UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Fisica e Astronomia "Galileo Galilei"

Corso di Laurea in Fisica

Tesi di Laurea

Solving the Closest Vector Problem via Quantum Annealing on the D-Wave® machine

Risoluzione del Closest Vector Problem utilizzando il Quantum Annealing sul computer D-Wave®

Relatore

Dr. Ilaria Siloi

Laureando

Nicolò Montagner

Correlatori

Prof. Simone Montangero

Marco Tesoro

Anno Accademico 2024/2025

Abstract

Il Closest Vector Problem (CVP) è un noto problema di ottimizzazione combinatoria con numerose applicazioni nella matematica computazionale e nella crittografia basata sui reticoli. Dato un reticolo, il CVP richiede di trovare il punto appartenente al reticolo più vicino a un punto target che invece non vi appartiene. Questo problema è dimostrato essere NP-Hard, il che implica che non esistono algoritmi classici in grado di trovare la soluzione esatta in modo efficiente per reticoli di dimensione arbitrariamente grande.

In questa tesi, introduciamo dapprima il problema matematico e l'algoritmo classico più efficiente attualmente conosciuto per trovare soluzioni approssimate anche per reticoli di dimensioni molto grandi. Successivamente, presentiamo un metodo per mappare un generico CVP in un'Hamiltoniana di spin glass, con l'obiettivo di migliorare le prestazioni dell'algoritmo classico. Nello specifico, ci concentriamo sui reticoli definiti nel contesto matematico sviluppato da Schnorr per la crittografia basata sui reticoli. Infine, dopo aver illustrato la teoria alla base della tecnica del Quantum Annealing, utilizziamo il quantum annealer di D-Wave® per trovare lo stato fondamentale dell'Hamiltoniana, che corrisponde alla soluzione del CVP.

I risultati ottenuti sono poi comparati con quelli ottenuti utilizzando algoritmi classici, evidenziando le differenze in termini di efficienza e prestazioni.

Contents

Introduction

Lattice problems are fundamental NP-hard problems in modern mathematics and computer science. The hardness in the search for the exact solution is a distinctive feature that has been exploited in various contexts. On one hand, solutions to the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP) are building blocks for encryption protocols in Lattice-based Post Quantum cryptography [1]. On the other hand, the CVP construction lies at the core of Schnorr's factoring algorithm [2] for the RSA encryption protocol. Schnorr's algorithm maps the factoring problem into an optimization problem defined on a set of lattices. Then it constructs the factors of the RSA key from the combinations of approximate solutions to a set of CVPs.

The CVP is an optimization problem that searches for the closest lattice point to a given target point that does not belong to the lattice. Babai's algorithm is currently the best-known method to find an approximate solution to the CVP in polynomial time. However, the accuracy of this approximation decreases with increasing lattice rank. Following a recent proposal by *Yan et al.* [3], we encode other possible solutions in the energy spectrum of a spin glass Hamiltonian constructed from the Babai's solution. This construction aims to achieve a set of additional points close to the target. We implement this mapping to find better approximate solutions to CVP in the case of Schnorr's lattices. Interestingly, this spin glass Hamiltonian is also amenable to be solved on quantum computers.

Quantum computing is a highly active research field crossing many areas of modern research. Several physical systems are currently being investigated as possible quantum computing platforms. Among the different technologies now available, quantum annealers stand out, as they employ the largest number of qubits achieved to date. Quantum annealing technology leverages the adiabatic theorem of quantum mechanics to reach the ground state of a particular Hamiltonian. For this reason, quantum annealing is especially well-suited for solving optimization problems properly mapped into spin glass Hamiltonians.

In this Thesis, we study the mapping of the CVP into a spin glass Hamiltonian extracting the explicit formulation of the coupling terms used for the mapping. We develop a Python code to calculate these coupling terms and perform quantum annealing on Schnorr's CVPs using the D-Wave® quantum hardware *Advantage_system 4.1*.

After an overview of classical CVP solutions provided by Babai's algorithm for Schnorr's lattices in Chapter 1, we present the explicit calculations of the coupling terms obtained from the expansion of the cost function, see Chapter 2. In this Chapter, we also introduce the method for mapping the CVP into a spin glass Hamiltonian. Additionally, we present exact results for small-rank lattices obtained via a brute-force method, which is subsequently used to validate the results from quantum annealing. In Chapter 3, we explain the methods used to perform quantum annealing on the D-Wave® quantum hardware.S pecifically, we examine the principles behind quantum annealing processes, the parameters of the quantum annealer and the limitations arising in the procedure of embedding the problem Hamiltonian in the physical qubits on the hardware. Chapter 4 is devoted to our main results: we perform over $2,800,000$ simulations on the *Advantage_system 4.1* and report the results obtained, focusing on improved vectors solutions compared to Babai's one. In the final Section, we report the post-processing of the results to check if the obtained CVP solutions sampled from D-Wave® are suitable for factoring semiprimes as described in Schnorr's sieving method [2].

Chapter 1

The Closest Vector Problem

In this Chapter, we examine the Closest Vector Problem (CVP) and present the best known classical algorithm to obtain an approximated solution in polynomial time: the Babai's nearest plane algorithm. We then outline a strategy to enhance Babai's solution via a mapping onto a spin glass Hamiltonian. Finally we exactly study this mapping for small lattices using a brute-force approach.

1.1 Mathematical definition of the CVP

Let us consider a n-rank lattice embedded in a m-dimensional space. For our purposes, we restrict the discussion to integer lattices. An integer lattice is a subset of the vector space $\mathbb{Z}^m$. Let $\{\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}\}$ be a linearly independent set of vector in $\mathbb{Z}^m$. The integer lattice is generated by a lattice basis $\{\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}\}$

$$L = \left\{ \sum_{i=1}^{n} l_i \mathbf{b_i} : l_i \in \mathbb{Z} \right\} \tag{1.1}$$

as an integer linear combinations of the $\mathbf{b_i}$. The lattice rank is n and the lattice dimension is m. If $n = m$ then L is said to be a full rank lattice [1].

The Closest Vector Problem (CVP) is defined as follows: for a given lattice L defined by a basis matrix $B = [\mathbf{b_1}, ..., \mathbf{b_n}]$, where columns are basis vectors $\{\boldsymbol{b}_j\} \in \mathbb{Z}^m$ for $j = 1, \ldots, n$, and a target vector $\mathbf{t} \in \mathbb{Q}^m$ find $\mathbf{v} \in L$ such that $||\mathbf{t} - \mathbf{v}||$ is minimal. The CVP problem is demontrates to be NP-hard, then it is at least as difficult to solve as the hardest problems in the NP class, the set of computational problems verifiable in polynomial time [4]. Algorithms to find solutions to NP-hard problems are typically not efficient. Therefore, we are interested in finding approximate solutions using the most efficient classical algorithm, Babai's nearest plane algorithm [5].

To solve the CVP, we aim to start from the best possible basis for L. Finding a "nice" basis is fundamental to solve efficiently many lattice problems, and in general a good basis choice allows fewer calculations. Ideally, we look for a lattice basis, composed of vectors that are both short and nearly orthogonal. There are established methods in the literature to transform a given basis into a "nice" one [1]. One established method is the Lenstra-Lenstra-Lovász (LLL) lattice basis reduction algorithm, a polynomial-time algorithm that builds on a rank-2 reduced basis, as defined by Lagrange and Gauss. The Lagrange-Gauss reduction algorithm ensures that the output is a 2×2 matrix basis, whose vectors are proven to be as short as possible [1]. The LLL algorithm generalizes the Lagrange–Gauss algorithm for lattices with a lattice rank greater than 2, and leverages the Gram–Schmidt orthogonalization. It is important to note that the Gram–Schmidt process is generally not practical for lattices because the coefficients $\mu_{i,j}$ typically do not lie in $\mathbb{Z}$, and therefore, the resulting vectors are not usually elements of the lattice. However, the LLL algorithm utilizes the Gram–Schmidt vectors to assess the quality of the lattice basis, while ensuring that the linear combinations used to update the lattice vectors remain within $\mathbb{Z}$. A basis obtained through this algorithm is called LLL-reduced basis [1].

The CVP has several important applications both in mathematics and cryptography, such as in Lattice-Based and Post-Quantum cryptography, where its NP-hardness ensures that even powerful adversaries, including those using quantum algorithms, cannot easily break the encryption. Beyond

Algorithm 1 LLL Reduction Algorithm with Euclidean norm

Input: A basis $B = \{\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}\}$ with $\mathbf{b_j} \in \mathbb{Z}^m$, a reduction parameter δ (typically $\frac{3}{4}$)
Output: LLL reduced basis $B = \{\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}\}$

Compute the Gram-Schmidt orthogonalization $\mathbf{b_1^*}, \mathbf{b_2^*}, \ldots, \mathbf{b_n^*}$ and coefficients $\mu_{k,j}$ for $1 \leq j \leq k \leq n$
Compute $B_k = \langle \mathbf{b_k^*}, \mathbf{b_k^*} \rangle = ||\mathbf{b_k}||^2$ for $1 \leq k \leq n$
$k = 2$
while $k \leq n$ **do**
    for $j = k - 1$ downto 1 **do**
        Let $q_i = \lfloor \mu_{k,j} \rceil$ and set $\mathbf{b_k} = \mathbf{b_k} - q_i \mathbf{b_j}$
        Update the value $\mu_{k,j}$ for $1 \leq j < k$
    end for
    if $B_k \geq (\delta - \mu_{k,k-1}^2) B_{k-1}$ **then**
        $k = k + 1$
    else
        Swap $\mathbf{b_k}$ and $\mathbf{b_{k-1}}$
        Update the values $\mathbf{b_k^*}$, $\mathbf{b_{k-1}^*}$, B_k, B_{k-1}, $\mu_{k-1,j}$ and $\mu_{k,j}$ for $1 \leq j < k$ and $\mu_{j,k}$,$\mu_{i,k-1}$ for $k < j \leq n$
        $k = \max(k - 1, 2)$
    end if
end while

that, CVPs have been proposed also in the context of factorization algorithms, such as Schnorr's lattice-based sieving [6]. Schnorr's algorithm encodes the factorization of a semiprime N into its prime factors trough the solution of a set of CVPs constructed as follows [2].

In this Thesis we will study the class of CVP problems defined in the context of Schnorr's lattice based sieving. We begin by selecting a basis of prime numbers $\{p_1, p_2, \ldots, p_n\}$ referred to as factoring basis. Then we construct Schnorr's matrix basis:

$$B_{n,f,c} = \begin{pmatrix} f(1) & 0 & 0 & \cdots & 0 \\ 0 & f(2) & 0 & \cdots & 0 \\ 0 & 0 & f(3) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & f(n) \\ \lfloor 10^c \ln p_1 \rceil & \lfloor 10^c \ln p_2 \rceil & \lfloor 10^c \ln p_3 \rceil & \cdots & \lfloor 10^c \ln p_n \rceil \end{pmatrix} \tag{1.2}$$

where the diagonal terms are given by a permutation f of the set $\{\lfloor \frac{1}{2} \rceil, \lfloor \frac{2}{2} \rceil, \lfloor \frac{3}{2} \rceil, ..., \lfloor \frac{n}{2} \rceil\}$, and $\lfloor \cdot \rceil$ is the nearest integer operation. The last row terms are the nearest integer to $10^c \ln p_i$ where p_i is the i-th element of the factoring basis; c is an integer parameter called *lattice precision parameter* that give us the precision of our calculation. The target of this CVP is depends on N, the semiprime that we aim to factorize. For our purpose it is defined as follows:

$$\mathbf{t} = (0, 0, 0, \ldots, 10^c \ln N). \tag{1.3}$$

1.2 Babai's nearest plane algorithm

Babai introduced a method to inductively find a lattice vector close to the target $\mathbf{t} \in \mathbb{Z}^n$ within a given approximation factor [1]. This algorithm uses the LLL-reduced lattice basis to identify the nearest plane to the target vector. The algorithm outputs a vector $\mathbf{v} \in L$ that does not necessarily minimize $||\mathbf{t} - \mathbf{v}||$, but it is proved that if the lattice basis is LLL-reduced matrix, then $||\mathbf{t} - \mathbf{v}||$ remains within an exponential factor of the minimal possible distance. Specifically, it has been shown that the nearest plane algorithm provides a vector $\mathbf{v}$ such that $||\mathbf{v} - \mathbf{t}|| < 2^{n/2} ||\mathbf{u} - \mathbf{t}||$ for any $\mathbf{u} \in L$ [1].

