Practical Test for Quantum Computing Scientist

Purpose and previous considerations

Continuing our last call, you have been invited to solve two different optimization problems to test your ability while solving these kinds of problems. These two problems are amongst the most classical and renowned problems around the optimization area of knowledge.

You are free to choose whatever programming language or tool you need to solve the problems, as well as everything needed to write the solutions to the problems and the document explaining your solutions. Just consider that we should be able to open any document in a Windows environment.

You can ask as many questions as you want until the day you send your solutions. This will not penalize in any way, so feel free to do that.

Deliverables

For each problem, you are required to:

- Develop a solution with the tools you want, to use python language is required. Although this is your criterion, the use a library such as pyqubo or dimod from D-Wave would be valuable since the position is focused on the quantum practice.
- Run this solution over the set of instances provided along with this document, if possible. If all the instances are huge enough to be run on your computer, you can create your own ones to test your code. The idea of running the solution is that we can do the same in our computers, so consider sending the project folder you used for an easy execution and not just the code.

- Write a short document or include this information in a Jupyter notebook:
    - With the QUBO (Quadratic Unbounded Binary Optimization) model that describes the problem if you have opted for solving the problem in a quantum-like way, detailing the objective function as well as the constraints you have considered.
    - With descriptions about the code and comments about the results obtained, such as execution time and quality of the solutions. Please, specify the external libraries and versions you have used, even for the programming language. Please consider attaching the requeriments.txt file to reproduce your results.
    - Any other considerations you want us to consider.

- Pack the code, results, and document into a zip file.

Please send all the zipped files in a single e-mail if possible.

1st problem: Quadratic Assignment Problem

Description of the problem

The Quadratic Assignment Problem (QAP) is a renowned problem in combinatorial optimization with applications in a wide variety of settings. The problem setting is as follows: we are given n facilities and n locations along with a flow matrix (fij) denoting the flow of material between facilities i and j. A distance matrix (dij) specifies the distance between sites i and j. The optimization problem is to find an assignment of facilities to locations to minimize the weighted flow across the system. The decision variables are xij = 1 if facility i is assigned to location j; otherwise, xij = 0. Then, the classic QAP model can be stated as:

$$\text{Minimize} \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ij} d_{kl} x_{ik} x_{jl}$$

$$\text{Subject to} \quad \sum_{i=1}^{n} x_{ij} = 1, j = 1, \dots, n$$

$$\sum_{j=1}^{n} x_{ij} = 1, i = 1, \dots, n$$

$$x_{ij} \in \{0,1\}, i, j = 1, \dots, n$$

Description of the instances

Each QAP instance correspond to a file containing the following information:

- Size of the problem, best solution value (if known)
- flow matrix
- distance matrix

Here is an example concerning a 5-variable instance:

```
          5 12902

 0  6 44 40 75
 6  0 79  0 89
44 79  0 35  9
40  0 35  0  1
75 89  9  1  0

 0 21 95 82 56
21  0 41  6 25
95 41  0 10  4
82  6 10  0 63
56 25  4 63  0
```

2nd problem: Quadratic Knapsack Problem

Description of the problem

Knapsack problems play a prominent role in the field of combinatorial optimization, having widespread application in such areas as project selection and capital budgeting. In such settings, a set of attractive projects is identified, and the goal is to identify a subset of maximum value (or profit) that satisfies the budget limitations. The classic linear knapsack problem applies when the value of a project depends only on the individual projects under consideration. The quadratic version of this problem arises when there is an interaction between pairs of projects affecting the value obtained.

For the general case with n projects, the Quadratic Knapsack Problem (QKP) is commonly modeled as:

$$\text{Maximize} \quad \sum_{i=1}^{n-1} \sum_{j=1}^{n} v_{ij} x_i x_j$$

$$\text{Subject to} \quad \sum_{j=1}^{n} a_j x_j \leq b$$

Where xj = 1 if project j is chosen; else, xj = 0. The parameters vij, aj, and b represent, respectively, the value associated with choosing projects i and j, the resource requirement of project j, and the total resource budget.

Description of the instances

Each QKP instance correspond to a file containing the following information:

- the reference of the instance (r_10_100_13 in the following example)
- the number of variables (n) (10 in the following example)
- the linear coefficients (c_i) of the objective function (91 78 22 4 48 85 46 81 3 26)
- the quadratic coefficients (c_ij) of the objective function
- a blank line
- 0 if the constraint is of type <= (i.e. always since we are considering QKP instances)
- the capacity of the knapsack (145)
- the coefficients of the capacity constraints (weights, a_i) (34 33 12 3 43 26 10 2 48 39)
- some comments

Here is an example concerning a 10-variable instance:

```
r_10_100_13
10
91 78 22 4 48 85 46 81 3 26
55 23 35 44 5 91 95 26 40
92 11 20 43 71 83 27 65
7 57 33 38 57 63 82
100 87 91 83 44 48
69 57 79 89 21
9 40 22 26
50 6 7
71 52
17

0
145
34 33 12 3 43 26 10 2 48 39

Comments

Density : 100.00 %
Seed : 13
```

This example corresponds to the following QKP instance:

$$
(QKP)\begin{cases}
\max & \begin{aligned}
&91x_1 + 78x_2 + 22x_3 + 4x_4 + 48x_5 + 85x_6 + 46x_7 + 81x_8 + 3x_9 + 26x_{10} \\
&+ 55x_1x_2 + 23x_1x_3 + 35x_1x_4 + 44x_1x_5 + 5x_1x_6 + 91x_1x_7 + 95x_1x_8 + 26x_1x_9 + 40x_1x_{10} \\
&+ 92x_2x_3 + 11x_2x_4 + 20x_2x_5 + 43x_2x_6 + 71x_2x_7 + 83x_2x_8 + 27x_2x_9 + 65x_2x_{10} \\
&+ 7x_3x_4 + 57x_3x_5 + 33x_3x_6 + 38x_3x_7 + 57x_3x_8 + 63x_3x_9 + 82x_3x_{10} \\
&+ 100x_4x_5 + 87x_4x_6 + 91x_4x_7 + 83x_4x_8 + 44x_4x_9 + 48x_4x_{10} \\
&+ 69x_5x_6 + 57x_5x_7 + 79x_5x_8 + 89x_5x_9 + 21x_5x_{10} \\
&+ 9x_6x_7 + 40x_6x_8 + 22x_6x_9 + 26x_6x_{10} \\
&+ 50x_7x_8 + 6x_7x_9 + 7x_7x_{10} \\
&+ 71x_8x_9 + 52x_8x_{10} \\
&+ 17x_9x_{10}
\end{aligned} \\
\text{s.t.} & \left| \begin{array}{l} 34x_1 + 33x_2 + 12x_3 + 3x_4 + 43x_5 + 26x_6 + 10x_7 + 2x_8 + 48x_9 + 39x_{10} \leq 145 \\ x_i \in \{0,1\}\ \forall i \in \{1,\ldots,n\} \end{array} \right.
\end{cases}
$$