Quantum Science and Technology

PAPER

Quantum molecular unfolding

Kevin Mato[1,*], Riccardo Mengoni[2], Daniele Ottaviani[2] and Gianluca Palermo[3]

[1] Technical University of Munich, Chair of Design Automation, Germany
[2] CINECA, Quantum Computing Lab, Italy
[3] Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Italy
[*] Author to whom any correspondence should be addressed.

E-mail: kevin.mato@tum.de

Keywords: molecular docking, combinatorial optimization, high performance computing, simulated annealing, quantum annealing, molecular unfolding, quantum computing

OPEN ACCESS

RECEIVED
16 January 2022

REVISED
25 April 2022

ACCEPTED FOR PUBLICATION
26 May 2022

PUBLISHED
9 June 2022

Abstract

Molecular docking is an important step of the drug discovery process which aims at calculating the preferred position and shape of one molecule to a second when they are bound to each other. During such analysis, 3D representations of molecules are manipulated according to their degree of freedoms: rigid roto-translation and fragment rotations along the rotatable bonds. In our work, we focussed on one specific phase of the molecular docking procedure i.e. *molecular unfolding (MU)*, which is used to remove the initial bias of a molecule by expanding it to an unfolded shape simpler to manipulate within the target cavity. The objective of the MU problem is to find the configuration that maximizes the molecular area, or equivalently, that maximizes the internal distances between atoms inside the molecule. We propose a quantum annealing approach to MU by formulating it as a high-order unconstrained binary optimization which was possible to solve on the latest D-wave annealing hardware (2000Q and advantage). Results and performances obtained with quantum annealers are compared with state of art classical solvers.

1. Introduction

Drug discovery [1] is a process that includes several phases, from virtual *in silico* simulations [2] to *in vitro* and *in vivo* experimentation. Molecular docking [3] is an important step of the drug discovery process which aims at calculating the preferred position and shape of one molecule to a second when they are bound to each other. The computational resources needed to address the docking-related tasks are usually quite large, as problems of this kind involve several degrees of freedom and rapidly growing dimensionality.

In recent years, the field of quantum computing (QC) [4] has undergone significant developments from both hardware and algorithms points of view. In particular, we are living in the so-called NISQ [5] era in which it is not clear whether the quantum devices currently available are actually capable of producing better or comparable results compared to classical methods.

In this work, we explored the possibility of using a QC technique called quantum annealing (QA) [6–8] to support the molecular docking phase. In particular, we focussed on the molecular unfolding (MU) process, which is the first step in geometric molecular docking techniques. MU aims at finding the molecular configuration that maximizes its volume, or equivalently, that maximizes the internal distances between atoms inside the molecule. This phase is not available in all the docking algorithms [9], but it is used for initializing a geometric procedure [10]. In any case, the interesting point of this procedure is that it involves the same degrees of freedom that a molecule has during the molecular docking phase, without involving a second entity, i.e. the target protein. Our aim was to develop a quantum MU approach and execute it on the latest QA hardware (D-wave advantage and 2000Q) in order to understand the capabilities of these devices and compare their performances with respect to classical methods.

The remainder of the paper is structured as follows. In section 2, the topic of molecular docking is introduced, while in section 3 the problem of MU is discussed. In sections 4 and 5, the concepts of QA are illustrated followed by the binary optimization formulation of the target problem. Later in sections 6 and 7,

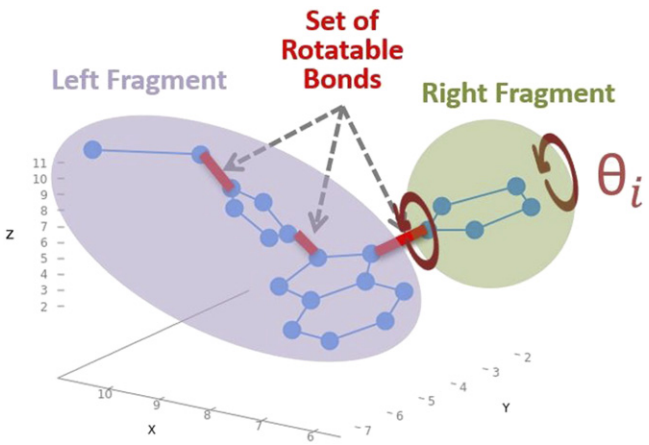

Figure 1. Rotatable fragments and rotatable bonds.

the dataset and pre-processing steps employed before the unfolding phase is presented. This is followed in section 8 that provides by a review of the state-of-the-art classical algorithms employed to compare solutions obtained with the quantum devices. In sections 9 and 10, the experimental results regarding the embedding on the QPUs are presented as well as a detailed performance comparison between the different methods. Finally, section 11 concludes the paper.

2. Preliminaries on molecular docking

In-silico approach to drug discovery [1] can be seen as a multi-tiered process that encompasses several sequential computational techniques with the aim of screening virtual libraries of the order of billions of compounds for the most suitable molecules to forward to later experiments.

Molecular docking [3] is an essential step in virtual screening which is used to simulate the atomic interactions of a ligand inside a protein binding site, in order to highlight their possible biochemical reactions and predict whether a stable complex could be formed.

The molecular docking process is divided into two main tasks:

- Detection of three-dimensional *poses* of the ligand, i.e. valid ligand conformations, positions, and orientations inside the active site of the protein (usually called *pocket*).
- Ranking of the poses via a scoring function. Usually the lower the docking score, the better the resulting *binding affinity*.

Since electrons inside atoms are repulsive to each other, their interaction affects both molecular shape and reactivity. Therefore, by controlling the molecular shape, it is possible to predict the ligand reactivity to a protein's pocket as well as the energy cost of such configuration. This direct connection between molecular conformation and binding affinity is what enables geometrical scoring functions [10].

In our approach, the docking process considers the pocket as a rigid structure, while the ligand is a flexible set of atoms. Furthermore, from a strictly geometrical interpretation, the ligand can be seen as a set of chemical bonds (or edges) with a fixed length, where only a subset of edges are *rotatable*.

Rotatable bonds (or torsionals) are bonds that split the molecule in two nonempty disjointed fragments when virtually removed. The two fragments that are linked to a rotatable bond can rotate independently from each other around the axis of the rotatable bond. In the remainder of the paper we will call such fragments as *rotatable fragments*. These concepts are graphically reported in figure 1, where the rightmost rotatable bond is splitting the molecule in left and right rotatable fragments. Finally, note that the number of rotatable fragments in a molecule is usually double the number of rotatable bonds.

In general, when applying a geometric approach such as in [10], it is possible to point out three main phases in molecular docking: *ligand expansion*, *intial placement* and *shape refinement* inside the pocket.

In this work, we focus on the ligand expansion phase, which is essential for improving the quality of a geometric docking procedure. An initial pose of the ligand that is set *a priori* may introduce shape bias affecting the final quality of the docking. MU is the technique used for removing such initial bias by expanding the ligand to an unfolded shape.

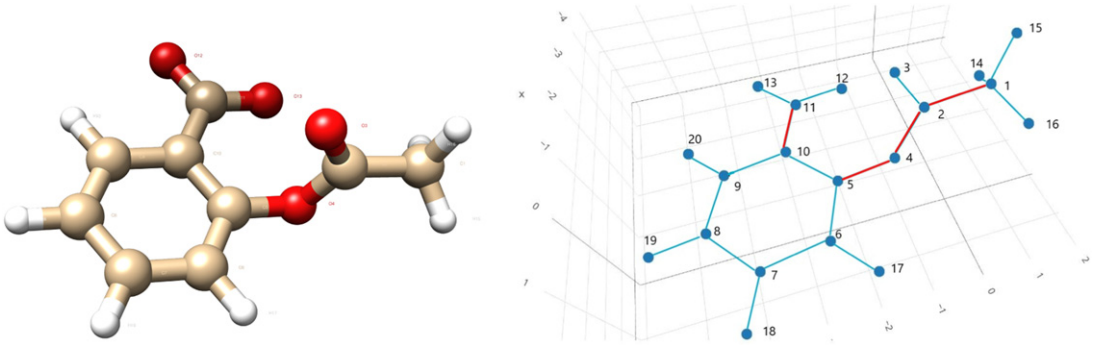

Figure 2. Illustration of a simple molecule. The molecular structure presents four rotatable bonds, highlighted in red in the graph representation.

3. Molecular unfolding problem definition

The objective of the MU problem is to find the unfolded shape of the ligand or, in other terms, the torsional configuration that maximises the molecular volume expressed as the total sum of internal distances between pairs of atoms in the ligand. The starting point is the folded molecule defined by a set of atoms together with its fixed and rotatable bonds. We can assign to each rotatable bonds t_i a variable corresponding to the angle θ_i which is responsible for *rotatable* fragment rotation. As a convention, we identify the ordered set of rotations by a vector:

$$t = \left[\theta_1, \theta_2, \ldots, \theta_n\right] \tag{1}$$

where each torsion θ_i around the bond's axis can assume values $[0, 2\pi)$. Given a molecule, it is possible to construct the associated graph as shown in figure 2 where edges represent bonds (which can not contract). Torsional bonds are depicted in red in the image below.