Algorithm 2 Babai nearest plane algorithm

Input: $\{\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}\}$, $\mathbf{w}$
Output: $\mathbf{v}$

Compute Gram-Schmidt basis $\mathbf{b_1^*}, \ldots, \mathbf{b_n^*}$
Set $\mathbf{w_n} = \mathbf{w}$
for i $= n$ downto 1 **do**
    Compute $l_i = \langle \mathbf{w_i}, \mathbf{b_i^*} \rangle / \langle \mathbf{b_i^*}, \mathbf{b_i^*} \rangle$
    Set $\mathbf{y_i} = \lfloor l_i \rceil \mathbf{b_i}$
    Set $\mathbf{w_{i-1}} = \mathbf{w_i} - (l_i - \lfloor l_i \rceil)\mathbf{b_i^*} - \lfloor l_i \rceil \mathbf{b_i}$
end for
return $\mathbf{v} = \mathbf{y_1} + \ldots + \mathbf{y_n}$

The idea behind Babai's algorithm is to find a vector $\mathbf{y} \in L$ such that the distance from $\mathbf{t}$ to the plane $U + \mathbf{y}$ is minimized, where $U = \text{span}\{\mathbf{b_1}, ..., \mathbf{b_{n-1}}\}$. Next, we define $\mathbf{t'}$ to be the orthogonal projection of $\mathbf{t}$ onto the plane $U + \mathbf{y}$ and set $\mathbf{t''} = \mathbf{t'} - \mathbf{y} \in U$. It is important to note that if $\mathbf{t} \notin L$, then $\mathbf{t''} \notin U$. By solving inductively the lower-dimensional closest vector problem for $\mathbf{t''}$ in L', we can find the solution to the original CVP as $\mathbf{v} = \mathbf{y} - \mathbf{y'}$. In Figure 1.1, we give a graphical illustration of how Babai's algorithm works.

The first step of our work is to study the solution provided by the Babai's algorithm for Schnorr's lattices when the target is built from a semiprime N. We begin with a geometrical analysis of Schnorr's lattices by varying the lattice precision parameter c and the dimension of the lattice n. Specifically, we calculate the volume and the orthogonality defect, namely the deviation from orthogonality of the LLL-reduced lattices, obtained by LLL-reduction from Schnorr's lattices. The volume of the fundamental parallelepiped is computed as $V = \sqrt{BB^T}$, and the orthogonality defect is defined as $o_{def} = \left(\prod_{i=1}^{n+1} |\mathbf{b_i}| \right) / V$. The orthogonality defect is calculated such that it equals 1 when the basis is orthogonal, whereas when the basis is not orthogonal, its value will be greater than 1. The orthogonality defect will increase with increasing non orthogonality of the basis. For the resolution of the CVP, we are seeking bases that are as orthogonal as possible, thus with an orthogonality defect approaching 1. We report the results of this calculations in Figure 1.2.

The goal of this Thesis is to is to study the conditions that improve Babai's solution. Specifically, in the next section, we explore corrections, i.e. roundings, in the directions discarded by Babai's method, utilizing strategic adjustments based on the basis vectors selected by the algorithm.

1.3 Beyond Babai's solution

It is well-established that Babai's algorithm provides an approximate solution to the CVP. Therefore, we aim to enhance this solution by exploring other roundings, then by examining the opposite direction of the basis vectors with respect to the integer approximation chosen by the algorithm. Following [3], we formulate a cost function to minimize. Starting from Babai's solution, we introduce a correction to the vector, which may adjust the solution by -1, 0, or +1 in the direction of each basis vector. The cost function is defined as follows:

$$\mathcal{F} = ||\mathbf{t} - \mathbf{v_{new}}|| = \left|\left| \mathbf{t} - \mathbf{v} - \sum_{j=1}^{n} k_j \mathbf{b_j} \right|\right|, \tag{1.4}$$

where $\mathbf{v_{new}}$ represents the new closest vector we aim to obtain, $\mathbf{v}$ is the solution provided by Babai's algorithm, $\mathbf{b_j}$ is the j-th vector of the lattice basis, and k_j is a variable that can assume 3 values: -1, 0 or 1.

We conduct a preliminary study to determine whether improving on Babai's solution requires Babai's solution as a starting point or a similar improvement can be obtained by using a generic

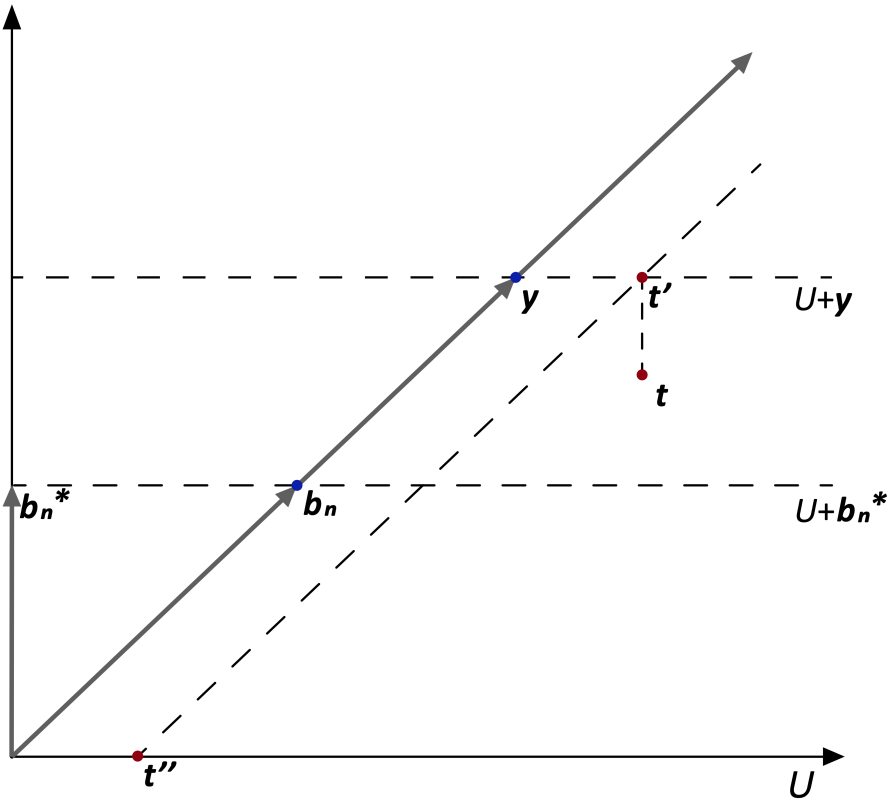


Figure 1.1: *Illustration of the Babai's nearest plane method.* Graphic representation of Babai's algorithm step by step for one iteration of the algorithm procedure. Next we set $\mathbf{t}'' = \mathbf{t}' - \mathbf{y}$ to be the target vector of the new problem.

projection of the target vector $\mathbf{t}$ onto the basis B of Schnorr's lattice. Using a computationally inefficient classical method, see Section 2.2, we compute the distance between the solutions provided in both cases and calculate the approximation ratio defined as:

$$\gamma_{Babai's} = \frac{|\mathbf{v} - \mathbf{t}|^2}{|\mathbf{v_{new}} - \mathbf{t}|^2} \qquad \gamma_{Projection} = \frac{|\mathbf{p} - \mathbf{t}|^2}{|\mathbf{v_{new}} - \mathbf{t}|^2}, \tag{1.5}$$

where $\mathbf{p}$ is the projection of the target vector $\mathbf{t}$ onto the lattice basis B, and $\mathbf{v_{new}}$ is the vector provided by a "brute-force" method for the minimization of the cost function $\mathcal{H}$. We calculate the approximation ratio across up to 40 random permutations of the diagonal matrix elements. We then plot the result for two different semiprimes, $N_1 = 27537500323$ with bit-length $\ell = 35$, and $N_2 = 784656869198858791459$ with bit-length $\ell = 70$. The bit-length is the length of N in binary basis, it is an important parameter that is useful to compare the number that we use with RSA keys (Section 4.2). We report the results in Figure 1.3. We observe that Babai's solution provides a good starting point to find CVPs solutions. Instead, by projecting the target vector onto the lattice basis we obtained an approximation ratio that grow exponentially with the lattice rank n. As expected is convenient to construct the cost function starting from classical Babai's solution.

Specifically, for large n values, this approach offers only an improvement over Babai's solution although it does not guarantee an exact solution to the Closest Vector Problem (CVP). However, the focus of this Thesis is leveraging this mapping to use the quantum annealing as implemented on D-Wave® machines to improve the solution provided by the classical algorithm for sufficiently large lattice dimensions.

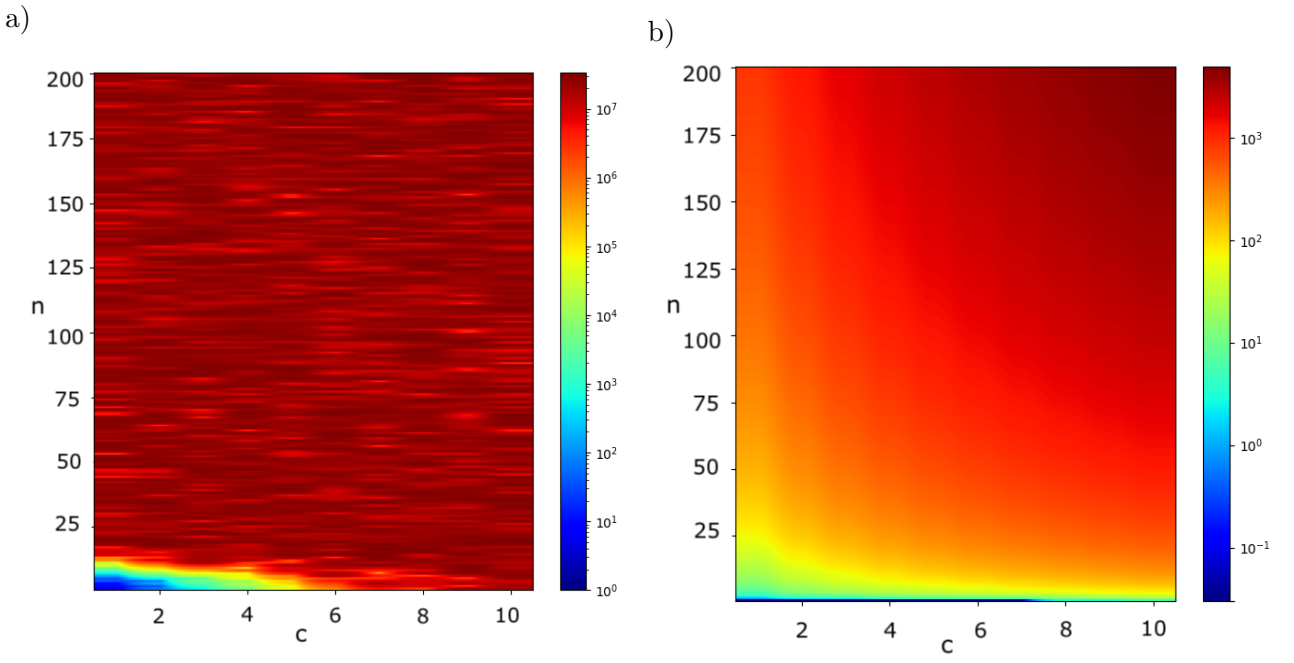


Figure 1.2: *Volume and orthogonality defect.* **a)** The elementary volume of the lattice construct from LLL-reduced basis computed from Schnorr's lattices as a function of n and c. The obtained results are consistent with the lattice theory: by increasing both n and c the volume increases; lattices with the smaller volume are the ones with smaller lattice rank and lower lattice precision c [1]. **b)** The orthogonality defect of a lattice constructed from LLL-reduced basis computed from Schnorr's lattice as a function of n and c. For larger lattice with a higher lattice precision the orthogonality defect is higher. For both figure **a** and **b** the color scale is logarithmic.

