Sphere packing using quantum algorithms

Rajas Chari

December 18, 2023

1 Introduction

These notes include an overview of a solution my team[1] proposed for our use case problem in BigQ Hackathon organized by the Chicago Quantum Exchange, and QuantX in CHicago in Septempber 2023. The Hackathon was organized in two stages, with this team handling the technical section where our objective was to use a specific quantum computing platform, Infleqtion(see Section 1.2), to solve an existing industry use case problem, in our case submitted by CSL Behring which we descibe in detail in Section 1.1. The use case problem can be briefly summarized as the problem of optimization of particle packing in chromatography columns. This question is of central importance in the downstream biopharmaceutical manufacturing workflow, in particular the solution to this problem can be used to optimize yields with high purity and homogeinity of target proteins when puriying proteins obtained from blood plasma.

The potential ability of Quantum computers to provide insight into optimizing particle packing and model the interactions inherent in the separation process is the motivation for this project. In our attempt at a solution, we focus on the problem of sphere packing, and provide a heuristic solution applicable to smaller system sizes(~ 33 particles on a machine with 2048 qubits) given the constraints of current day quantum computers. We provide an analysis of the algorithm in Sec. 2, where we explain our quantum-machine learning algorithm which uses a quantum computing module to provide a solution, and an ML based algorithm, also implemented on a quantum annealer, to tune the hyper-parameters of the objective functions. In Sec. 3, we provide results obtained by applying the algorithm to benchmark it against a classical heurisctic algorithm. We also conduct a thorough analysis of the algorithm's time complexity to enhance comprehension of its performance as system sizes increase, highlighting potential economies of scale, see Sec. 4.

1.1 Problem statement

The industry use case provided was that of engineering sphere packing for protein filtration columns. Protein filtration columns are used to seperate proteins in a mobile phases, by passing them through a chromatography bed, see Box below for details. The aim is to optimize the separation of proteins as they pass though the bed by finding the optimal configuration of particles in the bed. We ignore the microscopic detials of the problem, and pose it in a minimal manner in terms of a regular optimization problem, which we state as follows:

[1]The team, named SuperBalls, consisted of Carlo Seibenschuh, Bao Bach, Rohan Mehta, Shivam Mundhra and Kabir Dubey, and me(Rajas). We were also guided by our industry mentors: Victory Omole from Infleqtion, and Dr. Ian Njoroge from CSL Behring. We were also motivated by industry experts Dr. Noel Perez and Dr. Maen Qadan.

Problem statement: Given a column of linear dimension L, and particles with radii drawn randomly from a gaussian distribution $N(\mu, \sigma)$, pack the particles in the cloumn such that the below ojective funtions are optimized, while obeying the given constraints:

Objectives	Constraints
$o_1 : \sum_{i=1}^{N} z_i$	c_1: $r_i \leq x_i, y_i \leq L - r_i, W - r_i$ (Rectangular)
$o_2 : \left(\frac{\sum_i x_i \rho_i r_i^3}{\sum_i \rho_i r_i^3} - \hat{L}\right)^2 + \left(\frac{\sum_i y_i \rho_i r_i^3}{\sum_i \rho_i r_i^3} - \hat{W}\right)^2$	c_2: $\sigma < R - r_i$ (Cylindrical)
$o_3 : \max_{1 \leq i \leq N}(z_i + r_i)$	c_3: $(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2 \geq (r_i + r_j)^2$

where we want to optimize some effective objective function of the form: $O = \sum_i \lambda_i o_i$, with $\lambda_i \in \mathbb{R}$. The input parameters for the problem are defined below:

$$\begin{aligned} I &: \text{Set of particles} \\ m &: \text{number of particles} \\ r_i &: \text{radius of particle } i \in I \\ H, L, W(R) &: \text{height, length and width(radius) of the column} \end{aligned}$$

and the continuous variables used to characterize a solution or state of the system is given by:

$$(x_i, y_i, z_i) \in \mathbb{R}^3 \;\; \forall\, i \in I \tag{1}$$

which are defined relative to an appropriately defined orgin point on the column. We note that the optimization functions encourage the following configurations:

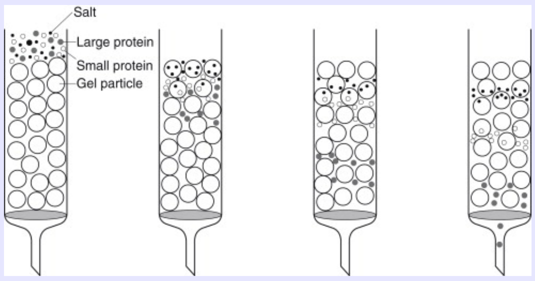
1. o_1 tries to minimize the height of the stacking.

2. o_2 tries to favour configurations in which particles are closer to a desired axis.

A detailed description of the problem statement provided during teh Hackathon can be found here.

Size Exclusion Chromatography(SEC)

Protein filtration using chromatography is implemented usually by passing a mobile phase, which contains the proteins we are interested in, through a stationary phase, which we also call the chromatography bed. We approximate the chromatography bed as consisting of particles which are suspended in a fluid medium. Size exclusion chomatography refers to separation of molecules according to their size. The typical scenario is demonstrated in the figure below, where the smaller paricles in the mobile phase take longer to pass through the medium due to travelling statisitically longer paths than larger molecules. This is because smaller particles have a larger set of paths they can take including paths through the particles in the stationary phase, and paths which avoid particles in the staionary phase. Whereas larger particles only have the option of going around the particles in the stationary phase.



The separation of the proteins can be undertood by looking at the concentration of the proteins collected at the end of the column as a function of time. This is done through a UV detector because proteins can absorb UV light. Sharper and well-separated peaks correspond to better separation of proteins. An example of the kind of data observed in experiments is shown below:

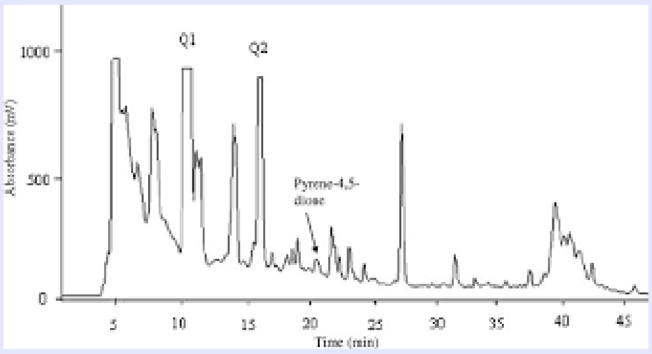


Figure 1: Experimental data showing absorbance as a function of time. Source?

The packing of the chromatography bed impacts the flow of the mobile phase throught the bed, which can lead to band broadening and loss of resolving power, which in turn negatively impacts the yeild of the proteins [1].

1.2 Resources and industry specifications

We were provided access to Infleqtion's Superstaq compiler on the Toshiba Bifurcation Machine, which is a quantum-inspired classical simulator. This allowed us to simulate an ideal quantum annealer, unlike previous experiments done on D-Wave's hybrid quantum-classical annealer.

2 Algorithmic details

In this section, we provide a detailed overview of the hybrid solution we proposed to solve the bin packing problem. The solution consists of a quantum computing module, discussed in Sec 2.1, which solves the bin packing problem discussed in the previous section, given a set of hyper-parameters $\{\lambda_i\}$. The solution is then analyzed using measures such as homogienity and optimality based on industry application, which we discuss in 2.2, according to which we then use an AI-based algorithm(see Sec. 2.3) to tune the hyper-parameters $\{\lambda_i\}$ in order to optimize said measures. A high-level caricature of this algorithm is shown in the below figure:

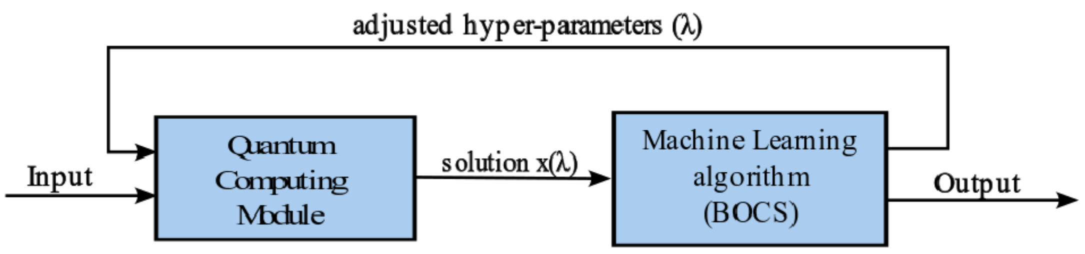

adjusted hyper-parameters (λ)
Input
Quantum Computing Module
solution x(λ)
Machine Learning algorithm (BOCS)
Output

Figure 2: Algortihm flow-chart.

where the quantum computing module outputs a solution $\{x_i(\lambda_\alpha)\}\ \forall\ i \in I$ which is the set of positions of all particles given a hyper-parameters $\{\lambda_\alpha\}$. The ML algorithm then computes a cost function based on given industry based measures, like the homohienity measure which we will denote as $H(\lambda_\alpha)$, and adjusts the hyper-paramters in a feedback loop. The algorithm terminates once a desired threshold of the cost function is achieved. It is important to note that the ML algorithm can also be formulated as a QUBO and implemented on the same quantum hardware as the other quantum module.

2.1 Quantum computing modules

The quantum computing module of the algorithm is mainly concerned with converting a given binary optimization problem into a from which can be solved on a quantum anneling platform, like Dwave. Since, the state-space of the problem stated in the previous section is continuous, ie. $\{x_i \in \mathbb{R}\}$, it is appropriate to use constrained quadratic model(CQM) based methods from the Ocean software libraries, which are used for running problems on quantum annealers. The CQM can then be fed into a post-processing subroutine which returns a binary quadratic model(BQM) as the output, which encodes the problem with a state space that is binary. The binary quadratic model (BQM) class encodes Ising and quadratic unconstrained binary optimization (QUBO) models used by samplers such as the D-Wave system, and hence we can feed this into a Dwave machine or a classical quantum anneling simulator like the Toshiba bifurcation machine.

In breif, the whole process amounts to rewriting the optimization problem in terms of a QUBO, which can then be encoded into a Ising Hamiltonian. This can be understood by rreformulating a binary optimization in terms of a single "effective Hamiltonian", which need to be optimized:

$$H = \sum_{i<j} x_i Q_{ij} x_j + \sum_i \lambda_i x_i \tag{2}$$

where $\{x_i\}$ are binary observables which describe a state, and $\{\lambda_i\}$ can be understood to be Lagrage multipliers associated to the constraints imposed on the solutions. Q_{ij} encode the objective function in the optimization problem. Finally, this effective Hamiltonian can then be simulated as an Ising model on a quantum annealer. We note that there are non-trivial issues when it comes to implementing certain constrains which cannot be easily recast into a quadratic form. The details of the implementation of the algorithm and pseudo-code can be made available upon reasonable request.

2.2 Homogienity measures

Given the output of the quantum part of the algorithm, which returns the positions of each type of particle, which we model as follows:

$$\text{output data} = \{(x_i, y_i, z_i, r_i) | i \in \text{I}\} \tag{3}$$

where the set I is part of the input to the quantum algorithm which is the set of particles, ie. **each particle is assigned an index in this set I.** This set can contain different types of particles, in general each type having different radius. Given this output data, we can define a measure of homogienity for each *type* of particle. The measure we consider is the analog of **coefficient of variation** in experimental literature[**cov˙1**, **cov˙2**]. Experimenters mix compounds and then extract the sample concentration means, denotes μ_i, for a given compound in the mixture, denoted x_i, for small samples taken from the mixture to calculate the sample variance:

$$s^2 = \frac{1}{n}\sum_i (\mu_i - \bar{\mu})^2 \tag{4}$$

where we consider n samples, and $\bar{x}$ is the mean of the sample concentrations, ie. $\bar{x} = (1/n)\sum_i \mu_i$. The results depend on a parameter called the *scale of scrutiny*, which in our case is the coarse graining scale, which we denote as a. Using this they calculate the coefficient of variation defined as:

$$\text{CoV} = \frac{s}{\bar{\mu}} \tag{5}$$

We consider the discretiation of space into cells which we will be treating as our samples, each of which is labelled by a coordinate at the center of the cell which we will denote as (X_i, Y_i). To be explicit the cell would be defined as the volume:

$$\text{Cell}_i = \{(x_i, y_i, z_i) \mid X_i - \frac{a}{2} \le x_i \le X_i + \frac{a}{2};\ Y_i - \frac{a}{2} \le y_i \le Y_i + \frac{a}{2};\ Z_i - \frac{a}{2} \le z_i \le Z_i + \frac{a}{2}\}\} \tag{6}$$

Now, given a particle type which we shall denote with Greek variables $\alpha, \beta ...$, we can compute the sample concentration by volume. We note that the coarse graining length is much larger thean the radii of particles, ie. $a \gg \max(r_i)\ \ \forall\ i \in I$, so that edge effects are suppressed to $O(r_i/a)$. We also want to sample from a region in which there are particles, so given the answer for the maximum height H we sample cells such that $Z_i \le H - (a/2)$[2]. Let the set $T_p = \{i \mid \vec{x}_i \in Vol(C_p)\}$ denote the set of particles which have their COM inside the cell C_p. And let the set $T_p^\alpha = \{i \mid \vec{x}_i \in Vol(C_p)$ and $r_i = R_\alpha\}$ denote the set of particles of type α which have their COM inside the cell C_p. The average concentration, by volume, is then given by:

[2]Check $H > (a/2)$ as a sanity check first!

$$\mu_p = \frac{\sum_{i \in T_p^\alpha} R_\alpha^3}{\sum_{i \in T_p} r_i^3} \tag{7}$$

using these sample means we can calculate the coefficient of variation for our problem by simply using Eqn. 4 and Eqn. 5. We use this homogienity measure as an industry specification to be incorporated in our acquisition function to tune the hyper-parameter.

2.3 ML module

In the ML part of the algorithm, we want hyperparameter optimization (HPO) $\lambda \in \mathbb{R}^{\binom{n}{2}}$ for quantum annealing, which impacts regularization of the problem. In the Bayesian setting, we tune selective components λ with few data points. And in BOCS 2018[2], we update $\lambda_i = \kappa$ if $i \in I$. TL;DR: the acquisition function optimization is also a binary quadratic program $\Longrightarrow$ entire optimization on the **same** quantum hardware.

3 Results

We benchmark the quantum computing module against stochastic search, which is a classical heuristic, and see a performance improvement over small system sizes.

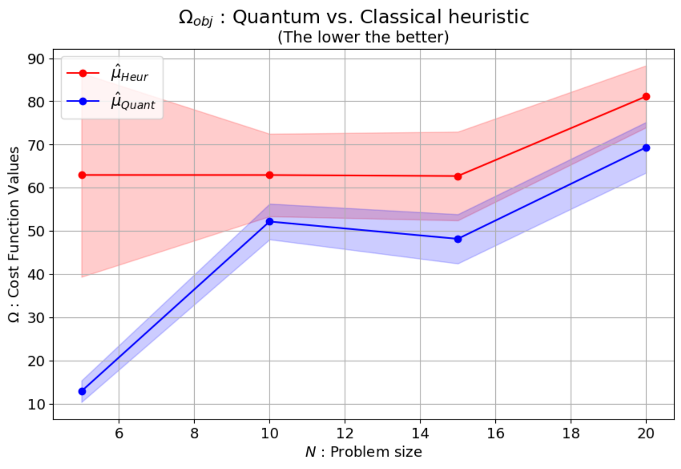


Figure 3: Experiment on capability of quantum annealing versus classical heuristic (stochastic search). On average, 33.6% improvement across problem sizes.