The objective for the MU problem can be expressed in mathematical terms as follows: given a molecule, find the torsional configuration

$$\mathbf{t}^{\text{unfold}} = [\theta_1^{\text{unfold}}, \ldots, \theta_M^{\text{unfold}}] \tag{2}$$

such that the following quantity is maximised

$$D(\vec{\mathbf{t}}) = \sum_{\substack{a,b\in\mathcal{M}\\ a\neq b}} D_{ab}(\Theta)^2. \tag{3}$$

The function $D_{ab}(\Theta)$ denotes the distance between two different atoms a and b belonging to the molecule $\mathcal{M}$ while $D(\vec{\mathbf{t}})$ is the sum of all square distances between atoms.

The rationale behind this choice is that the objective function is simple and relies just on the geometry of the molecule. Each distance $D_{ab}(\Theta)$ only depends on the angles induced by those rotatable bonds that appear in the shortest path connecting atom a to b in the graph.

It is worth noticing that it is not necessary to calculate all the pairwise distances that are expressed in equation (3). In fact, it is possible to apply simplifications to $D_{ab}(\Theta)$, as expressed by the following two conditions:

Condition 1. The shortest path between two atoms must contain at least one torsional. We define as *rigid fragment* a subset of the atoms of the molecule whose shortest path among each pair does not contain any rotatable bond. If two atoms belong to the same rigid fragment, their relative position never changes.

Condition 2. The shortest path connecting two atoms must have at least three edges. This is a consequence of the fact that atoms connected by only one or two bonds have always the same distance, even when bonds are rotatable.

A useful mathematical representation for rotations are rotation matrices, $R(\theta_i)$, which are function of the angle θ_i associated to torsion T_i. If two atoms are connected via multiple torsional, the rotation matrix $R(\Theta) = R(\theta_i, \theta_j, \ldots, \theta_k)$, characterizing their relationship is an ordered multiplication of single torsion rotation matrices

$$R(\Theta) = R(\theta_i, \theta_j, \ldots, \theta_k) = R(\theta_i) \times R(\theta_j) \times \cdots \times R(\theta_k). \tag{4}$$

Each rotation $R(\theta_i)$ is expressed via a 4×4 rotation matrix acting on atomic coordinates. The extremes of a bond connecting two atoms are identified by the coordinates of the atoms themselves. The explicit formulation of the rotation matrix can be found in the appendix.

Consider now the distance $D_{ab}(\Theta)$ and the initial positions of atoms a and b which are identified with $\vec{a_0}$ and $\vec{b_0}$ respectively. Relative positions are obtained by fixing one of the two atoms while rotating the other

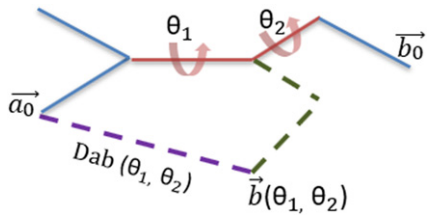

Figure 3. Visual representation of the action of two rotations θ_1 and θ_2 on the distance expressed in equation (6).

as follows

$$\vec{a} = \vec{a_0} \qquad \vec{b} = R(\Theta)\vec{b_0}. \tag{5}$$

Hence, we can rewrite the single contributes in equation (3) using the Euclidean distance as

$$D_{ab}(\Theta)^2 = \|\vec{a_0} - R(\Theta)\vec{b_0}\|^2. \tag{6}$$

Figure 3 shows how the relative positions change as function of the angles assumed by the two rotatable bonds.

4. Quantum annealing

QA is a meta-heuristic technique for addressing challenging combinatorial problems. It can be seen as a variation of the *simulated annealing (SA)* [11] algorithm where quantum fluctuation, instead of thermal effects, are employed in the exploration of the configuration space. QA devices, like those implemented with *superconductive qubits* by the Canadian company *D-wave*[4], can solve combinatorial problems expressed in quadratic unconstrained binary optimization (QUBO) terms. A generic QUBO problem is defined as

$$O(x) = \sum_i h_i x_i + \sum_{i>j} J_{i,j} x_i x_j \tag{7}$$

where $x_i \in \{0, 1\}$ are binary variables while h_i and J_{ij} are parameters whose values encode the optimization task in such a way that the minimum of $O(x)$ represents the solution to the optimization problem.

In general, a given optimization problem can be easily translated into binary combinatorial terms as a high-order QUBO (high-order unconstrained binary optimization (HUBO)) problem

$$O_{\text{Hubo}}(x) = \sum_i \alpha_i x_i + \sum_{i,j} \beta_{i,j} x_i x_j + \sum_{i,j,k} \gamma_{i,j,k} x_i x_j x_k + \cdots. \tag{8}$$

Fortunately, it is always possible to convert HUBOs into QUBOs. The trick used is to add new ancillary binary variables and substitute high-order terms with sums of quadratic expressions (made up with original binary variables and new ones) to preserve local and global minima [12].

As an example, a cubic term like $\pm x_1 \cdot x_2 \cdot x_3$ can be divided using the ancillary binary variable a_1 as

$$\pm x_1 \cdot x_2 \cdot x_3 \to \pm a_1 x_3 + 2(x_1 x_2 - 2x_1 a_1 - 2x_2 a_1 + 3a_1). \tag{9}$$

In practice, one can decide to build a custom function for quadratisation or exploit the functions present in the D-wave software offer, such as `make_quadratic`[5].

5. HUBO formulation

As a first step, it is necessary to formulate the MU problem in binary optimization terms to employ a QA approach.

Let us consider a discretisation of the angle θ_i associated to a rotatable bond T_i into d possible values

$$\theta_i = [\theta_i^1, \theta_i^2, \theta_i^3, \ldots, \theta_i^d]. \tag{10}$$

[4] dwavesys.com

[5] https://docs.ocean.dwavesys.com/en/stable/docs_dimod/reference/higherorder.html.

As a consequence, continuous functions of the angles like sine and cosine can be discretised as well in d values

$$\sin(\theta_i) = [\sin(\theta_i^1), \sin(\theta_i^2), \sin(\theta_i^3), \ldots, \sin(\theta_i^d)] \tag{11}$$

$$\cos(\theta_i) = [\cos(\theta_i^1), \cos(\theta_i^2), \cos(\theta_i^3), \ldots, \cos(\theta_i^d)]. \tag{12}$$

Moreover, since torsional angles can assume only one value at a time, it is necessary to associate each θ_i with only one value among all the possible θ_i^k. A *one-hot* encoding strategy is adopted for each torsional T_i, to which are assigned binary variables x_{ik}, with $1 \leqslant k \leqslant d$, such that

$$x_{ik} = \begin{cases} 1 & \text{if } \theta_i = \theta_i^k; \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

Definition (13) tells us that only one among the binary variables can be assigned to a one or truth value. This means that their sum is always equal to one:

$$\sum_{k=1}^{d} x_{ik} = 1. \tag{14}$$

Note that equation (14) must hold for all rotatables. To convert such algebraic constraint into QUBO, the following term has to be considered:

$$\sum_i \left(\sum_{k=1}^{d} x_{ik} - 1 \right)^2. \tag{15}$$

The constraint of equation (15) is usually called a *hard constraint* because it is mandatory to satisfy this term i.e. minimize it in the combinatorial optimization.

The selection of the one-hot encoding strategy has been done because of the direct and simple problem formulation. Recently, a novel encoding technique called *domain-wall* [13] have been proposed for similar problems demonstrating a better embedding capability [14, 15]. The evaluation of different encoding strategies is out of the scope of the current paper and it is considered a future work.