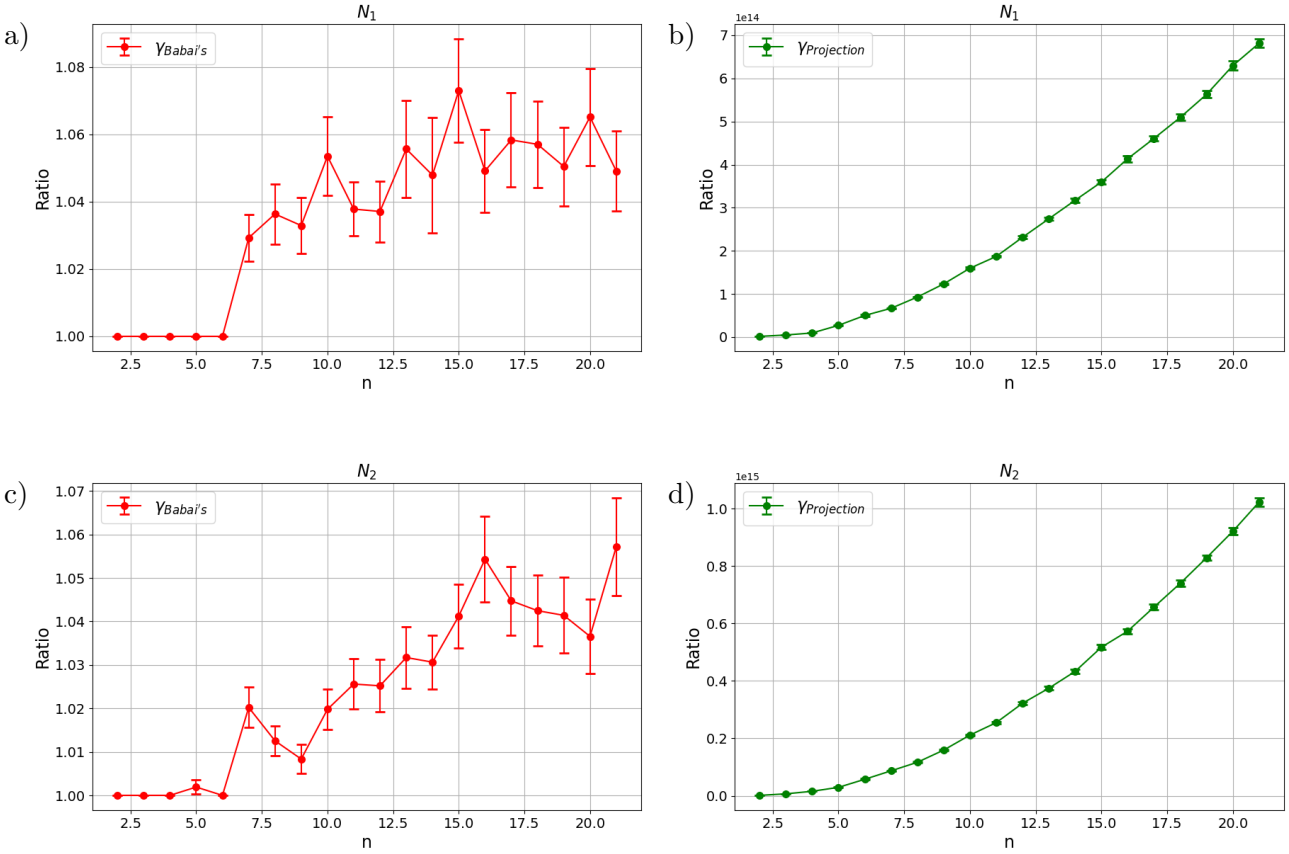


Figure 1.3: *Approximation ratio for Babai's solution and projection of the target on the lattice basis.* The approximation ratio as a function of the lattice rank n for $N_1 = 27537500323$ for Babai's solution **a)** and for target projection **b)**. The approximation ratio as a function of the lattice rank n for $N_2 = 784656869198858791459$ for Babai's solution **c)** and for target projection **d)**. In both cases we average the approximation ratio over 40 different random diagonal of Schnorr's lattice basis 1.2.

Chapter 2

Spin glass Hamiltonian formulation of the CVP

The spin glass are a mathematical models, originally developed to describe certain ferromagnetic materials that exhibit complex phase diagrams. These models describe a system of interacting spins featuring all-to-all pairwise couplings [7]. Over time, spin glass model have found applications in a wide variety of fields, i.e. quantum computing and optimization, as spin-1 / 2 can be easily mapped into qubits [7]. In this Chapter, we formulate the minimization of the cost function in Eq. (1.4) as a ground-state search problem of a corresponding spin glass Hamiltonian. We obtain this mapping for the CVP by representing the single binary rounding variable $k_j \in \{-1, 0, +1\}$ as classical spin-1 / 2, and subsequently promoting these to quantum spin operators.

2.1 Encoding the CVP in a spin glass Hamiltonian

Ising spin glasses are known to be NP-hard problems for classical computers [8]. The vast majority of combinatorial optimization problems, from the traveling salesman problem to graph coloring, which are highly significant not only from a theoretical perspective but also in practical applications, can be mapped onto suitable spin glass models [8]. The CVP is no exception, and we now present a spin glass mapping built around Babai's solution, starting directly from the cost function defined in Eq. (1.4).

A classical Ising-like spin glass model is described by the following Hamiltonian, which is a quadratic function of a set of spins $\{s_1, s_2, \ldots, s_n\}$:

$$\mathcal{H}_{Ising} = \sum_{j=1}^{n} h_j s_j + \sum_{j=1}^{n} \sum_{l=j+1}^{n} J_{l,j} s_j s_l \tag{2.1}$$

where $s_j \in \{-1, 1\} \to \{\downarrow, \uparrow\}$ for all $j \in \{1, \ldots, n\}$ and n is the number of spin variables in the spin glass system. In the spin glass Hamiltonian in Eq. (2.1), h_j the external longitudinal magnetic field acting onto the j-th spin, also called bias, and $J_{l,j}$ is the interaction that couple the l-th spin with the j-th one. Without loss of generality, we neglect any constant term in the Hamiltonian in Eq. (2.1) as it does not change the physics of the system, merely representing an energy shift.

The CVP can be formulated as a spin glass by mapping the cost function defined in Eq. (1.4) onto a spin glass Hamiltonian of the form described in Eq. (2.1). The minimum of the cost function Eq. (1.4) is the same of the function $\mathcal{H}$ defined as follows:

$$\mathcal{H} = \sum_{k=1}^{n+1} \left[t_k - b_k - \sum_{j=1}^{n} k_j d_{j,k} \right]^2 \tag{2.2}$$

where the subscript k stand for the k-th component of the vector. Finding the minimum distance corresponds to finding the minimum of the squared distance.

We aim to allow the variable k_j to assume values 0 or ± 1 depending on the approximation per-

formed by Babai's algorithm. In particular k_j can be 1 or 0 if the algorithm makes a downward approximation, and 0 or -1 if the algorithm makes an upward approximation. To determine the type of approximation made by the algorithm, we examine the coefficient vector $\mu = (\mu_1, ..., \mu_n)$ extract from Babai's algorithm and compute the sign function of the difference $\mu_j - c_j$ where $c_j = \lfloor \mu_j \rceil$ is μ_j nearest integer. Consequently we build k_j as:

$$k_j = \frac{1}{2}\text{sign}(\mu_j - c_j)(1 - s_j) \quad \Rightarrow \quad \begin{aligned} k_j &= \frac{1 - s_j}{2} = \begin{cases} 0 & \\ 1 & \end{cases} \quad \text{if } \mu_j - c_j > 0 \\ k_j &= \frac{s_j - 1}{2} = \begin{cases} 0 & \\ -1 & \end{cases} \quad \text{if } \mu_j - c_j < 0 \end{aligned} \tag{2.3}$$

By substituting the variable k_j with its explicit formula, the cost function in Eq. (2.2) can be rewritten as follows:

$$\mathcal{H} = \sum_{k=1}^{n+1} \left[t_k - b_k - \sum_{j=1}^{n} \frac{1}{2}\text{sign}(\mu_j - c_j)(1 - s_j) d_{j,k} \right]^2 \tag{2.4}$$

We can now encode the cost function in Eq. (2.4) into a quantum Ising spin glass Hamiltonian. This involves promoting each classical spins $s_j \in \{-1, 1\}$ to quantum spin operators $\hat{\sigma}_j^z$ aligned with the z-axis, thus expressing the cost function as a quantum many-body Hamiltonian:

$$\hat{H} = \sum_{k=1}^{n+1} \left[t_k - b_k - \sum_{j=1}^{n} \frac{1}{2}\text{sign}(\mu_j - c_j)(1 - \hat{\sigma}_j^z) d_{j,k} \right]^2 \tag{2.5}$$

In the next Section, we explicit provide the computation of the coupling terms $\{h_j\}_{j=1}^n$ and $\{J_{lj}\}_{j\,>\,l}^n$, starting from Eq. (2.5).

2.2 Coupling calculation

In this Section, we present the explicit calculation of the couplings that describe the spin model used to encode the CVP we aim to solve. What follows is obtained by explicitly developing the square parenThesis in Eq. (2.5), leading to the general form for the magnetic field h_j on site j and the coupling J_{lj} on sites l and j. These formulas are crucial for the implementation of a Python code that automates the creation of the spin glass for a generic Schnorr's CVP instance. The couplings thus obtained will serve as input for the D-Wave quantum annealer to find CVP solutions.

Starting from Eq. (2.5) we have

$$\hat{H} = \sum_{k=1}^{n+1} \left[(t_k - b_k)^2 + \left(\sum_{j=1}^{n} \frac{1}{2}\text{sign}(\mu_j - c_j)(1 - \hat{\sigma}_j^z) d_{j,k} \right)^2 + (b_k - t_k) \sum_{j=1}^{n} \text{sign}(\mu_j - c_j) d_{j,k} (1 - \hat{\sigma}_j^z) \right] \tag{2.6}$$

Expanding the second term we obtain:

$$\begin{aligned} \left(\sum_{j=1}^{n} \frac{1}{2}\text{sign}(\mu_j - c_j)(1 - \hat{\sigma}_j^z) d_{j,k} \right)^2 &= \left(\sum_{j=1}^{n} \frac{1}{2}\text{sign}(\mu_j - c_j)(1 - \hat{\sigma}_j^z) d_{j,k} \right) \left(\sum_{l=1}^{n} \frac{1}{2}\text{sign}(\mu_l - c_l)(1 - \hat{\sigma}_l^z) d_{l,k} \right) \\ &= \frac{1}{4} \sum_{j=1}^{n} \sum_{l=1}^{n} \text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l) d_{j,k} d_{l,k} (1 - \hat{\sigma}_j^z - \hat{\sigma}_l^z + \hat{\sigma}_j^z \hat{\sigma}_l^z) \end{aligned} \tag{2.7}$$

We can observe that:

$$-\frac{1}{4}\sum_{j=1}^{n}\sum_{l=1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k}\hat{\sigma}_j^z = -\frac{1}{4}\sum_{j=1}^{n}\sum_{l=1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k}\hat{\sigma}_l^z \quad (2.8)$$

Therefore

$$\begin{aligned}\hat{H} = \sum_{k=1}^{n+1}\Bigg[(t_k - b_k)^2 + \frac{1}{4}\sum_{j=1}^{n}\sum_{l=1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k} - (t-b)\sum_{j=1}^{n}\text{sign}(\mu_j - c_j)d_{j,k} \\ + \sum_{j=1}^{n}\hat{\sigma}_j\left\{\text{sign}(\mu_j - c_j)d_{j,k} - \frac{1}{2}\sum_{l=1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k}\right\} \\ + \frac{1}{4}\sum_{j=1}^{n}\sum_{l=1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k}\hat{\sigma}_j^z\hat{\sigma}_l^k\Bigg] \quad (2.9)\end{aligned}$$

Observing that $\hat{\sigma}_j\hat{\sigma}_j = (\hat{\sigma}_j)^2 = 1$, $\text{sign}(\mu_j - c_j)\text{sign}(\mu_j - c_j) = 1$ and that $d_j d_l = d_l d_j$ we can rewrite the last term as:

$$= \frac{1}{4}\sum_{j=1}^{n}d_{j,k}^2 + \frac{1}{2}\sum_{l=j+1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k}\hat{\sigma}_j^z\hat{\sigma}_l^z \quad (2.10)$$

