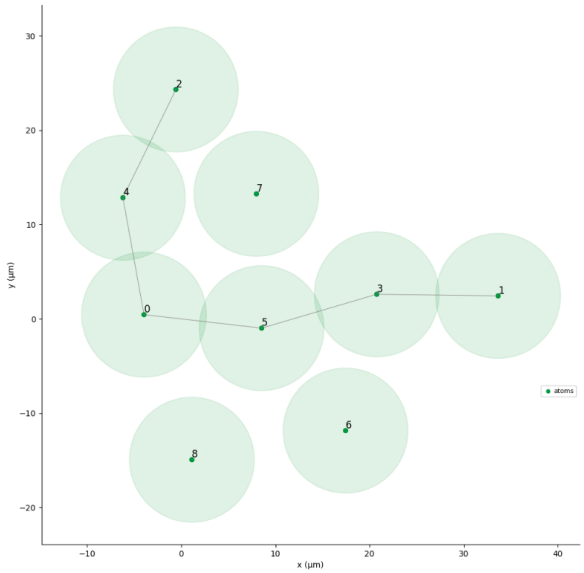

Figure 3: Example embedding for 3×3 grid

layouts). Once the problem is encoded in a UDG-MWIS, it can be embedded on a square lattice (whose diagonal sets the unit disk radius). Overall, any QUBO problem on N variables can be encoded in a UDG-MWIS problem with at most $4N^2 + O(N)$ vertices. Below image show the encoding of a random QUBO matrix of size 2×2. This can be found in the `misc.ipynb` notebook in the first section `multi-detuning`.

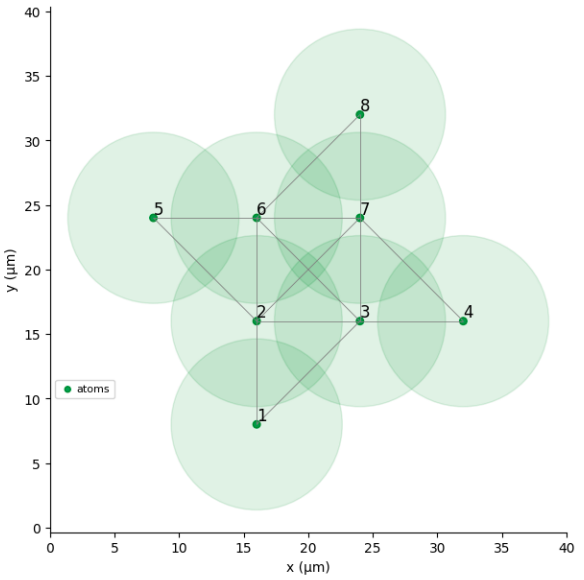

Figure 4: UDM-MWIS embedding for a 2×2 QUBO matrix

Since the UDM-MWIS embedding results into large variable count, the QbSolv algorithm can help to solve the subproblems one at a time and therefore reducing the variable count overhead. The challenge lies in efficiently solving the MWIS problem on the Pasqal's neutral-atom devies. The vertex weights of the MWIS problem in this Hamiltonian can

be implemented on the hardware with individual atom detunings (specifying $\Delta(t)$ for each atom). However, based on the tutorial on local addressability with DMM [Pula], the detuning has to be negative and thefore the final detuning expression for an atom becomes

$$\epsilon_i \Delta(t) + \Delta_g(t)$$

where $\Delta_g(t)$ represents the global detuning. This has to represent a straight line like the one shown below. As of now, we are not able to figure out the exact form and need mentor's help on this part of the problem.

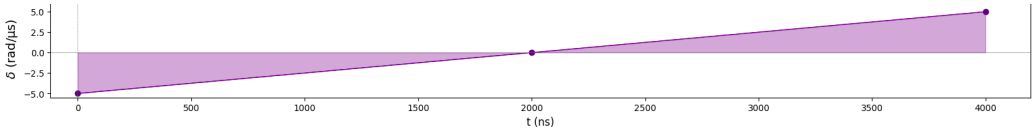

Figure 5: Example detuning

10 Appendix

Table 2: Configuration common for both classical and quantum

max subproblem size	10
max iterations	10

Table 3: Configuration specifc to quantum

duration of pulse (ns)	50000
sampling rate	0.1
number of samples	1000
n steps (Qutip emulator)	2000

References

[GL98] Fred Glover and Manuel Laguna. "Tabu Search". *Handbook of Combinatorial Optimization: Volume1–3*. Ed. by Ding-Zhu Du and Panos M. Pardalos. Boston, MA: Springer US, 1998, pp. 2093–2229. ISBN: 978-1-4613-0303-9. DOI: 10.1007/978-1-4613-0303-9_33. URL: https://doi.org/10.1007/978-1-4613-0303-9_33.

[BRR17] Michael Booth, Steven P. Reinhardt, and Aidan Roy. "Partitioning Optimization Problems for Hybrid Classical/Quantum Execution TECHNICAL REPORT". 2017. URL: https://api.semanticscholar.org/CorpusID:44160984.

[Dal21] Henriet Loïc Dalyac Constantin. "Qualifying quantum approaches for hard industrial optimization problems. A case study in the field of smart-charging of electric vehicles". *EPJ Quantum Technology* (2021).

[CDH22] Wesley da Silva Coelho, Mauro D'Arcangelo, and Louis-Paul Henry. "Efficient protocol for solving combinatorial graph problems on neutral-atom quantum processors" (July 2022). arXiv: 2207.13030 [quant-ph].

[Sen+22] Arik Senderovich et al. "Exploiting Hardware and Software Advances for Quadratic Models of Wind Farm Layout Optimization". *IEEE Access* 10 (2022), pp. 78044–78055. DOI: 10.1109/ACCESS.2022.3193143.

[Ngu+23] Minh-Thi Nguyen et al. "Quantum Optimization with Arbitrary Connectivity Using Rydberg Atom Arrays". *PRX Quantum* 4 (1 Feb. 2023), p. 010316. DOI: 10.1103/PRXQuantum.4.010316. URL: https://link.aps.org/doi/10.1103/PRXQuantum.4.010316.

[Pula] Pulser. *Local Adressability with DMM*. URL: https://pulser.readthedocs.io/en/stable/tutorials/dmm.html.

[Pulb] Pulser. *QAOA and QAA to solve QUBO problem*. URL: https://pulser.readthedocs.io/en/stable/tutorials/qubo.html.