Regarding those functions that appear in the rotation matrix $R(\theta_i)$ such as $\sin(\theta_i)$ and $\cos(\theta_i)$, they can be expressed as

$$\sin(\theta_i) = \sum_{k=1}^{d} \sin(\theta_i^k)\ x_{ik} \tag{16}$$

$$\cos(\theta_i) = \sum_{k=1}^{d} \cos(\theta_i^k)\ x_{ik}. \tag{17}$$

Therefore, a rotation matrix $R(\theta_i)$ becomes a function of all the binary variables x_{ik} needed to represent the angle θ_i

$$R(\theta_i) = R(x_{i1}, x_{i2}, \ldots, x_{id}). \tag{18}$$

For a generic rotation we have

$$\begin{aligned} R(\Theta) &= R(\theta_i) \times R(\theta_j) \times \cdots \times R(\theta_k) \\ &= R(x_{i1}, x_{i2}, \ldots, x_{id}) \times R(x_{j1}, x_{j2}, \ldots, x_{jd}) \times \cdots \\ &\times R(x_{k1}, x_{k2}, \ldots, x_{kd}). \end{aligned} \tag{19}$$

Note that, for what concerns the granularity of the rotations, in order to obtain a precision of $\Delta\theta_i = \theta_i^{k+1} - \theta_i^k$ in the representation of angle θ_i, the number of variables x_{ik} needed for each torsion is given by

$$d = \frac{2\pi}{\Delta\theta_i} = \frac{2\pi}{\theta_i^{k+1} - \theta_i^k}. \tag{20}$$

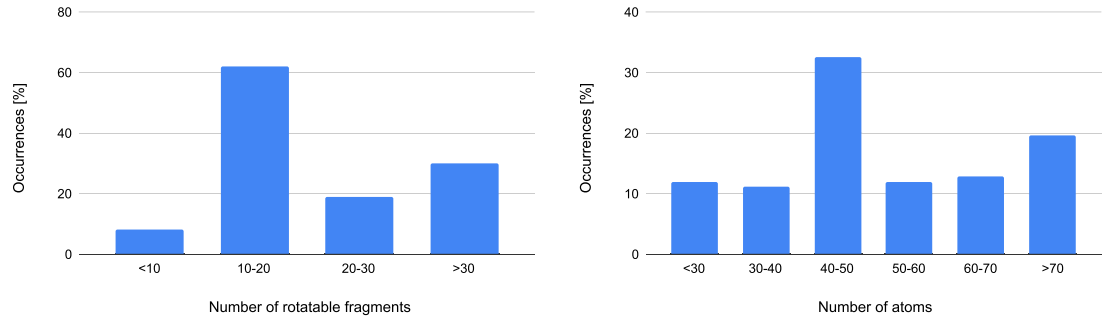

Figure 4. Analysis of the distribution for number of atoms and number of rotatable fragments.

Given a number M of torsional bonds, the total number of binary variables is

$$n = d \times M = \frac{2\pi}{\Delta\theta_i} \times M. \tag{21}$$

The general form of the HUBO optimization function can be written as

$$O(x_{ik}) = A_{\text{const}} \sum_i \left(\sum_{k=1}^{d} x_{ik} - 1 \right)^2 - \sum_{a,b} D_{ab}(\Theta)^2. \tag{22}$$

In the last equation, the second term identifies the *optimization term*, which has a minus sign in front because it should be maximized (while the whole expression is minimized). The first term instead represents the *hard constraint*, where the parameter A_{const} is a penalty scalar that modulates its strength. Finally, it is worth noticing that, given a molecule with M torsionals, the highest order term that appears in equation (22) is $2M$, where the number 2 comes from the squared distances.

6. Dataset and pre-processing

6.1. Ligand dataset

The ligand dataset in our possession is derived by PDBbind [16] and made up of 118 molecules, with a number of atoms ranging from 20 to more than 120 and a number of torsionals up to about 50. Figure 4, shows an overview of the ligand distributions for the dataset under analysis in terms of rotatable fragments and number of atoms. It is possible to see that 60% of the ligands in our database have between 10 and 20 rotatable fragments, which means 5–10 rotatable bonds. The number of atoms has a peak at about the 40–50 interval. Within our study, we considered molecules with a number of rotatable bonds up to 12 and a number of atoms up to 50. This gave us the chance to highlight the increment of the problem complexity by increasing the two characteristics of the molecules, without making it explode too much.

6.2. Pre-processing phase

The pre-processing starts with the parsing of the information about the molecule via analysis of its MOL2 file [17]. A graph containing a three-dimensional description of the ligand together with the bond type (bonds could be single, double, triple, amide, or aromatic) is constructed.

Two simplifications related to chemical–physical properties of ligands are applied in the pre-processing phase: one is because only single bonds can be rotatable, while the other concerns the removal of terminal hydrogen atoms whose distances with the rest do not affect the total sum of internal distances. Note that the removal of terminal hydrogens may trigger the generation of conformations where the final position of atoms does not satisfy their minimum Van der Waals distance. Such invalid shapes are discarded in post-processing.

After this, the graph associated to the molecule is divided into rigid fragments as shown in figure 5. At this stage, the betweenness centrality [18], defined in equation (23), is employed to obtain an ordering of the atoms within the graph. As a consequence, this procedure also specifies an ordering of rotatable bonds

$$\text{Betweenness centrality of atom } v: \quad b_v = \sum_{s,t} w_{s,t}^v = \sum_{s,t} \frac{n_{\sigma_{s,t}}^v}{n_{\sigma_{s,t}}}. \tag{23}$$

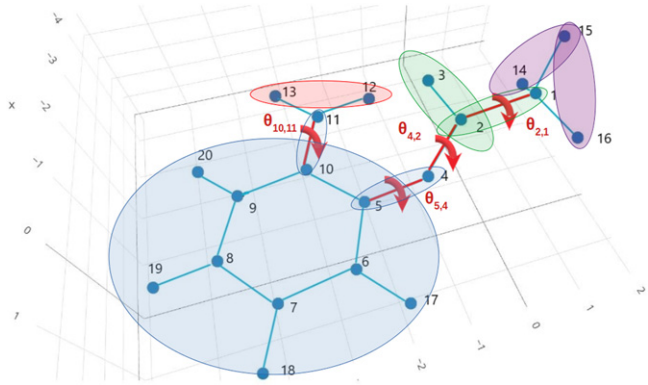

Figure 5. Atoms belonging to the same rigid fragment, depicted using colours, are affected by the same rotatables influence set.

The betweenness centrality value for an atom v is calculated by accumulating its contribution $w^v_{s,t}$ for each pair of atoms, s and t, of the molecules. Let us consider, $\sigma_{s,t}$ as the shortest path between s and t. The value $w^v_{s,t}$ is derived as the number of all the possible shortest paths connecting atoms s and t which also cross v, $n^v_{\sigma_{s,t}}$, divided by the total number of shortest paths between s and t, $n_{\sigma_{s,t}}$. The weight $w^v_{s,t}$ has a value between 0 and 1 and represents the betweenness of the atom v with respect to the pair s, t. If $s = t$ and then $n_{\sigma_{s,t}} = 0$ the corresponding weight $w^v_{s,t}$ is considered 0. The atom with the greatest centrality across the molecule is chosen as the centre of the graph and origin for deriving the torsional ordering. As an example, looking at the molecule in figure 5 the atom with the highest value of betweenness centrality is the number 5.

Each rotatable fragment is identified as a set of atoms influenced by the same set of rotatable bonds, called *rotatable influence set*, with respect to the central atom of the molecule. The definition of rotatable influence set for a generic atom a_i is the following:

$$\text{Rotable influece set for } a_i: \quad I^{a_i}_s = E_{\sigma_{a_i,c}} \cap E_R \tag{24}$$

where E_R is the entire set of the rotatable bonds within the molecule, c is the atom with greatest betweenness centrality, while $E_{\sigma_{a_i,c}}$ are the Bonds on the shortest path between a_i and c. Atoms belonging to the same rigid fragment have the same set of rotatable bonds within their shortest path to the central atom.

Finally, contributions appearing in equation (6) are obtained for each rigid fragment, where distances between atoms are calculated only once and respecting conditions 1 and 2. Distances between atoms belonging to the same rigid fragment are fixed, hence not considered.

7. Model development

As mentioned earlier, the complexity of the creation of the HUBO model shown in equation (22) grows with the dimension of the molecule and by its flexibility (rotatable bonds). This brings to two problems. First, the limitation in the number of qubits available on the physical machine. Second, our current approach builds a HUBO formulation for each molecule to be unfolded using a symbolic framework and this requires time to be created. To reduce both the complexity of the problem to be solved and the overhead in the model creation, we studied and applied different approaches that we describe hereafter.

7.1. Distance simplification

The first simplification concerns the calculation of contributions in equation (6), as the number of terms increases quadratically with the number of atoms involved. Hence, instead of measuring the Euclidean distance between each atom a in rigid fragment A and each atom b in rigid fragment B, only distances between different rigid fragments have been considered. In particular, each rigid fragment was identified with the coordinates of the median atom inside the set.

Partial contributions for single fragments are summed up together in the HUBO as in equation (22). Note that such optimization function is a non-linear symbolic expression where symbolic variables representing torsional angles appear in the rotation matrix in the form of trigonometric functions as defined in equation (16).

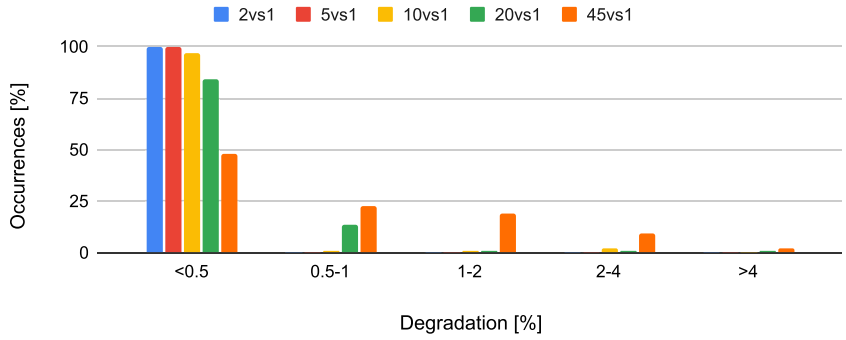

Figure 6. Analysis of the degradation by transitioning to a coarse-grained rotation.