Finally, we get

$$\begin{aligned}\hat{H} = \sum_{k=1}^{n+1}\Bigg[(t_k - b_k)^2 + \frac{1}{4}\sum_{j=1}^{n}\sum_{l=1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k} - (t-b)\sum_{j=1}^{n}\text{sign}(\mu_j - c_j)d_{j,k} \\ + \frac{1}{4}\sum_{j=1}^{n}d_{j,k}^2 + \sum_{j=1}^{n}\hat{\sigma}_j\left\{\text{sign}(\mu_j - c_j)d_{j,k} - \frac{1}{2}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k}\right\} \\ + \frac{1}{2}\sum_{j=1}^{n}\sum_{l=j+1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k}\hat{\sigma}_j^z\hat{\sigma}_l^z\Bigg] \quad (2.11)\end{aligned}$$

where we can recognize the generic form of a quantum spin glass Hamiltonian

$$\mathcal{H}_{Ising} = C + \sum_{j=1}^{n}h_j\hat{\sigma}_j^z + \sum_{j=1}^{n}\sum_{l=j+1}^{n}J_{l,j}\hat{\sigma}_j^z\hat{\sigma}_l^z \quad (2.12)$$

and consequently extract the general expressions for the model couplings, including the constant (irrelevant) term C:

$$\begin{aligned}C = \sum_{k=1}^{n+1}\left[(t_k - b_k)^2 + \frac{1}{4}\sum_{j=1}^{n}\sum_{l=1}^{n}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k} - (t-b)\sum_{j=1}^{n}\text{sign}(\mu_j - c_j)d_{j,k} + \frac{1}{4}\sum_{j=1}^{n}d_{j,k}^2\right] \\ h_j = \sum_{k=1}^{n+1}\left[\text{sign}(\mu_j - c_j)d_{j,k} - \frac{1}{2}\text{sign}(\mu_j - c_j)d_{j,k}\sum_{l=1}^{n}\text{sign}(\mu_l - c_l)d_{l,k}\right] \\ J_{l,j} = \sum_{k=1}^{n+1}\left[\frac{1}{2}\text{sign}(\mu_j - c_j)\text{sign}(\mu_l - c_l)d_{j,k}d_{l,k}\hat{\sigma}_j^z\hat{\sigma}_l^z\right] \quad (2.13)\end{aligned}$$

2.3 Brute-force results

The spin glass couplings reported in Eq. (2.13) fully determined the CVP Hamiltonian encoding once a problem instance is given. To verify the validity and the correctness of our calculations, we compare the exact minimum of the CVP cost function in Eq. (1.4) for small lattice rank n with the exact ground-state of the CVP Hamiltonian computed via exact diagonalization of Eq. (2.5).

The minimum of the CVP cost function can be computed through a brute-force approach, which involves evaluating the cost function for every possible classical configuration $s = \{s_1, s_2, ..., s_{n-1}, s_n\}$ where $s_j \in \{-1, 1\}$. The configuration yielding the minimum cost value is selected as the optimal solution representing the closest vector to target constructed around Babai's solution. The time-complexity of this method scales as $O(2^n)$, where n is the lattice rank. Consequently, it is highly inefficient for large lattices and thus unsuitable for improving Babai's algorithm.

As for the exact solution of the quantum CVP Hamiltonian in Eq. (2.5), we perform an exact diagonalization of the Hamiltonian matrix in Eq. (2.5). To build the Hamiltonian matrix we start by the definition given in Eq. (2.12). Considering the operator $\hat{\sigma}_j^z$ as the z Pauli matrix $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ acting on the j-th qubit of the system, therefore we can describe $\hat{\sigma}_j^z$ acting on the full Hilbert space of the system as the 2×2 identity matrix $\mathbb{I}_{(2\times2)}$ acting on the other $n-1$ qubits. In general we can construct it as follows:

$$\hat{\sigma}_j^z = \mathbb{I}_{(2\times2)} \otimes \mathbb{I}_{(2\times2)} \otimes \ldots \otimes \underbrace{\hat{\sigma}^z}_{j\text{-th site}} \otimes \ldots \otimes \mathbb{I}_{(2\times2)} \otimes \mathbb{I}_{(2\times2)} \tag{2.14}$$

Starting from this definition and using the coupling terms provided in Eq. (2.13), we can construct the $\hat{H}$ matrix. Typically, we aim to diagonalize this matrix to find the eigenvalues. However, in this case, the matrix is already diagonal because it originates from a classical optimization problem where all terms in the Hamiltonian commute with each other. Therefore, to find the ground state, it is sufficient to identify the lowest value on the Hamiltonian diagonal. Similarly, the entire spectrum can be found by sorting all the diagonal elements. Since the Hamiltonian is diagonal, its eigenvectors are vectors of a canonical basis of size n.

Although the diagonalization algorithm operates on a polynomial time scale, the construction of the matrix is highly inefficient, as it requires storing a $2^n \times 2^n$ matrix. Despite its inefficiency, exact diagonalization provides not only the ground state of the Hamiltonian—the lowest energy level—but also all the excited states and their respective degeneracies. It is therefore a useful method to perform preliminary studies for small lattices and check the validity of the spin glass mapping.

We use a Python script to compute solutions for $n < 25$, which allows us to perform preliminary calculations (reported in Figure 1.3) of the approximation ratio γ Eq. (1.5) and validate results obtained from the quantum-inspired approach.

In both methods discussed, as expected, the algorithmic complexity is exponential in the size of the lattice n. For this reason, in the next chapter, we will employ the technique of quantum annealing and utilize a quantum platform to scale the size of the lattices and solve more complex CVPs.

Chapter 3

Quantum annealing

The fundamental idea behind quantum computing is to harness quantum phenomena, such as superposition and entanglement, to tackle computationally challenging problems [4]. Among the various potential applications of quantum computing in tackling complex computational challenges, one particularly promising area is its ability to solve optimization problems beyond the efficient reach of classical computers.

In gate-based quantum computers, classical binary bits are replaced with qubits, i.e., quantum systems with two levels that can exist in any superposition of the two classical binary states. In gate-based quantum computing, the computation is achieved by applying gates that perform unitary transformations on a single or a set of qubits. This unitary evolution is carried out through various methods, which depend on the nature of the qubits involved [4].

Adiabatic quantum annealers work on a different principle, based on the adiabatic theorem of quantum mechanics [9]. At the beginning of the protocol, qubits are prepared in the ground state of a simple, easily solvable initial Hamiltonian. Then, the system is evolved slowly until it reaches the final state encoding the desired solution [10].

From quantum mechanics, we know that the Hamiltonian operator $\hat{H}$ is the observable corresponding to the total energy of a quantum system. Our goal is to construct a time-dependent Hamiltonian in which the time-dependent component varies sufficiently slowly, and the gap between the eigenvalues is large enough to satisfy the constraints of the adiabatic theorem. The solutions to the time-dependent Schrödinger equation are expressed in terms of the eigenvalues of the instantaneous Hamiltonian

$$\hat{H}(t)|\psi_a(t)\rangle = E_a(t)|\psi_a(t)\rangle \tag{3.1}$$

where $E_a(t)$ is the energy associated with the state $|\psi_a(t)\rangle$. If $H(t)$ varies slowly in time, a system initially in a non-degenerate state $|\psi_a(t = t_0)\rangle$ with energy $E_a(t = t_0)$ will evolve into the corresponding state $|\psi_a(t)\rangle$ with energy $E_a(t)$ at a later time $t > t_0$ without making any transitions between energy levels. The annealing time required is inversely proportional to the energy gap between the eigenvalues. The larger the energy gap, the faster the transition can occur [11].

3.1 How to run quantum annealing on the D-Wave® machine

In this Thesis, our goal is to apply the principles of adiabatic evolution and quantum annealing to solve the CVP as described by the spin glass Hamiltonian in Eq. (2.5). We will now briefly explain how to utilize the D-Wave® machine to accomplish this and outline several important considerations for analyzing the results produced by this technology.

The D-Wave® *Advantage_system 4.1* is a quantum computer that allows us to perform quantum annealing with more than 5000 qubits. The D-Wave® system contains a Quantum Processing Unit (QPU) that must be kept at a temperature near absolute zero and isolated from the surrounding environment to behave quantum mechanically. The system achieves these requirements as follows [12]:

- Cryogenic temperatures, achieved using a closed-loop cryogenic dilution refrigerator system. The QPU operates at temperatures below 20 mK.

- Shielding from electromagnetic interference, achieved using a radio frequency (RF)–shielded enclosure and a magnetic shielding subsystem.

The D-Wave® QPU is a lattice of tiny metal loops, each of which is a qubit or a coupler. Below temperatures of 9.2K, these loops become superconductors and exhibit quantum-mechanical effects [12].

To perform the annealing process and find a solution to the CVP spin glass Hamiltonian, the QPU is prepared in an easy-to-prepare many-qubit state, which is the ground state of an initial Hamiltonian $\hat{H}_I$. After that, magnetic fields are applied to facilitate the transition between this initial state and the final state as close as possible to the ground state of a Hamiltonian $\hat{H}_F$ encoding the optimization of interest. The quantum evolution implemented by the D-Wave® machine is controlled by the following time-dependent Hamiltonian

$$\hat{H}(t) = A(t)\hat{H}_I + B(t)\hat{H}_F, \tag{3.2}$$

where $A(0) \gg B(0)$ and, at the final time $t_f > 0$, $A(t_f) \ll B(t_f)$.

At the end of the annealing procedure, the system is measured to determine its state configuration. Since the final Hamiltonian $\hat{H}_F$ represents a classical Ising model with only commuting operators $\hat{\sigma}_z$ (the z-component of the Pauli matrix), the solution can be directly obtained as the state of individual qubits in the computational basis [9]. It is important to note that when measuring a quantum state, the resulting classical bit-string in the computational basis is produced non-deterministically, with each outcome's probability determined by its overlap with the prepared quantum state before measurement. Rather than performing a single annealing run, multiple runs are executed, with the system measured at the end of each run and the QPU reinitialized before the next. This approach generates a sample set that reflects the probability distribution of specific states measured after the annealing process.

Assuming the system evolves sufficiently slowly, it will remain in its ground state throughout the entire annealing process, following the adiabatic theorem. Consequently, if the evolution is adiabatic, the final state should ideally represent the solution of the target Hamiltonian $\hat{H}_F$. In this context, the annealing procedure involves gradually transforming the system from the initial quantum Hamiltonian $\hat{H}_I$ to the classical Hamiltonian of an Ising-like spin system.

The choice of $\hat{H}_I$ is critical: it is designed to ensurer that it does not commute with the Ising Hamiltonian. As a result, there is no common basis of eigenstates, preventing the system from getting stuck in the eigenstates of $\hat{H}_F$ [9]. In our specific case, the initial Hamiltonian consists in applying a transverse magnetic field to each qubit, resulting in a so-called transverse field (or driver) Hamiltonian:

$$\hat{H}_I = -\frac{1}{2}\sum_{j=1}^{n} \hat{\sigma}_j^x \tag{3.3}$$

where $\hat{\sigma}_j^x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ being the x-Pauli matrix at site j [9]. The time-dependent Hamiltonian implemented on the D-Wave® to solve the CVP is as follows:

$$\hat{H} = -A(t)\sum_{j=1}^{n} \hat{\sigma}_j^x + B(t)\left(\sum_{j=1}^{n} h_j\hat{\sigma}_j^z + \sum_{l>j} J_{l,j}\hat{\sigma}_l^z\hat{\sigma}_j^z\right) \tag{3.4}$$

where the couplings in the classical part are those defined in Eqs. (2.13) in Section 2.2. The calculation of these couplings can be programmed into a computer code and executed in polynomial time.

As explained before, for each CVP problem instance we perform multiple runs to generate a distribution of solutions, sampling from low-energy states of the problem Hamiltonian. This repeated sampling allows to gather a statistical set of outcomes, which can then be analyzed. When studying the sampling set obtained by the QPU, we must consider two important factors:

- The D-Wave® machine is not a perfectly isolated quantum system, therefore there could be interactions with the environment causing decoherence disrupting the quantum states, and reducing the overall accuracy of the quantum annealing process;

- During the sampling phase, errors may occur as a result of the measurement procedure.

a)

b)

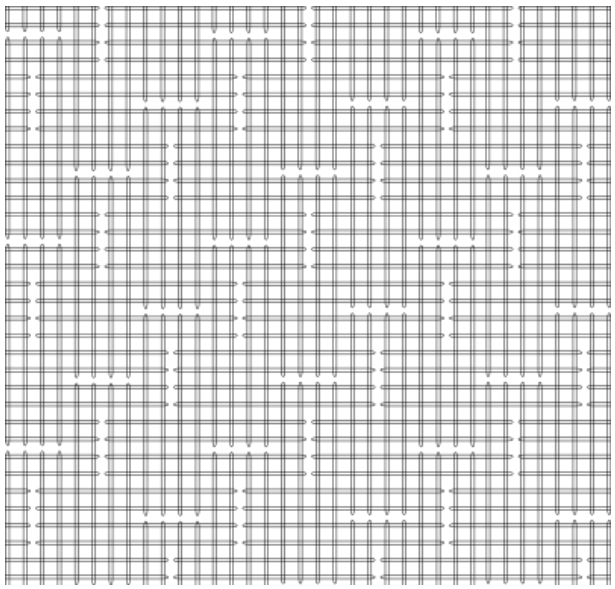


Figure 3.1: **a)** *D-Wave® Advantage system.* Photo of D-Wave® Advantage system hardware [12].
b) *A cropped view of the Pegasus topology with qubits represented as horizontal and vertical loops.* This graphic shows approximately three rows of 12 vertical qubits and three columns of 12 horizontal qubits for a total of 72 qubits, 36 vertical and 36 horizontal. [13]

Given the presence of noise and potential measurement errors, it is essential to run the QPU many times to obtain a robust sample set that accurately reflects the energy landscape of the problem.

