for those of you who are not familiar with the the term, quantum annealing....

So now, for each product, we have a prediction of how many units will we need to buy at each day in order to supply the demand. The next challenge is to minimize the number of orders needed, without compromising the amount of stock needed. In order to do so, we can take into account many factors:

A. expiration date of the product

B. budget for a given period of time

C. warehouse capacity

D. If products have common suppliers, consider making orders with multiple products

Warehouse capacity doesn't look interesting for this particular case as we are talking about products that are usually small. Furthermore, budget has not been mentioned during the presentation of the problem, so we will consider it as negligible. On the other hand, expiration date and suppliers seem very interesting topics to dip into.

Is very easy to find a good solution considering only the expiration date with a greedy approach. However, when we add more products from the same supplier that doesn't look feasible anymore.

Decision variables

$x_{q,i,d}$ takes value 1 if q units of product i are ordered at day d, and takes value 0 if not.

Minimization term

$$H_{min} = \sum_{d} s_d^0$$

Where s_d^0 is a slack variable that should be 1 if some order is being placed at day d. By doing so, we are minimizing the number of orders made, and therefore optimizing the number of shipments needed under the assumption that all the selected products come from the same supplier.

To make the slack variable have the behavior we need, we need to consider that we have $|I|$ different products being ordered at the same day. Therefore, for each day d we can have

$$s_d = s_d^0 + \sum_{i=1}^{\lceil log_2|I| \rceil - 1} 2^{i-1} s_d^i + c_i s_d^{\lceil log_2|I| \rceil}$$

number of products being ordered. Here $c_i = |I| - \sum_{i=1}^{\lceil log_2|I| \rceil - 1} 2^{i-1} s_d^i$ making our variable s_d belong to the interval $[0, |I| + 1]$.

To ensure that s_d takes that value, we use the constraint:

$$H_{sum} = (s^d - \sum_{i} \sum_{q} x_{q,i,d})^2$$

$$H_{slack} = (1 - s_d^0)(\sum_{i \neq 0} s_d^i)$$

Constraints

We will add all these constraints as Lagrangian multipliers into the minimization term.

Given a product i and a day d, only one quantity q can be send:

$$H_{q\_encoding} = (\sum_i \sum_d x_{q,i,d})(\sum_i \sum_d x_{q,i,d} - 1)$$

To consider the expiration date, let's assume product i has expiration date t_i. Let's assume our previous model predicts that we need an amount of $q_{i,d}$ units on day d. For that, we need to impose:

We need to supply the total demand:

$$\sum_d q_{i,d} \leq \sum_d q x_{q,i,d}$$

This could be a soft constraint, and we saw how we can impose those inequalities using slack variables in order to obtain penalization terms. We need to ensure that we have the exact amount or more, as stock in healthcare can be critical.

We make sure that for each request there is non-expired stock available. We could find a good solution by imposing inequalities:

$$q_{i,d} \leq \sum_q \sum_{d-t_i<d'<d} q x_{q,i,d} \quad \text{Supply demand of day } d$$

$$\sum_{d-t_i<d'<d} q_{i,d} \leq \sum_q \sum_{d-2t_i<d'<d} q x_{q,i,d} \quad \text{Supply demand of days } d - t_i < d' < d$$

And so on... (until needed) Obtaining:

$$H^1_{sum}, H^1_{slack}, H^2_{sum}, H^2_{slack}, ...$$

The pros of doing a model like this is that we can easily add as many more contraints as we need, and play with their weights to have full control over the priorities of the model regarding soft constraints.

Unfortunately, we don't have the license needed to run this on a quantum device, but we were thinking of D-wave, using the dimod library to define the QUBO model.