7.2. Coarse-grained rotations

The number of variables needed to express the MU problem in combinatorial optimization form depends on the discretisation of the angles, as shown in equation (20). It is therefore essential to understand how to choose an optimal granularity for the torsionals. In this regard, previous knowledge about the dataset was used to obtain approximate solutions achieving an acceptable quality.

Figure 6 shows the effect of the rotation angle discretization (columns) with respect to the degradation of the molecule volume obtained (x-axis). The vertical axis represents the occurrence in percentage of the molecules included in the target dataset within a certain degradation interval. The degradation level of the volume is measured against the one obtained by applying one degree as rotation interval. The plots shows how for all the coarse grain angles from 2 to 20, the quality of the unfolding is comparable with the one degree case. The surprising result is that also considering a discretization for the rotation angle equal to 45 degrees, the average degradation is around 0.9%, while remaining always below 4.5%.

For this reason, even though we introduced in the problem formulation the possibility of using different angular discretisations, as shown in equation (20), we focuses on the rest of the paper to a granularity of 45 degrees. Thus, the combinatorial optimization would only require eight binary variables for each rotatable bond to identify its angle while keeping the final quality loss acceptably low.

7.3. Threshold approximation

The generation of the HUBO phase is one of the most computationally intensive since we are using a symbolic framework. The challenges that arise are strictly related to the enormous amount of memory used and the unparalleled implementation of the symbolic engine. Hence, a straight derivation of the HUBO becomes unrealistic when increasing problem size.

For this reason, a last simplification was introduced, which consists in eliminating the optimization terms in equation (22) (only those not related to the hard constraints) whose coefficient is strictly less than a certain threshold value, since it was proved to be efficient in [19].

The effect of such threshold approximation (also called *chop*) was to speed up the computation of the HUBOs, as shown in figure 7, and decrease the number of terms in the QUBO function to match the QA hardware specifications and limitations such as the number of qubits (i.e. binary variables) and connectivity (i.e. quadratic terms) of the problem. Our study does not go beyond the eight rotatable bonds as HUBO construction involving ten torsionals exceeded the time limit with the current setup and software.

8. Alternative search strategies

In this section, we presents classical optimization strategies used to compare QA results.

8.1. Random search

Random search [20] algorithm works by picking up randomly the candidate solution from a uniform distribution. A big strength of this method is the high parallelization and the possibility of using Bayesian techniques or probability distributions that are more suitable for a given problem (algorithm 1).

8.2. GeoDock search

The GeoDock search [10] is a greedy algorithm that considers a single torsional at a time and makes a decision locally for each rotatable bond. The algorithm usually does not lead to an optimal solution but it can reach good approximate solutions in a reasonable time.

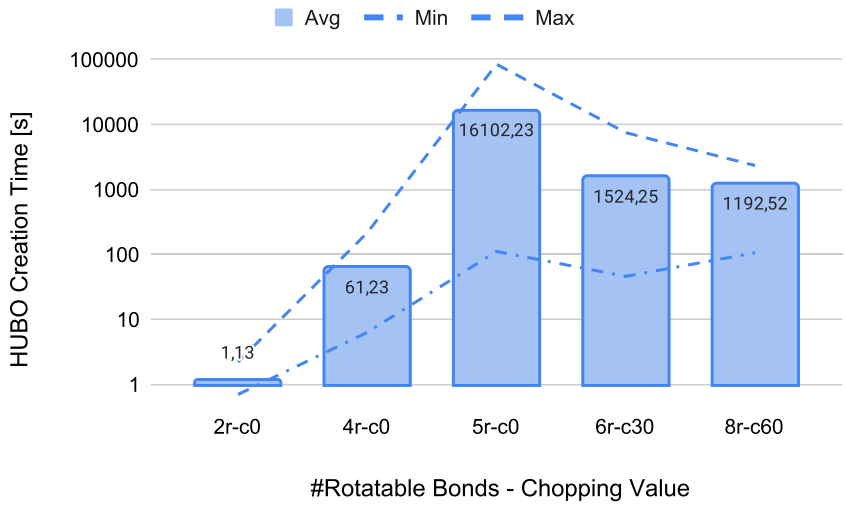

Figure 7. Average creation time for HUBOs. We were able to generate HUBOs without the threshold approximation, up to five torsionals. In the case of six and eight rotatable bonds, the threshold was increase to the vale of 30 and 60 respectively. The configuration *Xr–cY* is the case for molecules with *X* rotatable bonds and a chopping threshold at *Y*.

Algorithm 1. Random search.

Require: $A = Angles, Poly = D_{ab}(\Theta)^2, T = Torsional\ Bonds\ Set$
Ensure: $\gamma_{\max}$: max $D_{ab}(\gamma_{\max})^2$
$|T| = m$
$\Gamma = A_1 \times A_2 \times \cdots \times A_m$ ▷ Cartesian product of the set of angles—m times
while $t < MaxTime$ **do**
$\gamma = pickranduniform(\Gamma)$
Evaluate Poly$(\Theta = \gamma)$ ▷ Evaluate assignments and record the optimum.
end while
return $[\gamma_{\max}, \max Poly]$

Algorithm 2. GeoDock search.

Require: $A = Angles, Poly = D_{ab}(\Theta)^2, T = Torsional\ Bonds\ Set$
Ensure: $a_{\max}$: max $D_{ab}(a_{\max})^2$
$|T| = m$
for t_i, θ_i *in* T_{ordered} **do** ▷ Ordered by betweenness centrality.
for a *in* A **do**
Evaluate Poly$(\theta_i = a)$ ▷ Record intermediate optimums.
end for
end for
return $[\ \underline{a}_{\max}, \max Poly]$

This algorithm (algorithm 2) is the only one that is not employing symbolic computation. Each bond is rotated independently, starting from the most central and internal bond with respect to betweenness centrality. Only the conformation that improves the volume is recorded as a solution. The whole process is repeated a fixed number of times or until a plateau in the improvement is reached.

8.3. Simulated annealing

SA is a meta-heuristic technique, useful in the search of the optimum in large solution spaces (algorithm 3).

Since we can visualize the solution space as points belonging to a graph, SA starts with a random initial guess that is set as the initial best solution ($a_{\max}$), and it considers its evaluation Poly($a_{\max}$). The successive step is to pick a neighbouring solution, and if the neighbour has a lower evaluation, we take it as the current solution. If the neighbour has a higher evaluation, we accept it with a probability written as:

$$P_{\text{accept}} = \min\left(1,\ e^{-\frac{\text{Poly(neigh.)} - \text{Poly}(a_{\max})}{T}}\right) \tag{25}$$

where T is the temperature that starts from a high value and decreases in time according to a *cooling schedule*. Since SA solves the QUBO formulations of our problems, this algorithm was taken as the main comparison against QA.

Algorithm 3. Simulated annealing.

Require: $A = Angles, Poly = D_{ab}(\Theta)^2, T = Torsional\ Bonds\ Set, N_{\max} = Total\ Iterations$
Ensure: $a_{\max}$: max $D_{ab}(a_{\max})^2$
$T \leftarrow T_{\max}$
$a_{\max} = \text{NULL}$
while $T > T_{\min}$ *and* $N < N_{\max}$ **do**
$next = pickNeighbour(T, a_{\max})$
$\Delta E = Poly(next) - Poly(a_{\max})$
$r = randomNumberUniform(0, 1)$
if $\Delta E < 0$ **then**
$a_{\max} = next$ ▷ The neighbour is accepted as the new optimum.
else if $r < e^{\frac{-\Delta E}{k_b * T}}$ **then** ▷ Moving to a neighbour.
$a_{\max} = next$
end if
$T = geometricDecrease(T)$
end while
return [$\underline{a}_{\max}$, max $Poly$]

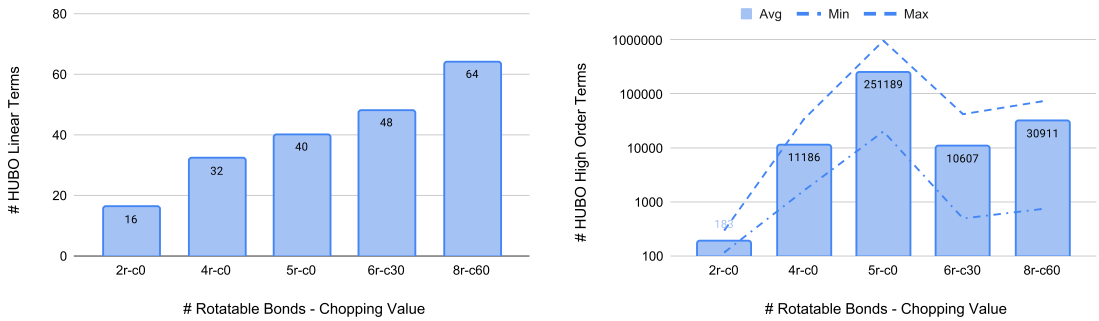

Figure 8. Mean number of linear and non-linear terms in the HUBOs, obtained varying the number of torsionals and value of the approximation threshold. The configuration *Xr–cY* is the case for molecules with *X* rotatable bonds and a chopping threshold at *Y*.