3.2 The D-Wave® quantum computer

The hardware used for our simulations is the D-Wave® quantum computer *Advantage_system 4.1* Figure 3.1 a), a quantum annealer equipped with 5627 superconducting qubits that operate at an extremely low temperature of 15.4 ± 0.1 mK. This ultra-low temperature is necessary to achieve a superconducting state, which is essential for maintaining the quantum properties of the qubits. Over the effective qubits, the system use more than 35,000 couplers to let the qubits interact. To build both qubits and couplers, it uses over 1,000,000 Josephson junctions [12].

Additionally, D-Wave® provides the *Ocean* software suite [14], which allows for remote access to QPUs and allows users to formulate and solve their optimization problems.

We use one of the most advanced QPU architecture available: the *Pegasus* Figure 3.1 b) topology. In this topology, each qubit is coupled to 15 other qubits. The architecture consists of a square lattice of unit cells, with each cell containing 24 qubits. The Advantage QPU employed in our computations features a 16×16 grid of this unit cells, yielding a total of $24 \times 16 \times (16-1) = 5760$ qubits. Due to the densely packed spatial arrangement of qubits and the physical manufacturing process, it is expected that some qubits may exhibit suboptimal performance. D-Wave® estimates that approximately 5% of the qubits in the Pegasus topology may have imperfections [11].

The *Ocean* software provides the ability to calibrate several parameters. In Table 3.1, we report the default setting values for the *DWaveSampler*, the configurable sampler provided by *Ocean*. It is important to note that some parameters are dependent on others. In this Thesis, we do not discuss every single parameter in detail. However, comprehensive information is available on the D-Wave® website [15].

Parameters	**Default values**	**Range**
anneal_offsets	0	depend on the qubit
annealing_schedule	$20\mu s$	$[[0, 0.5\mu s], [1, 2000\mu s]]$
annealing_time	$20\mu s$	$[0.5\mu s, 2000\mu s]$
answer_mode	histogram	
auto_scale	True	
fast_anneal	False	
flux_biases	0	depend on the qubit
flux_drift_compensation	True	
h_gain_schedule	$[[0, 1], [t_{final}, 1]]$	$[[0, -4.0], [1, 4.0]]$
initial_state	$(\|1\rangle + \|-1\rangle) / \sqrt{2}$	$[\|0\rangle, \|1\rangle]$
max_answers	Total number of distinct answers	
num_reads	1	$[1, 10000]$
programming_thermalization	$1000.0\mu s$	$[0.0, 10000.0]$
readout_thermalization	$0.0\mu s$	$[0.0, 10000.0]$
reduce_intersample_correlation	False	
reinitialize_state	True	

Table 3.1: *Sampler parameter.* In this table we report all the settable parameters of the D-Wave® *Advantage_system 4.1* with relative default values and the relative range [15].

In this Thesis, we focus on configuring a few specific parameters while keeping the others at their default settings. One of the most important parameters is *auto_scale*. This parameter automatically adjusts the biases and the coupling terms, rescaling the values of h_j and $J_{l,j}$ to maximize the use of the available range. By enabling *auto_scale*, we can fully utilize the range of biases and coupling strengths that the system offers. These biases and coupling strengths are fixed and depend on the specific QPU being used [15].

Other important adjustable parameters are the *annealing_schedule* and the *annealing_time*. These two parameters enable us to control the annealing process and duration. Our focus is on the *annealing_schedule* as it determines the *annealing_time*, which is the total duration of the annealing process [15]. The D-Wave® machine *Advantage_system 4.1* has a standard annealing schedule that can be modified by specifying certain points the process must pass through, with a maximum of 10 points. These points are defined by specifying a pair $[t, s]$, where t is the time in microseconds and s represents the intensity of the magnetic field ranging from $s = 0$ (magnetic fields turned off) to $s = 1$ (maximum value of the magnetic field). If no specific annealing schedule points are set $[[0.0, 0.0], [20.0, 1.0]]$, the D-Wave® machine follows the default evolution of $A(s)$ and $B(s)$ shown in Figure 3.2 a). From this plot, we note that for $s \to 0$ $B(s) \to 0$ and for $s \to 1$ $A(s) \to 0$, as we want.

3.3 The embedding problem

In this Section, we focus on how many physical qubits are used for our calculations. We know that for various reasons, including noise, distant qubits coupling, and the architecture of the QPU, the quantum annealer uses more than one physical qubit to build a single logical qubit.

D-Wave® QPU is built with a particular topology (*Pegasus*) on which the CVP Hamiltonian has to be mapped. This process is called minor-embedding, D-Wave® provides automatic minor-embedding tools, and the solver handles all interactions with the QPU [16]. This process maps a logical qubit of the problem submitted into multiple physical qubits to allow coupling between qubits located in different cells in the QPU.

Using a built-in function in *Ocean*, it is possible to determine how many physical qubits are used to encode a specific problem. We consider n, the lattice rank, to be the number of our logical qubits, as we construct our Ising Hamiltonian using n binary variables. In Figure 3.2 b), we plot the number of physical qubits against the number of logical qubits.

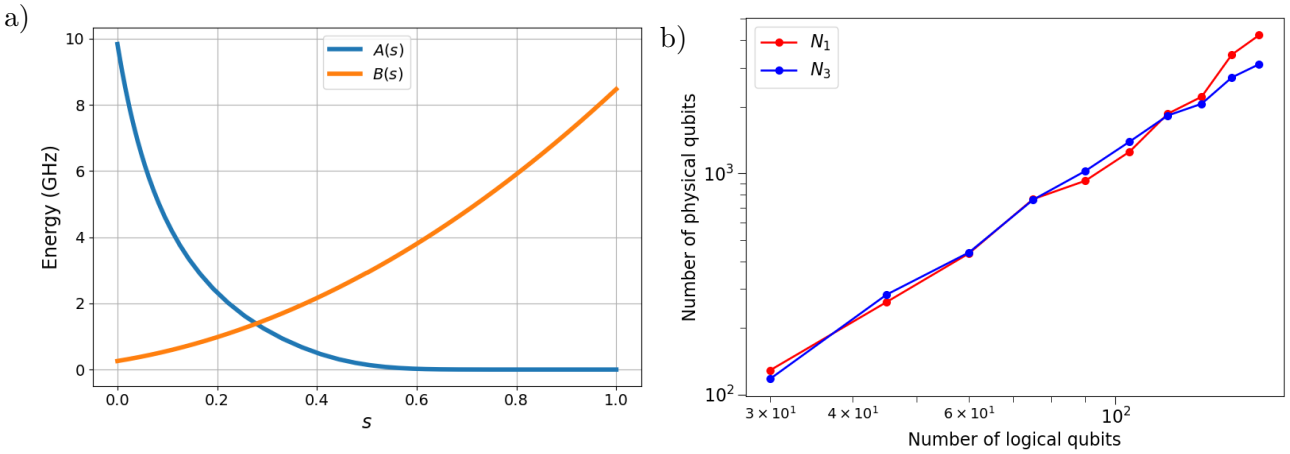


Figure 3.2: **a)** *Evolution of $A(s)$ and $B(s)$.* The time evolution of the Ising Hamiltonian parameters $A(s)$ and $B(s)$ as a function of s, the magnetic field intensity parameter. This particular *annealing_schedule* is the default annealing setup of *Advantage_system 4.1*. Data for the plot are available in D-Wave® web site [17].
b) *Physical qubits versus logical qubits.* The number of physical qubits used by the QPU to embed the problem as a function of the number of logical qubits, i.e., the lattice rank n. Both axes are plotted on a logarithmic scale. This plot is generated while solving the CVP constructed from two different semiprimes $N_1 = 27537500323$ and $N_3 = 1020970387714133542345824517187$. As expected, we observe that the number of physical qubits is independent from the semiprime N.

This result shows that resource usage is sub-polynomial not only for logical qubits but also for physical qubits. This class encompasses computational problems that a quantum computer can solve in polynomial time with a bounded probability of error. We can also observe that using this particular D-Wave® quantum machine, we can perform annealing for CVPs with at most 165 qubits.

Chapter 4

Results

In this Chapter, we present the results obtained from D-Wave® quantum simulations. First, we explain how we set up the simulations and which parameters of the CVP we adjusted to assess whether, and to what extent, the annealer can improve Babai's approximation and obtain more accurate solutions to the CVP. We then present histograms illustrating the sampling process from the state prepared at the end of the annealing procedure. We highlight the importance of robust statistics for evaluating D-Wave®'s performance in solving Schnorr's CVP instances. Finally, we discuss some preliminary insights on incorporating D-Wave® into the RSA number factorization pipeline within Schnorr's sieving [6], which relies on approximate solutions to a set of CVPs.

4.1 Simulation setup

As described in Chapter 1, this work focuses on solutions for specific instances of the CVP introduced by Schnorr in the context of factoring RSA security keys [2]. As detailed in Eqs. (1.2) and (1.3), a single instance of Schnorr's CVP is defined by selecting the RSA number to be factorized, the semiprime N, and specifying the size n of the prime number basis $P_1 = \{p_1, \ldots, p_n\}$, which determines the lattice rank. It is important to note that the rank corresponds to the number of logical qubits needed on the D-Wave® 's QPU to encode CVP solutions in the spin glass mapping, see Chapter 2. Table 4.1 shows the choices of the semiprime N and the lattice rank n we made to evaluate the performance of the D-Wave® machine in this Section.

Specifically, we selected three cryptographic keys of varying bit-lengths ℓ, measured by the number of bits required to represent the semiprime N. We recall that factoring such numbers becomes increasingly computationally difficult as this length grows. We considered two different strategies for the number of qubits n used to encode CVP solutions centered on Babai's approximation. In the first, we used Schnorr's proposal of sublinear scaling based on number theory and heuristic arguments [2]

$$n_1 = \left\lfloor \frac{\ell}{\log_2(\ell)} \right\rceil \tag{4.1}$$

where $\lfloor \cdot \rceil$ represents the nearest integer rounding operation and ℓ is the bit-length of the semiprime we aim to factorize. In the second case, we chose three increasing powers of two $n_2 = 16, 32$, and 64, one for each N. This choice is motivated simply by the convenience of working with powers of two within the tensor-network framework introduced in [6].

While solving the CVP independently from the factorization of N, our goal is to analyze D-Wave® 's solution as the lattice rank (the number of qubits) increases, where we expect Babai's approximation to become less accurate [1]. This is why we use two different values of the lattice rank $n_2 > n_1$, for a fixed N. For each combination of N and n, for a total of 6 different factorization problems (reported in Table 4.1), we run the quantum annealing procedure described in Chapter 3 on the D-Wave® machine to prepare a quantum state as close as possible to the ground-state of the CVP spin glass Hamiltonian in Eq. (2.5). We extract 40 random permutation of Schnorr's lattice diagonal in Eq. (1.2) for each problem and build different lattices using them. We want to study CVP solutions with different diagonal permutations looking at the factorization process proposed by Schnorr [2].

	N	bit-length	$n_1 = \left\lfloor \frac{l}{\log_2(l)} \right\rceil$	n_2
N_1	27537500323	35	6	16
N_2	784656869198858791459	70	11	32
N_3	1020970387714133542345824517187	100	15	64

Table 4.1: *Schnorr's lattices parameters.* RSA keys N used in this Section simulation and the different values of the lattice rank n_1 and n_2 used to encode the corresponding Schnorr's CVP instance as a spin glass Hamiltonian.