4 Scaling analysis

In this section, we compute the scaling behavior of our algorithm. We first compute the space complexity, or number of qubits that would be required to simulate a system with N particles in d dimensions. And then we go on to give a heuristic measure for the time complexity of the algorithm.

4.1 Space complexity

Given particle sizes with $r \sim 1\mu$m, and a box of linear size $L \sim 10$ cm, assuming we aim for a resolution of $\delta x \sim 10$ nm, we would need 20 bits of data to encode the position of one particle in one dimension. Hence, given N particles and $d = 3$ dimensions, we would need $60N$ qubits to encode a state of the problem:

$$H_P = \sum_{ij} J_{ij} x_i x_j + \sum_i h_i x_i$$

.

As discussed in the section on algorithm analysis, the objective functions and constraints are encoded in the matrix J_{ij} and h_i in the Ising model Hamiltonian. For example, D-Wave's Chimera machine (2043 functional qubits), we could simulate the problem of packing $N \approx 33$ particles.

4.2 Time complexity

The relavant time scale is called the time-to-solution and is equal to the annealing time(t_{program}) times the number of gauges G, ie. $\text{TTS} = Gt_{\text{program}}$. The time complexity of the quantum annealing algorithm on the D-wave machines scales linearly with the number of qubits in the problem[3]:

$$\text{TTS}(t_f) = t_f R(t_f) \frac{N}{N_{\text{max}}}; \qquad t_f = G \times t_{\text{inst}}$$

- t_{inst} is the runtime for a single instance of the algorithm($\sim 10\mu$sec for Chimera[3]) with G different gauge choices for $|\psi_{\text{in}}\rangle$.

$$H(s) = H_0(1-s) + H_P(s); \qquad |\psi_{\text{in}}\rangle$$

- N/N_{max} accounts for the possibility for parallelization of the code.
- $R(t_f)$ is a measure for the required number of runs for success.

where we can see clearly that the algorithm scales linaearly in the number of qubits used in the algorithm, which from the previous section we know is directly proportional to the number of partcles we want to simulate. Hence, we get linear scaling with time for the quantum computing module of the algorithm.

5 Future directions

An optimistic picture can be painted for the future of algorithms such as the one descibed in this manuscript. There have been rapid developments in theoretical quantum adiabatic optimization. With recent work on annealing within the constrained subspace[4] and better problem mappings with:

- Physics-inspired objectives (e.g. homogeneity)
- Linearizing constraints
- Encoding contact graphs

Gate-based implementations via discretization and trotterization have also been explored for use in simalar problems.

[3]D-wave has 2048 functional qubits in its Chimera machine. minimum annealing times for all D-Wave processors involved in benchmarking studies to date are: 5μs for the D-Wave One, D-Wave Two X, and D-Wave 2000Q, and 20μs for the D-Wave Two.

References

[1] Darryl Yc Kong et al. “Effects of bed compression on protein separation on gel filtration chromatography at bench and pilot scale”. en. In: *J Chem Technol Biotechnol* 93.7 (Oct. 2017), pp. 1959–1965.

[2] Ricardo Baptista and Matthias Poloczek. “Bayesian Optimization of Combinatorial Structures”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, Oct. 2018, pp. 462–471. URL: https://proceedings.mlr.press/v80/baptista18a.html.

[3] Tameem Albash and Daniel A. Lidar. “Demonstration of a Scaling Advantage for a Quantum Annealer over Simulated Annealing”. In: *Phys. Rev. X* 8 (3 July 2018), p. 031016. DOI: 10.1103/PhysRevX.8.031016. URL: https://link.aps.org/doi/10.1103/PhysRevX.8.031016.

[4] Itay Hen and Federico M. Spedalieri. “Quantum Annealing for Constrained Optimization”. In: *Phys. Rev. Appl.* 5 (3 Mar. 2016), p. 034007. DOI: 10.1103/PhysRevApplied.5.034007. URL: https://link.aps.org/doi/10.1103/PhysRevApplied.5.034007.