9. Problem complexity and embedding

9.1. Resources estimation

The metrics considered for evaluating the complexity of the problem are the number of linear and high-order terms appearing in the HUBO (plotted in figure 8). The number of linear terms follows the relation *Number of rotatables* × *granularity of rotation*. Recall that a granularity of 45 degrees was selected, i.e. $d = 8$ in equation (20).

The number of non-linear terms grows rapidly going from two to six torsions. However, by increasing the problem size, the threshold approximation allowed us to avoid the exponential growth that these numbers would otherwise follow.

Since the final objective is to be able to embed our problems into a QPU, a second threshold approximation acting on the QUBO has been applied. The choice of the amount of chop is done by a fine-tuning of the threshold, which is lowered until the embedding cannot be performed anymore. In figure 9, we plot the average number of linear and quadratic terms by varying the number of torsional and the value of the approximation threshold. The reported values are only for those molecules which problem we were able to embed into the QPU.

The plots in figure 9 show an increasing trend in the number of QUBO linear terms associated with the same threshold, for an increasing number of rotatables. Linear terms also decrease by increasing the threshold when the number of torsionals is fixed. Quadratic terms have a similar behaviour, except done for the class with eight rotatable where more complex problems seem to peak at a threshold equal to 200. This behaviour is due to the fact that with eight rotatable bonds (but also with six) the threshold value of 100 was not sufficient to embed the problem in the QPU for most of the cases except those that were simpler.

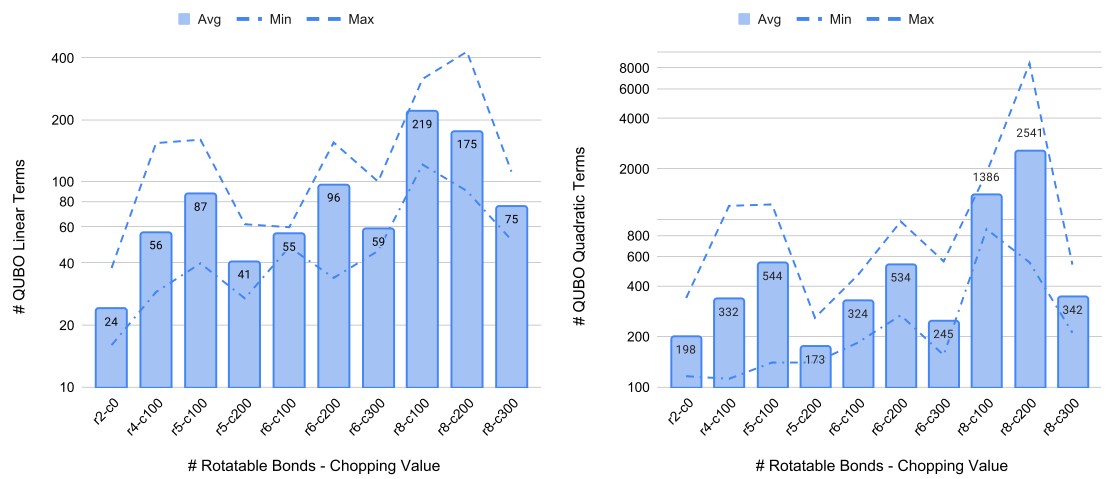

Figure 9. Average number of linear and quadratic terms in the QUBOs, obtained varying the number of torsionals and value of the approximation threshold. The configuration *Xr–cY* is the case for molecules with *X* rotatable bonds and a chopping threshold at *Y*.

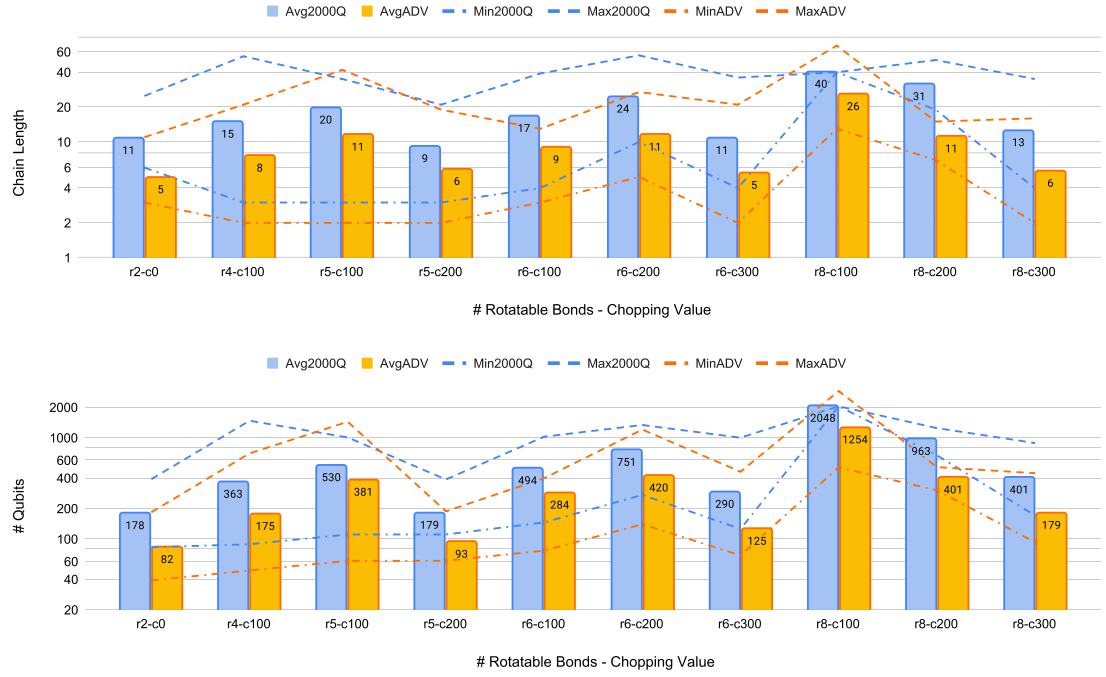

Figure 10. Plots depicting the variation of average chain length and average number of qubits in the embeddings by increasing number of rotatables and value of the approximating threshold (also called chop). The configuration *Xr–cY* is the case for molecules with *X* rotatable bonds and a chopping threshold at *Y*.

9.2. D-wave embedding

In the embedding phase, a heuristic algorithm tries to find the optimal matching between problem resources and physical D-wave hardware. Several embeddings are possible, so the one that uses the least physical qubits out of a few trials is employed. It is implicit in this choice that such embedding will also have shortest chains on average, i.e. chains of physical qubits representing a single QUBO variable.

The embeddings shown in figure 10 are highly representative of the capabilities of each QPU topology. On average, the chains obtained with a Chimera topology are 2.09 times longer than those found using the Pegasus topology, which therefore returns 52% shorter chains.

Regarding the number of qubits, Pegasus is again the winner as it enables embedding with 1.39 up to 2.4 times fewer qubits with respect to Chimera. On average, advantage is 1.96 times more efficient than 2000Q, or 51% less expensive in terms of qubits; this will be reflected in the quality of results. There are only two results slightly diverging from our expectations. The columns regarding the class 6–100 are shorter than

expected because, due to the complexity of the problem, only simpler instances were embedded. Hence the average is biased towards a lower value.

Another important detail shown in the figure involves the class 8–100 where both the average chain length and the average number of qubits are saturated to the maximum value for the 2000Q Chimera topology. This happened because the device could not embed all the possible molecules for that class, therefore radically increasing the average.

10. Experimental results

A short remark should be made on the way these algorithms run. Regarding the QA set-ups, since the QUBO constant A_{const} modulating the strength of the hard constraint must be tuned, ten runs for each value assigned to A_{const} are performed. A_{const} is evaluated as the maximum coefficient in the optimization contribution multiplied by an increasing factor. Since we were interested in studying the solution quality by reducing as much as possible the required QPU time, each run is performing 10 000 cycles of forward anneals, with an annealing schedule of 1 μs. SA is performed on the same QUBOs, with the possibility of choosing the number of epochs and the function which is responsible for the temperature decrease. In our case, we used 500 annealing epochs and a geometrical decrease function.