Next, we treat each single lattice-target pair as an individual CVP. For each CVP, we repeat the annealing evolution a certain number of times $N_{samples}$ to obtain the statistics of the measured qubit state. In Figure 4.1, we present results obtained by sampling $N_{samples} = 2000$ final states of a chosen diagonal, while in Figure 4.2 we show the case of $N_{samples} = 10000$ for the same CVP instances. In both Figures, we present an example instance for every different problem. Overall, we get $10000+2000$ samples for each diagonal and for each problem, for a total of $12,000 \times 40 \times 6 = 2,880,000$ annealing processes. Looking at the plot in Figures 4.1 and 4.2 we can observe that the number of samples $N_{samples}$ does not appear to affect the probability distribution of CVP solutions.

For completeness, we perform the annealing process with varying *annealing_time* settings of $50\mu s$ and to $100\mu s$, given the strong dependence of the annealing procedure on this parameter. We aim to investigate whether a longer annealing time increases the probability of reaching the ground state or reveals new lower-energy states. Figure 4.3 presents the results obtained with an annealing time of $50\mu s$, while Figure 4.4 shows the results for $100\mu s$. In both cases, the calculations use the same diagonal elements employed in Figures 4.1 and 4.1. Analyzing the plots, we observe no significant improvement in the probability of finding lower energy solutions. Additionally, there is no evidence suggesting that the sampled lower energy states differ from those obtained with an annealing time of $20\mu s$.

As our objective is to enhance Babai's solution to Schnorr's CVP, we compute the probability of finding lattice points that are closer to the target compared to Babai's vector. In Figure 4.5, we present these probabilities obtained by evaluating 40 different permutations of the diagonal for each CVP and calculating the average frequency at which the sampled vectors outperform Babai's solution.

We can observe that the improvement over Babai's solution with different probability for each different CVP varies across different CVP instances. Additionally, we find that the outcomes are relatively consistent, regardless of whether we perform 2000 or 10,000 annealing runs, indicating that increasing the number of samples does not significantly change the results in most cases.

We observe that the degree of improvement over Babai's solution varies across different CVP instances. Note that applying Schnorr's sublinear scaling theory yields a higher probability of improvement compared to using a larger lattice rank. This is expected, as solving the CVP for higher lattice ranks becomes more challenging. Consequently, sublinear lattices are easier to solve compared to those characterized by $n_2 > n_1$.

4.2 Assessing the role of the D-Wave®'s CVP solver in RSA factorization

In his paper [2], Schnorr proposes using approximate solutions derived from Babai's algorithm for an N-related set of CVPs constructed as detailed in Chapter 1, to find the two prime factors p and q decomposing N, thus breaking the widely used RSA encryption protocol in polynomial time. Schnorr's original idea was to combine multiple CVP solutions that satisfy a specific algebraic condition known as *smoothness* to derive p and q.

To define a smooth number, we introduce a second basis of prime numbers $P_2 = \{p_1, \ldots, p_r\}$ of dimension $r \geq n$, where n is the dimension of the factoring basis P_1. We use the basis P_2 as a smoothness basis and we call the last element of this basis p_r the smoothness bound. An integer is

said to be p_r-smooth on the selected bound if all its prime factors are less than or equal to the p_r.

In Schnorr's factoring algorithm, the smoothness of a lattice point is determined in the following way [2, 6]: first, we compute the components on the lattice basis of a given lattice point $\mathbf{v}$

$$\mathbf{z} = \lfloor B^T \cdot \mathbf{v} \rceil \tag{4.2}$$

where $\mathbf{B}$ is the lattice basis Eq. (1.2). Then, we define the triplet $(u, v, u - vN)$

$$u = \prod_{\substack{z_j \geq 0 \\ j \in [1,n]}} p_j^{z_j} \tag{4.3}$$

$$v = \prod_{\substack{z_j < 0 \\ j \in [1,n]}} p_j^{-z_j} \tag{4.4}$$

where N is the semiprime that we want to factorize. To compute u and v we use the basis P_1 used to build Schnorr's lattice, and the largest basis P_2 for the smoothness check. We classify a lattice point as a smooth point if $u - vN$ is p_r-smooth on the chosen bound p_r. In Schnorr's article [2], he proposes to set the dimension of the smoothness basis to n, thus achieving the factorization of N efficiently.

However, Schnorr's theoretical approach has faced criticism, as reported in [3, 18, 19, 20], and practical implementations of the algorithm have revealed that the sublinear smoothness bound he proposed does not yield enough smooth lattice points for successful factorization of the RSA key N. As a result, the algorithm has been revisited, and quantum variants have been proposed to speed-up the collection of smooth CVP solutions in [3, 18]. These quantum versions search for smooth pairs by encoding each CVP into a spin glass Hamiltonian, as we outlined in Chapter 2, aiming to improve the overall factorization pipeline. Babai's solutions, which provide at most one smooth point per lattice, are insufficient for succesful factorization, as demonstrated in [6], due to the absence of p_r-smooth lattice points on a sublinear smoothness basis.

The advantage of the quantum approach lies in the fact that, through quantum mapping, more than just Babai's solution can be obtained. Each eigenstate of the Hamiltonian represents a potential smooth lattice point. Theoretical and heuristic insights suggest that lower-energy eigenstates are more likely to correspond to useful lattice points for factorization [2]. However, a recent approach using tensor networks [6] has shown that, for effective factorization, the smoothness basis P_2 must be significantly larger than the one proposed by Schnorr to generate enough smooth pairs, and also the number of qubits n needed to constuct and map the CVPs must be much larger than its sublinear theory. It also revealed that not only the ground state and the lowest excited states can be smooth, but even higher-energy states may offer viable solutions for factoring N.

In this Section, we explore whether using D-Wave® and the approach discussed so far to search for smooth lattice points across different CVPs can provide a quantum advantage over the classical quantum-inspired tensor network methods proposed in [6]. To assess this, we analyze the average smoothness properties of the eigenstates sampled by the D-Wave® quantum annealer. Our rationale is that the imperfect nature of adiabatic evolution featured by the real QPU could allow us to sample a substantial number of eigenstates. Unlike in previous Sections, where strict adiabaticity was crucial for solving the CVP, successful factorization doesn't necessarily require reaching the true ground state. This opens up the possibility of leveraging the broader spectrum of states produced by the quantum annealer to collect smooth CVP solutions.

We execute the smoothness check for lattice points obtained from sampling of CVPs generated by diagonal permutation showed in Section 4.1 with $N_{samples} = 10000$. To check Schnorr results we first perform the smoothness check with a sublinear smoothness basis P_1 containing n prime numbers, where n correspond to the lattice rank. We report the results obtained in Figures 4.6. We observe no smooth lattice point for this basis, as predicted in [3].

Next, we perform the smoothness check for the same results with a different basis P_2 with a dimension dependent from the lattice rank n and the bit length of the semiprime ℓ we aim to factorize. Specifically, we use as P_2 dimension $r = 2n\ell$. We report the results obtained in Figure 4.7.

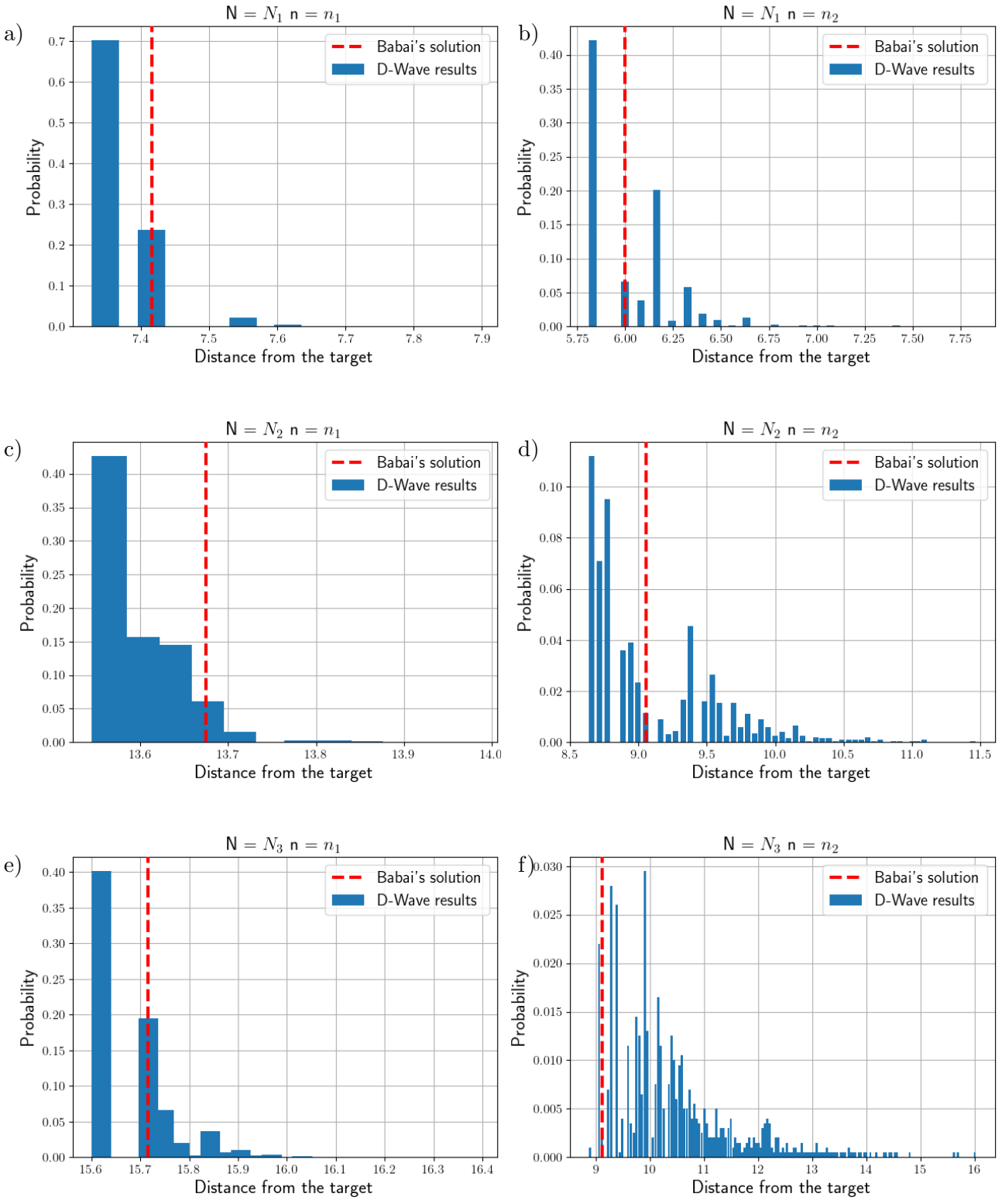


Figure 4.1: *Sampled D-Wave® spectrum of CVP solutions with* $N_{samples} = 2000$. Each plot represents the measured eigenstates (lattice points) after the annealing process for each of the factorization problems determined by the pairs (N, n) for a specific randomly-permuted lattice diagonal $\{f(1), \ldots, f(n)\}$. The x-axis shows the distance $\|\boldsymbol{b} - \boldsymbol{t}\|$ of the sampled lattice points from the target vector. The y-axis indicates the probability of obtaining the corresponding eigenstate of the spin glass Hamiltonian in Eq. (2.5) using the D-Wave® sampler. The red dashed line marks the distance of Babai's solution from the target.