Since the execution of the quantum and SA are repeated many times in the attempt of finding the best solution, also the other classical techniques are executed until their run-time does not exceed a time limit that was set from the beginning. Random search divides the solution space in N parallel processes and as long as it does not run out of time, it executes and records the maximal value found. The Geodock search is trying an incremental approach, which consists in performing the rotation on an incremental number of bonds.

10.1. Volume gain in time

Figure 11 show the average trends in volume gain obtained by each algorithm on a fixed time window, increasing rotatable bonds. On the Y-axis, the maximum volume reached by every algorithm at each second is displayed. The plots show the algorithmic behaviour up to 100 s since this is the interval where greatest changes in volume appear. The maximum volume gain shown in the plots is measured against the best solution found by all the methods adopted in the experimental campaign. While for small problems it represents the global optimum, for larger ones this is not guaranteed.

The first thing worth noticing is that SA gave the best overall performance as it was able to provide good solutions falling inside the $[90\%, 100\%]$ volume gain range with a run-time that never exceeded the 15 s.

It is also possible to observe that advantage is the best among D-wave devices. For an increasing number of rotatable bonds, volume achieved by 2000Q quickly dies off, while the results of advantage remain competitive for a good number of torsionals.

When two torsionals are considered, the MU problem looks like an easy task and all the approaches are able to recover the optimal solution (only GeoDock gets close but fails to find optimal). Random search, SA, and the advantage QA can reach the 100% volume gain within the first 5 s of runtime.

Increasing the problem size to four torsionals we can foresee a deviation of the trends: SA is again the fastest to reach a plateau while the quantum annealers perform better than random search in the first 5 s. After that, random search presents a disruptive improvement that comes with a delay due to overheads in the memory access. A promising result is obtained after 25 s, where advantage is able to find the optimal solution among the methods. This indicates that QA could be a competitive approach when solving medium-small sized problems.

It is possible to see another reversal of the trends when the number of torsions rises to 6, in which SA and random search establish themselves as the most reliable solvers, followed by the advantage quantum annealer. This behaviour is confirmed at eight rotatable bonds, where performances of 2000Q become very similar to that of the GeoDock greedy algorithm.

10.2. Comparison between SA and QA

Comparing classical algorithms with quantum annealers in terms of absolute time is not fair, since the quantum devices have run-times that are comprising of programing time, readout time, and delay time, besides the time spent in actually performing the annealing.

Therefore, a more suitable metric for comparing annealing solvers is given by the time to solution (TTS),

$$\text{TTS} = \frac{\text{total execution time}}{\text{occurrence of the best solution}}. \tag{26}$$

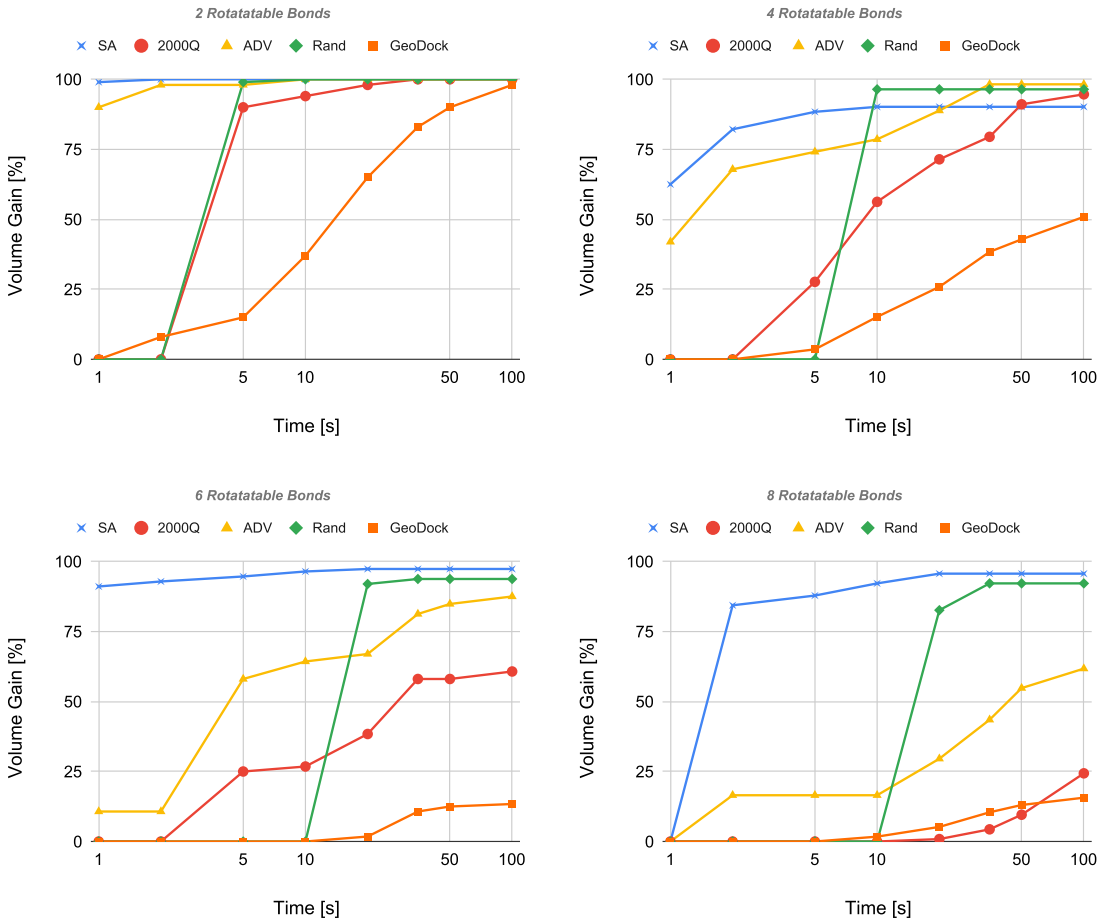

Figure 11. Comparison of average volume gained on a time window of 100 s for two, four, six and eight rotatable bonds.

Table 1. Comparison on the TTS measured in milliseconds, mean and range (Min–Max values), while increasing number of rotatable bonds.

	SA		2000Q		ADV	
Rot. Bonds	Mean (ms)	Range (ms)	Mean (ms)	Range (ms)	Mean (ms)	Range (ms)
2	0.01	[0.00–0.01]	0.87	[0.02–8.29]	0.16	[0.01–1.22]
4	4.14	[0.28–33.89]	65.48	[0.29–395.22]	12.82	[0.61–39.52]
5	5.50	[1.35–13.92]	63.00	[13.26–132.63]	34.51	[2.97–61.60]
6	10.38	[2.30–26.34]	204.46	[66.31–663.09]	50.43	[30.00–106.98]
8	12.94	[4.04–23.26]	503.94	[66.31–1326.26]	43.36	[33.39–71.34]

Table 2. Comparison on the volumes, mean value and range (Min–Max), while increasing number of rotatable bonds.

	SA		2000Q		ADV	
Rot. Bonds	Mean	Range	Mean	Range	Mean	Range
2	1.00	[1.00–1.00]	1.00	[0.97–1.00]	1.00	[0.97–1.00]
4	0.92	[0.00–1.00]	0.86	[0.01–1.00]	0.97	[0.86–1.00]
5	0.99	[0.96–1.00]	0.84	[0.49–0.93]	0.95	[0.90–0.98]
6	0.98	[0.94–1.00]	0.62	[0.00–0.95]	0.88	[0.59–1.00]
8	0.93	[0.82–1.00]	0.20	[0.00–0.66]	0.66	[0.40–0.91]

The total execution time can be calculated as the number of anneals multiplied by the total access time for a single anneal. The occurrences of the best solution simply count the number of times the best solution is found in the annealing. The TTS metric can be interpreted as the inverse of the probability of finding the optimal configuration in a unit time interval. Moreover, the TTS is telling us how long we have to wait on average before the annealer outputs the best solution. For this reason, better solvers have low values of TTS.

Table 3. Normalized volume gain over TTS for each algorithm, at different number of rotatable bonds.

Rot. Bonds	SA	2000Q	ADV
2	2.001 031	2.000 690	2.005 038
4	0.003 190	0.022 822	0.025 046
5	0.002 582	0.023 171	0.009 112
6	0.001 354	0.005 269	0.005 776
8	0.001 031	0.000 690	0.005 038

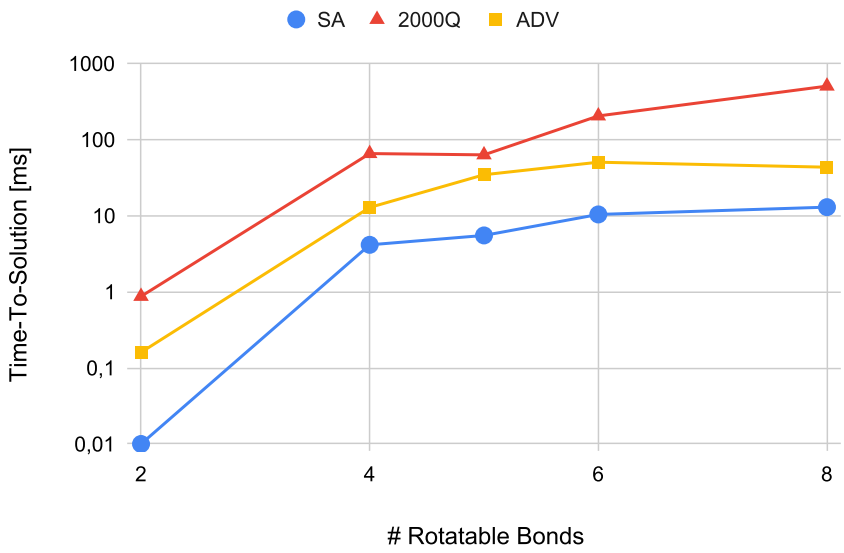

Figure 12. A comparison between SA, advantage and 2000Q on TTSs measured in milliseconds (ms), for increasing number of rotatable bonds. Numerical data can be found in table 1.

Figure 12 shows that the TTS is very high for 2000Q, followed by advantage, while SA has the best TTS. SA has a TTS several times smaller than advantage, which means that its convergence to the optimum is much faster than in QA.

Regarding the volumes achieved, figure 13 represent the degradation in the final volume gain as the problem complexity increases. The volumes have a common starting point since all the three algorithms actually reach the maximum steadily and completely for problems involving two rotatable bonds. As seen in the previous section, advantage works well up to four torsions, SA has an inflection on four bonds but then recovers and finishes as the best in absolute terms; it has a volume which is 4.65 times better than the one found by 2000Q at eight bonds, and 1.4 times better than advantage.

The last concept worth mentioning is the measure of how much volume in percentage can be gained per unit of TTS. Practically, it is computed as the normalized ratio between the volume gain V_g and TTS, as described by the following equation:

$$\mathrm{Norm}\left(\frac{V_g}{\mathrm{TTS}}\right) = \frac{V_g/\mathrm{TTS}}{(V_g/\mathrm{TTS})_{\max} - (V_g/\mathrm{TTS})_{\min}} \tag{27}$$

where V_g/TTS is the volume gain divided by TTS, while $(V_g/\mathrm{TTS})_{\max}$ and $(V_g/\mathrm{TTS})_{\min}$ are respectively, the maximum and the minimum values obtained by V_g/TTS varying the number of rotatables.

A negative slope in the plot of figure 14 can be interpreted as an increasing TTS required to achieve the same volume when scaling up to a more difficult problem. A higher negative slope means faster worsening of the algorithm with the increasing difficulty, while a smaller slope is a synonym for stability. Therefore, trends running towards the bottom of the plot belong to algorithms that are defective. The SA line falls very fast from two to four torsionals, and then starts becoming stable. The same happens to advantage which slows down this negative acceleration before. 2000Q never stops decreasing its quality, if not for a special case between four and five torsionals.

By checking the slopes $\frac{\Delta(V_g/\mathrm{TTS})}{\Delta(N_{\mathrm{Rot}})}$ when increasing the number of torsionals, we can understand how much the algorithms are worsening adding more degrees of freedom. For this purpose, slopes are calculated in the plateau region, i.e. starting from four up to eight torsionals. On average, we observed that the

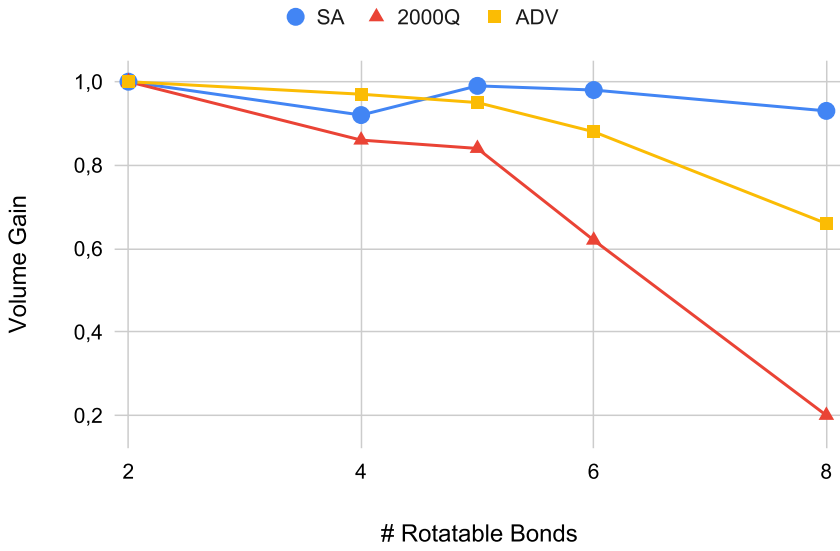

Figure 13. A comparison between SA, advantage and 2000Q on volumes. Numerical data can be found in table 2.

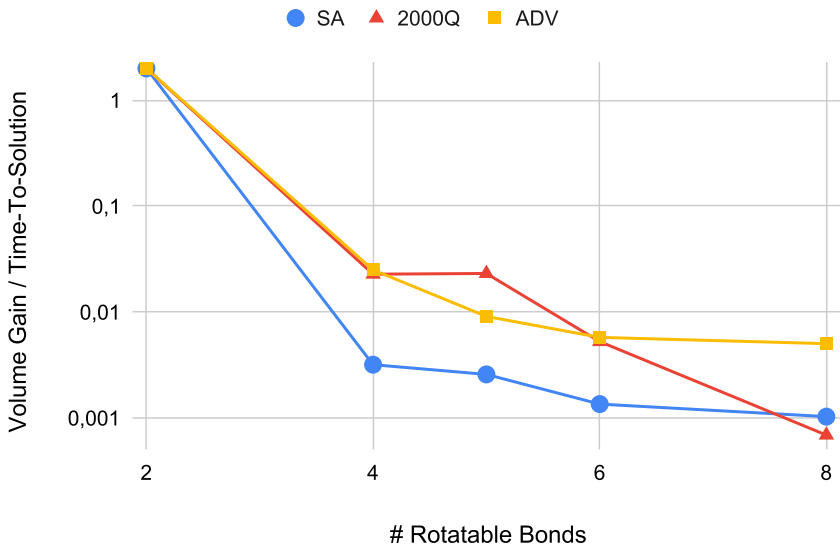

Figure 14. A comparison between SA, advantage and 2000Q on the normalised decreasing quality of the volumes per TTS. Numerical data can be found in table 3.

smallest growth in scaling of the slopes belongs to advantage, with 1.36, whereas SA presents a higher 1.61, meanwhile 2000Q has an average of 6.02.

Although the results obtained with advantage may not be the winners in absolute terms, the fact that advantage has good performances with respect to SA according to this metric is very promising. This result reflects how much QA is still immature but also how far from saturation the margin of growth for QA really is. Improvements in the next generation of QPU hardware could definitely affect both TTS and the quality of sampling in a positive way.

11. Conclusions

In this work, we explored the possibility of using a quantum annealer to support the drug discovery process within the in-silico virtual screening phase. In particular, we studied a new method regarding the process of MU, which is the first step in geometric molecular docking techniques. This phase aims at finding the molecule's configuration that maximizes its volume, or equivalently, that maximizes the internal distances of the atoms that compose the molecule. We proposed our quantum MU model to execute it on D-wave's latest hardware, advantage and 2000Q. The usage of the quantum annealers has the objective of

understanding the capabilities of these QA devices and discovering if it is possible to improve the quality and the throughput of the ligand expansion in the state-of-art molecular docking methods.

The model is constructed starting from the identification of the rotatable bonds which are the parameters of the problem. Once their discrete rotations are rewritten via one-hot encoding thanks to the introduction of binary variables, the total sum of internal atomic distances can be expressed as a HUBO. The HUBO is then transformed into a QUBO after three steps are applied: (i) distance simplification, (ii) coarse-grained rotations, and (iii) threshold approximation. These three approximations enabled us to reduce the complexity of the model with the aim of embedding and running complex instances on the QPUs.

We compared the performances of advantage and 2000Q, with three other optimization methods, which are parallel random optimization, SA, and a GeoDock greedy algorithm. The outcomes of the experiments show that advantage has good performances on medium-small problems while classical techniques are better than quantum approaches when problem size increases. Interestingly, the results obtained by the quantum annealer, both while running on advantage and 2000Q, are better than the GeoDock greedy approach. In terms of the evolution of the quantum annealers, embedding our problems on advantage, compared to 2000Q, was on average 51% less expensive in terms of qubits and with chains 52% shorter. Advantage significantly outperforms 2000Q in terms of TTS and volume gain when increasing the number of torsionals. Advantage also presents the best scaling of the normalized volume gain over time-to-solution when compared to both 2000Q and SA; this last result is one of the most promising as D-wave devices are only bound to get better.