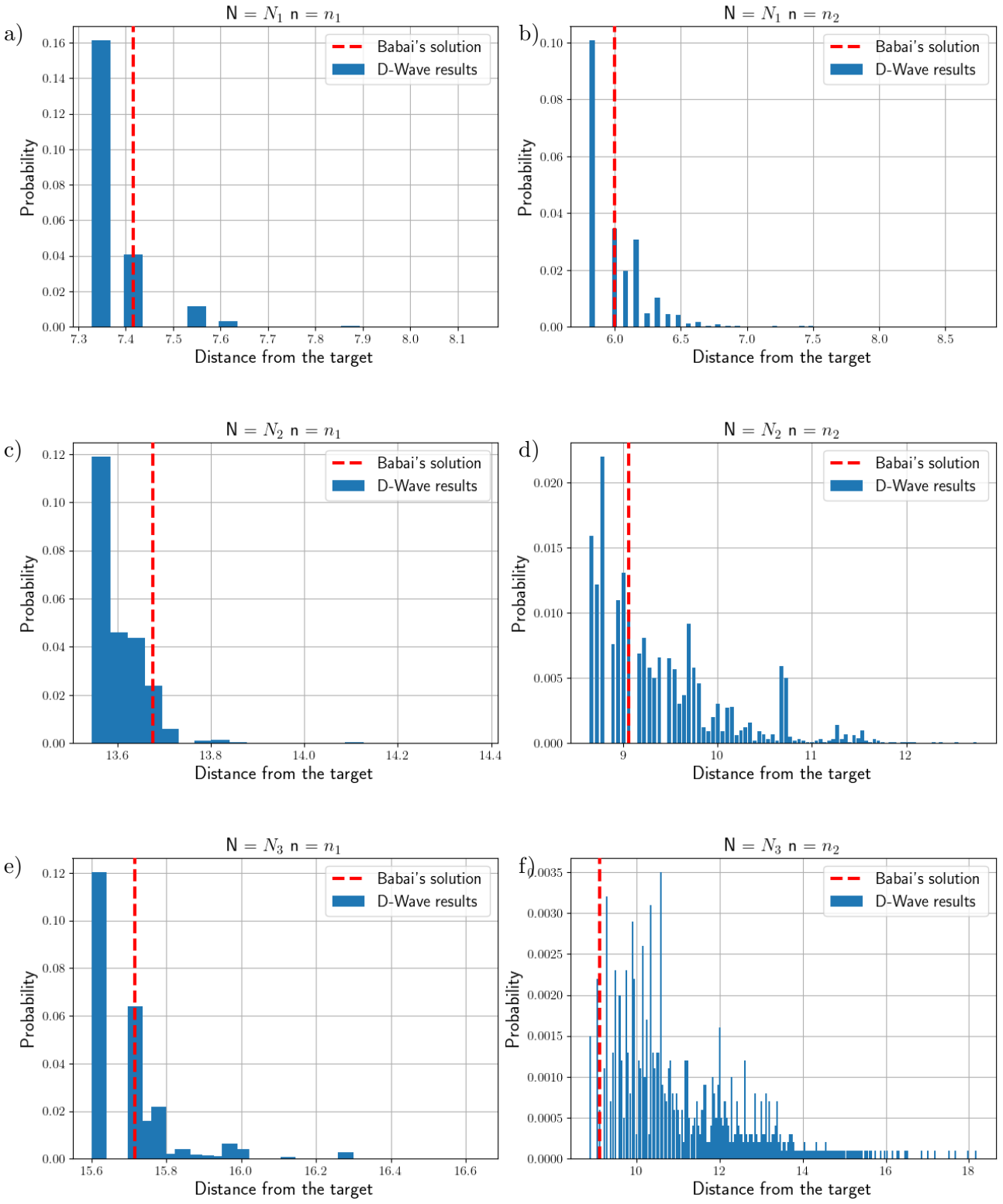


Figure 4.2: *Sampled D-Wave® spectrum of CVP solutions with* $N_{samples} = 10000$. Each plot represents the measured eigenstates (lattice points) after the annealing process for each of the factorization problems determined by the pairs (N, n) for the same lattice diagonal $\{f(1), \ldots, f(n)\}$ used in Figure 4.1 to check if the number of samples modifies the distribution. The x-axis shows the distance $\|\boldsymbol{b} - \boldsymbol{t}\|$ of the sampled lattice points from the target vector. The y-axis indicates the probability of obtaining the corresponding eigenstate of the spin glass Hamiltonian in Eq. (2.5) using the D-Wave® sampler. The red dashed line marks the distance of Babai's solution from the target.

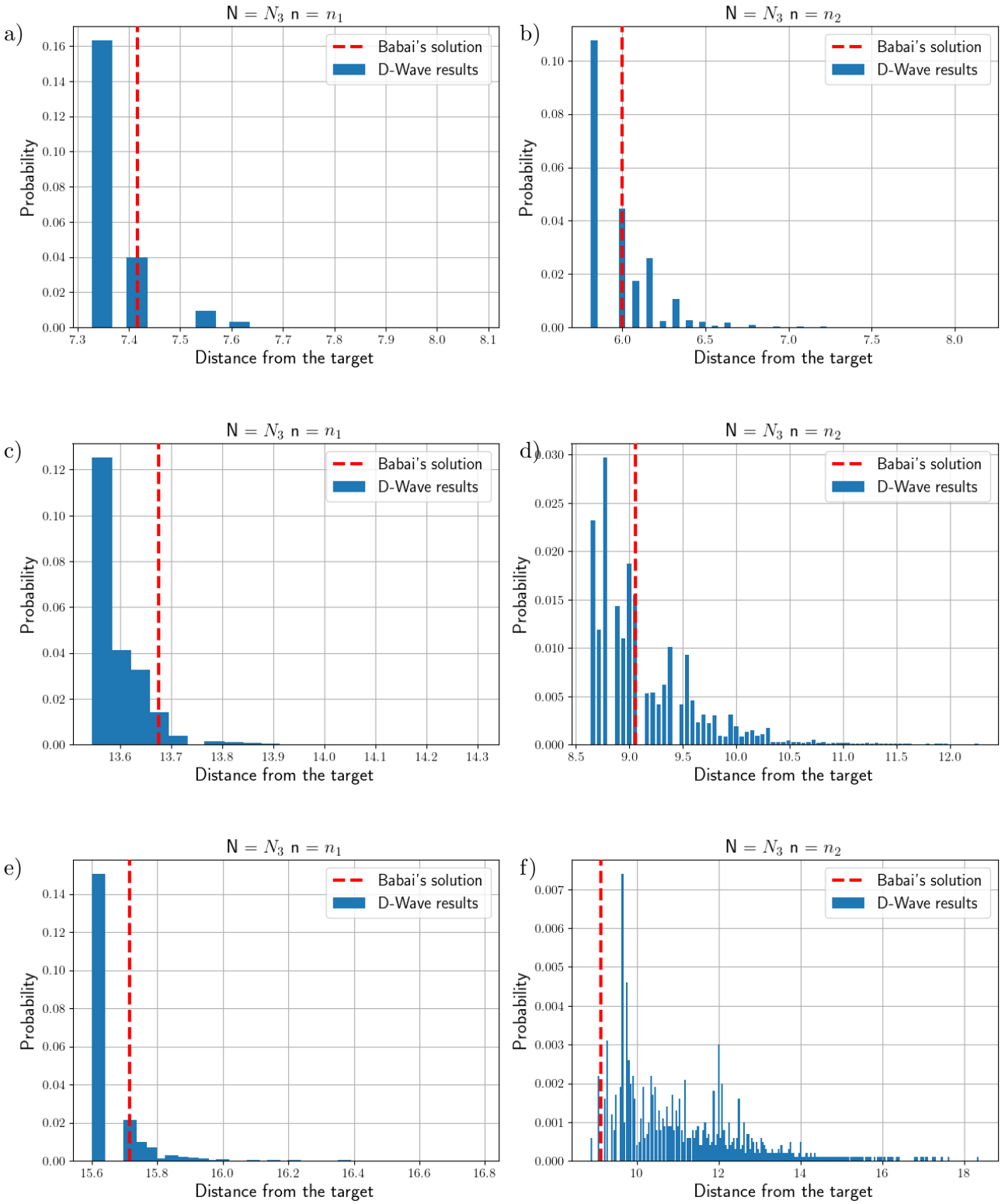


Figure 4.3: *Sampled D-Wave® spectrum of CVP solutions with* $N_{samples} = 10000$ *and* annealing_time=50μs. Each plot represents the measured eigenstates (lattice points) after the annealing process for each of the factorization problems determined by the pairs (N, n) for the same lattice used in Figure 4.2. The x-axis shows the distance $\|\boldsymbol{b} - \boldsymbol{t}\|$ of the sampled lattice points from the target vector. The y-axis indicates the probability of obtaining the corresponding eigenstate of the spin glass Hamiltonian in Eq. (2.5) using the D-Wave® sampler. The red dashed line marks the distance of Babai's solution from the target.

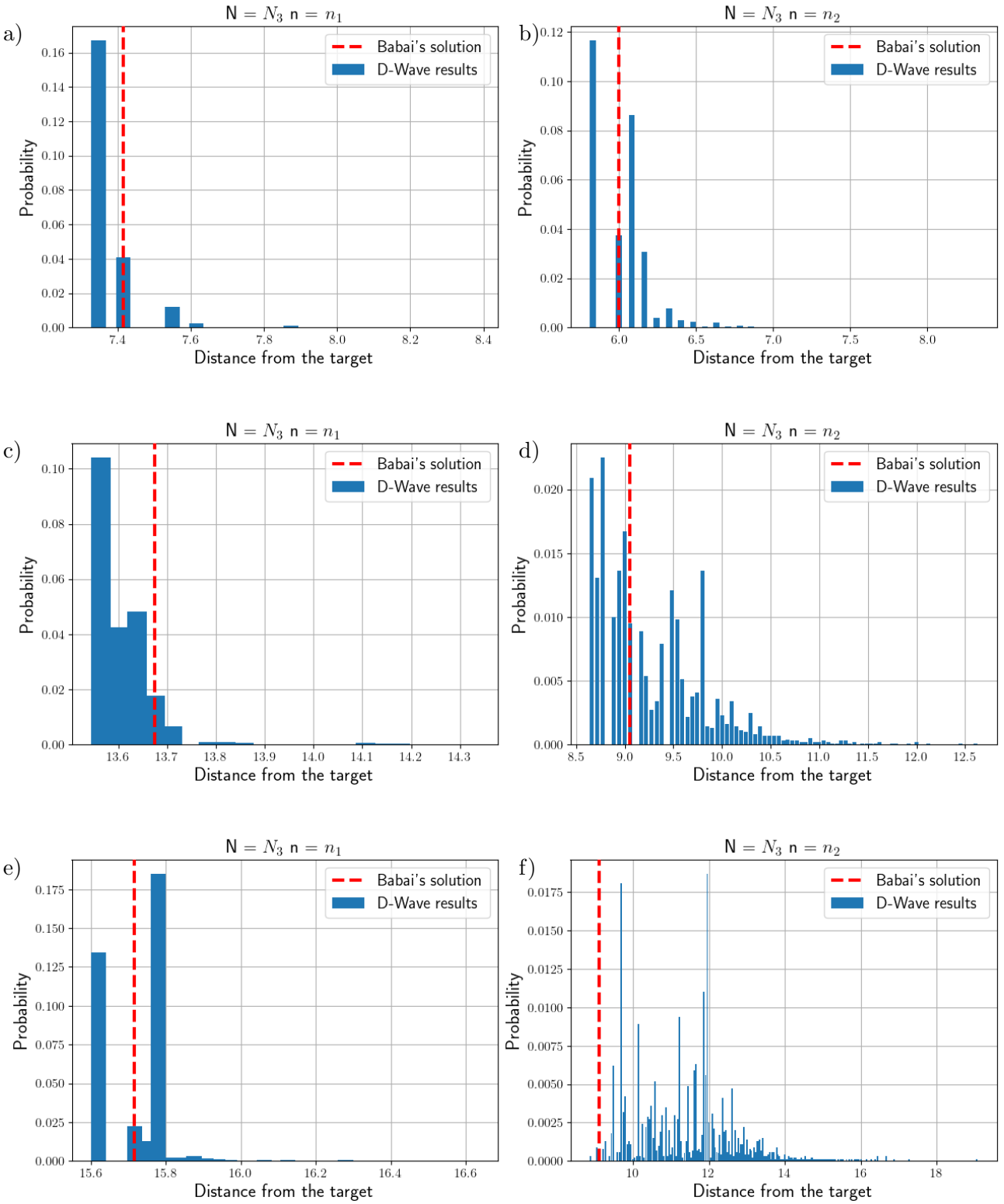


Figure 4.4: *Sampled D-Wave® spectrum of CVP solutions with* $N_{samples} = 10000$ *and* annealing_time=100μs. Each plot represents the measured eigenstates (lattice points) after the annealing process for each of the factorization problems determined by the pairs (N, n) for the same lattice used in Figure 4.2. The x-axis shows the distance $\|\boldsymbol{b} - \boldsymbol{t}\|$ of the sampled lattice points from the target vector. The y-axis indicates the probability of obtaining the corresponding eigenstate of the spin glass Hamiltonian in Eq. (2.5) using the D-Wave® sampler. The red dashed line marks the distance of Babai's solution from the target.