Future work could consider alternative encoding strategies for the discrete rotations such as the recently proposed 'domain-wall' encoding scheme [13–15], to reduce the number of resources required by the quadratic formulation. Moreover, additional developments could consist of the introduction of new dynamic thresholds approximation, which may lead to smaller embeddings but still of high quality. Finally, the introduction of solution refinement techniques like reverse annealing and pausing may increase the probability of finding close to optimal results [21, 22], provided that a sufficient QPU access time is available. A more difficult but also feasible and interesting task could be extending the QA approach to the whole docking process.

Finally, we are confident that this study will shed further light on the applications of QC, thus enriching knowledge on topics that are becoming central in computational sciences. The HUBOs and the code for solving the problem using the different algorithms reported in the paper have been made available at https://gitlab.hpc.cineca.it/rmengoni/quantum-molecular-unfolding.

Acknowledgments

This work has been supported in terms of resources by CINECA-ISCRA initiative, project QA4SMU—HP10C60OYF, and by D-WAVE under the global response to COVID-19 action.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Appendix A

A.1. Machines and libraries

The platform used for running the classical algorithms is based on NUMA nodes, featuring two Intel(R) Xeon(R) CPU E5-2630 v3 CPUs (@2.40GHz), Virtualisation VT-x, caches L1d, L1i cache of 32K. L2 and L3 caches are, respectively, 256K and 20480K. RAM memory of 125 Gb and 114 Gb of SWAP memory. Operative system used is Ubuntu 18.04.5 LTS Bionic. The code was written in python 3.9.1, compiled with gcc 9.2.0. The parallelization was actuated with the usage of mpi4py 3.0.3, compiled with MPICH 3.3.2.

Since the classical probabilistic algorithms and random search, could be run in embarrassingly parallel mode, mpi4py [23] was used. For each problem instance, the number of processes used for each run was 32.

Other libraries of support are: Pandas 1.2.1, Numpy 1.19.5, symbolic calculations were performed thanks to Sympy 1.7.1 [24].

A.2. Rotation matrix $R(\theta)$

Here in the following we are reporting the rotation matrix used to modify the position of each atom within a rotatable fragment once known the related rotatable bond. This matrix represents the base for the

formulation reported in the paper within sections 3–5. The matrix is generated considering the angle θ of the rotation and the coordinates of the two atoms connected by the considered rotatable bond ($a' = \langle x', y', z' \rangle$ and $a'' = \langle x'', y'', z'' \rangle$). The matrix is build considering the rotation happening around an arbitrary line passing through the rotatable bond, and having a direction vector from a' to a'' [25].

Let us define $d_x = x'' - x'$, $d_y = y'' - y'$, $d_z = z'' - z'$ as the x, y and z components of the direction vector from a' to a'' respectively, while $l = \sqrt{(d_x^2 + d_y^2 + d_z^2)}$ as the length of the vector. Moreover for space reason, let us use a compressed version for $\sin(\theta) = s_\theta$ and $\cos(\theta) = c_\theta$. We can now write the rotation matrix $R(\theta)$ as follows:

$$\begin{bmatrix} \frac{(d_x^2 + (d_y^2 + d_z^2)c_\theta)}{l^2} & \frac{(d_x d_y(1-c_\theta) - d_z l s_\theta)}{l^2} & \frac{(d_x d_z(1-c_\theta) + d_y l s_\theta)}{l^2} & \frac{((x'(d_y^2 + d_z^2) - d_x(y'd_y + z'd_z))(1-c_\theta) + (y'd_z - z'd_y)l s_\theta)}{l^2} \\ \frac{(d_x d_y(1-c_\theta) + d_z l s_\theta)}{l^2} & \frac{(d_y^2 + (d_x^2 + d_z^2)c_\theta)}{l^2} & \frac{(d_y d_z(1-c_\theta) - d_x l s_\theta)}{l^2} & \frac{((y'(d_x^2 + d_z^2) - d_y(x'd_x + z'd_z))(1-c_\theta) + (z'd_x - x'd_z)l s_\theta)}{l^2} \\ \frac{(d_x d_z(1-c_\theta) - d_y l s_\theta)}{l^2} & \frac{(d_y d_z(1-c_\theta) + d_x l s_\theta)}{l^2} & \frac{(d_z^2 + (d_x^2 + d_y^2)c_\theta)}{l^2} & \frac{((z'(d_x^2 + d_y^2) - d_z((x'd_x + y'd_y))(1-c_\theta) + (x'd_y - y'd_x)l s_\theta)}{l^2} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ORCID iDs

Kevin Mato https://orcid.org/0000-0003-2073-5226
Riccardo Mengoni https://orcid.org/0000-0003-2717-7298
Daniele Ottaviani https://orcid.org/0000-0002-4002-8345
Gianluca Palermo https://orcid.org/0000-0001-7955-8012

References

[1] Lyne P D 2002 Structure-based virtual screening: an overview *Drug Discov. Today* **7** 1047–55
[2] Walters W P, Stahl M T and Murcko M A 1998 Virtual screening-an overview *Drug Discov. Today* **3** 160–78
[3] Meng X-Y, Zhang H-X, Mezei M and Cui M 2011 Molecular docking: a powerful approach for structure-based drug discovery *Curr. Comput.-Aided Drug Des.* **7** 146–57
[4] Nielsen M A and Chuang I L 2010 *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge: Cambridge University Press)
[5] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79
[6] McGeoch C C 2014 *Adiabatic Quantum Computation and Quantum Annealing: Theory and Practice* (*Synthesis Lectures on Quantum Computing* vol 5) (San Rafael: Morgan & Claypool Publishers)
[7] Babej T and Fingerhuth M 2018 Coarse-grained lattice protein folding on a quantum annealer (arXiv:1811.00713)
[8] Marchand D J J, Noori M, Roberts A, Rosenberg G, Woods B, Yildiz U, Coons M, Devore D and Margl P 2019 A variable neighbourhood descent heuristic for conformational search using a quantum annealer *Sci. Rep.* **9** 13708
[9] Kellenberger E, Rodrigo J, Muller P and Rognan D 2004 Comparative evaluation of eight docking tools for docking and virtual screening accuracy *Proteins* **57** 225–42
[10] Gadioli D *et al* 2020 Tunable approximations to control time-to-solution in an HPC molecular docking mini-app *J. Supercomput.* **77** 841–69
[11] Kirkpatrick S, Gelatt C D and Vecchi M P 1983 Optimization by simulated annealing *Science* **220** 671–80
[12] Mengoni R, Ottaviani D and Iorio P 2020 Breaking RSA security with a low noise D-wave 2000Q quantum annealer: computational times, limitations and prospects (arXiv:2005.02268)
[13] Chancellor N 2019 Domain wall encoding of discrete variables for quantum annealing and QAOA *Quantum Sci. Technol.* **4** 045004
[14] Chen J, Stollenwerk T and Chancellor N 2021 Performance of domain-wall encoding for quantum annealing *IEEE Trans. Quantum Eng.* **2** 1–14
[15] Dridi R, Berwald J and Chancellor N 2021 Understanding domain-wall encoding theoretically and experimentally (arXiv:2108.12004)
[16] Wang R, Fang X, Lu Y, Yang C-Y and Wang S 2005 The PDBbind database: methodologies and updates *J. Med. Chem.* **48** 4111–9
[17] Tripos Inc. 2005 SYBYL 7.1—MOL2 user guide http://chemyang.ccnu.edu.cn/ccb/server/AIMMS/mol2.pdf
[18] Freeman L C 1977 A set of measures of centrality based on betweenness *Sociometry* **40** 35
[19] Sax I, Feld S, Zielinski S, Gabor T, Linnhoff-Popien C and Mauerer W 2020 Approximate approximation on a quantum annealer *Proc. 17th ACM Int. Conf. Computing Frontiers* (New York: ACM)
[20] James B and Bengio Y 2012 Random search for hyper-parameter optimization *J. Mach. Learn. Res.* **13** 281–305
[21] Marshall J, Venturelli D, Hen I and Rieffel E G 2019 Power of pausing: advancing understanding of thermalization in experimental quantum annealers *Phys. Rev. Appl.* **11** 044083
[22] Ikeda K, Nakamura Y, Humble T S and Humble 2019 Application of quantum annealing to nurse scheduling problem *Sci. Rep.* **9** 12837
[23] Dalcin L D, Paz R R, Kler P A and Cosimo A 2011 Parallel distributed computing using python *Adv. Water Resour.* **34** 1124–39
[24] Meurer A *et al* 2017 SymPy: symbolic computing in python *PeerJ Comput. Sci.* **3** e103
[25] Van Verth J M and Bishop L M 2015 *Essential Mathematics for Games and Interactive Applications* 3rd edn (London: Taylor and Francis)