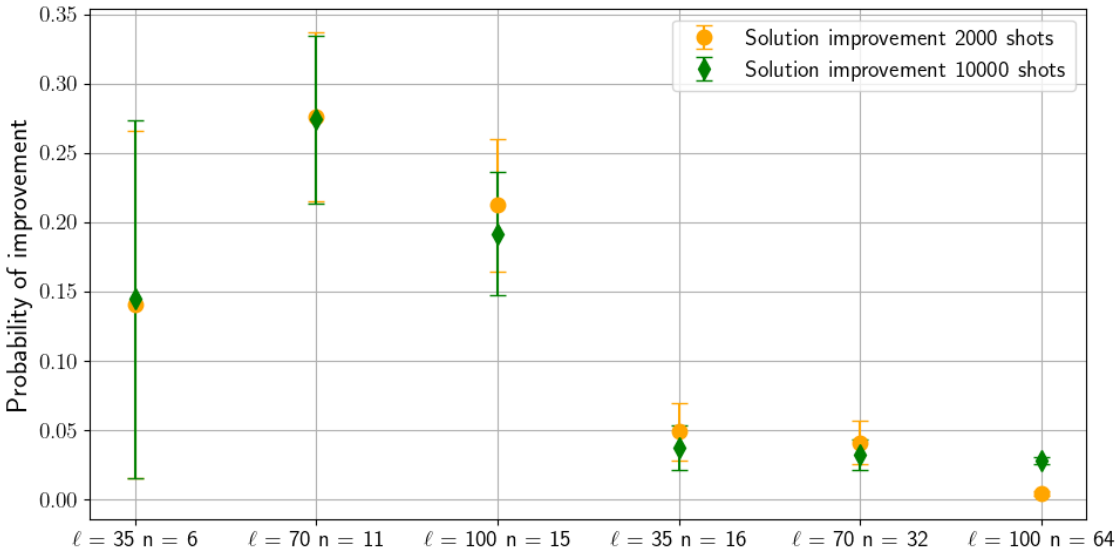
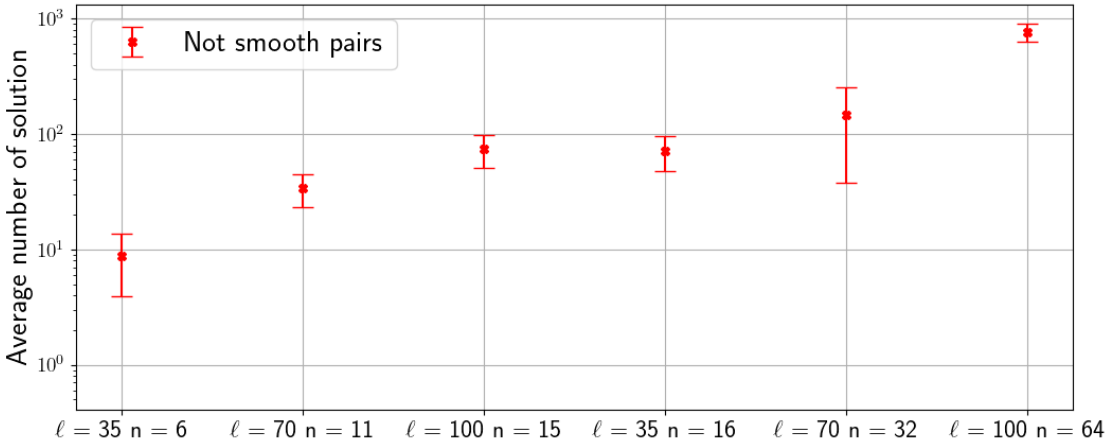


Figure 4.5: *Probability of improvement of Babai's solution.* The probability of finding a vector closer to the target than Babai's solution from the D-Wave® quantum annealing, averaged on 40 different permutations of the diagonal determining Schnorr's CVP for each factoring problem (N, n) considered.

a)



b)

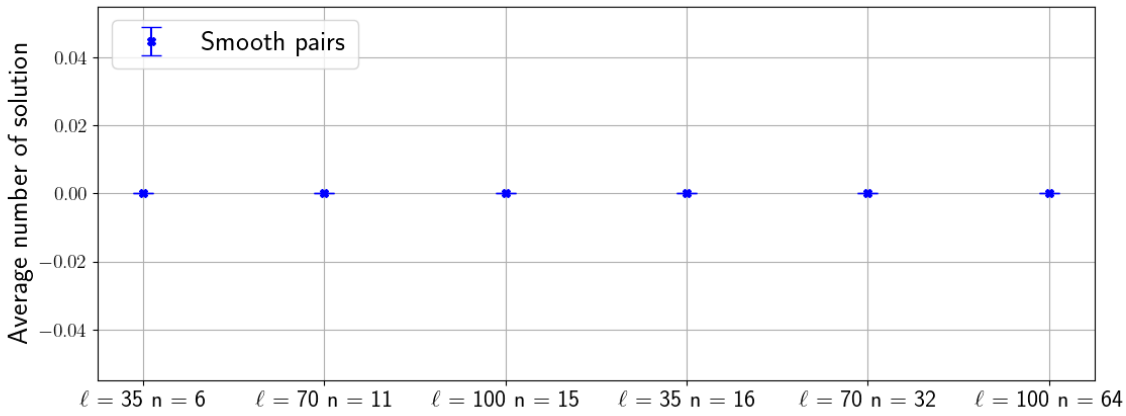


Figure 4.6: *Number of smooth and not smooth lattice points.* The number of smooth **b)** and not smooth **a)** lattice point for 40 permutation of the diagonal respect to a basis of n prime numbers. For a n-smooth basis of prime number we observe no smooth lattice points as predicted in [3]. We use a logarithmic scale on y-axis in **a)** but not in **b)** to show correctly points on 0.

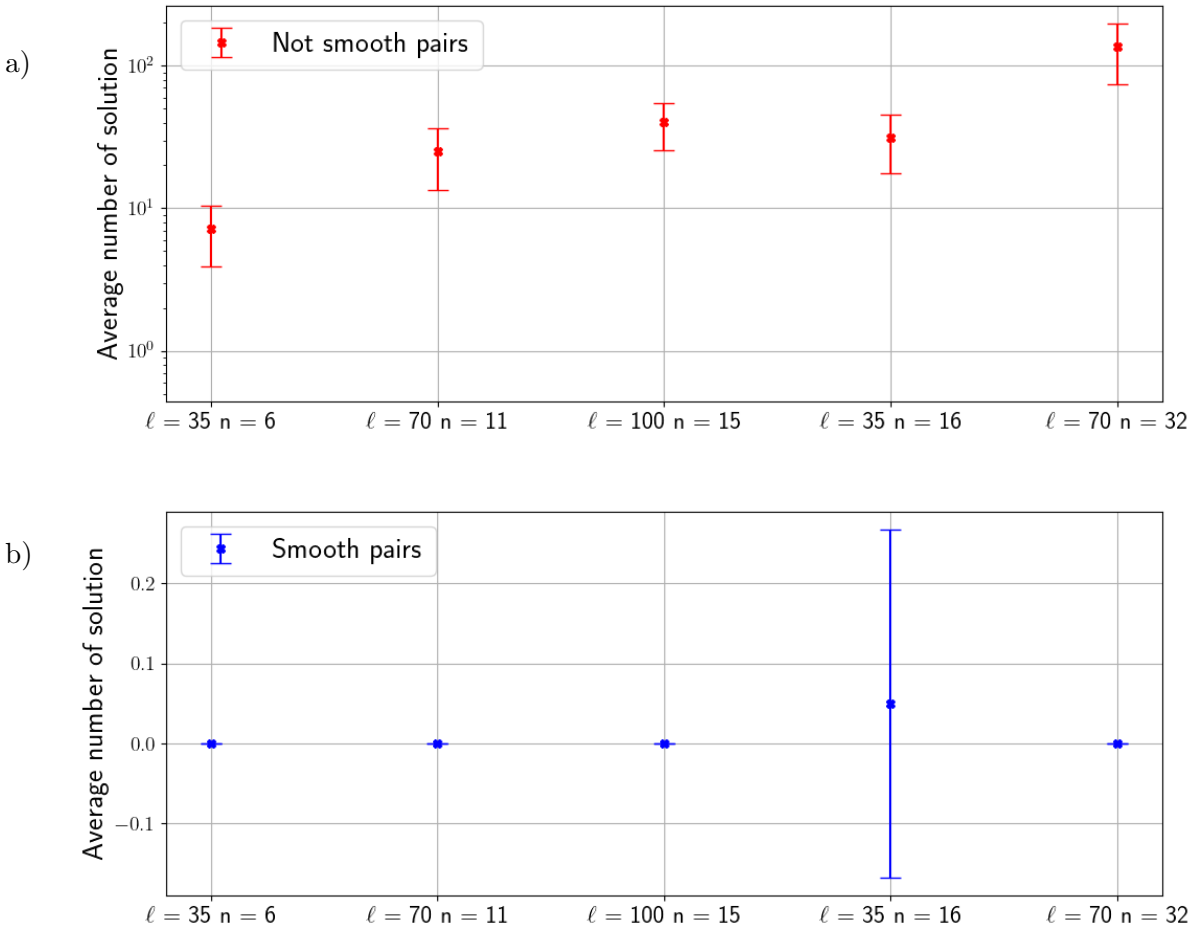


Figure 4.7: *Number of smooth and not smooth lattice points for basis P_2.* The number of smooth **b)** and not smooth **a)** lattice point for 40 permutation of the diagonal respect to a basis of r prime numbers. For a r-smooth basis of prime number we observe smooth lattice points only for the case N_1 n_2. We use a logarithmic scale on y-axis in **a)** but not in **b)** to show correctly points on 0.

Conclusions and outlooks

This Thesis addressed the approximate solution of the Closest Vector Problem (CVP) using a quantum annealer. To achieve this objective, the lattice problem was reformulated into a mathematical problem compatible with the D-Wave® quantum annealer. Specifically, the problem was mapped onto a spin glass Hamiltonian, where the low-energy states encode potential improvements to state-of-the-art classical approximate solutions given by Babai's algorithm. Our study focused on Schnorr's lattices [2], which are at the core of Schnorr's method for factoring RSA numbers [3, 6, 19].

A considerable part of this work was dedicated to deriving the mapping of the CVP onto an Ising-like spin glass Hamiltonian. After obtaining the coupling terms of the Hamiltonian, quantum simulations were carried out using the D-Wave® quantum annealer to sample solutions for various CVP instances. The primary objective was to explore whether the low-energy spectrum of the spin-glass Hamiltonian, accessed via quantum annealing, could yield better solutions than Babai's classical approximation.

For each Schnorr's CVP problem considered, the quantum annealer consistently produced at least one vector closer to the target than Babai's solution, with sampling probabilities ranging between 2.6% and 27.8%. Additionally, we studied the smoothness property of the sampled lattice points to investigate if this method can be efficient for factoring semiprimes within Schnorr's sieving algorithm [3, 6]. However, the solutions obtained via quantum annealing were not effective for factorization.

Future research could explore several avenues for improvement. First, fine-tuning the parameters of the D-Wave quantum annealer is essential to enhance the efficiency of the quantum annealing process and to achieve final states closer to the true ground state of the Hamiltonian. Such advancements could enable the use of the quantum annealer in Schnorr's algorithm for factorization. Moreover, the findings of this Thesis may serve as a benchmark for Tensor-Network approaches, such as the one proposed in [6].

As discussed in Section 3.3, the scalability of physical qubits remains a critical challenge. To achieve RSA key factorization, quantum processing units (QPUs) capable of embedding more than 165 logical qubits will be necessary [6]. Finally, another promising development would involve refining Babai's solution through rounding corrections of $\pm M$, with $M \in \mathbb{N}$, rather than the ± 1 corrections applied in this work, thus proposing a novel quantum mapping of CVPs. These strategies could then further enhance the accuracy of solutions derived through quantum annealing.

Bibliography

[1] S.D. Galbraith. *Mathematics of Public Key Cryptography*. Cambridge University Press, 2012, pp. 16–18. ISBN: 9781107013926. URL: https://books.google.it/books?id=owd76BElvosC.

[2] Claus Peter Schnorr. *Fast Factoring Integers by SVP Algorithms, corrected*. Cryptology ePrint Archive, Paper 2021/933. 2021. URL: https://eprint.iacr.org/2021/933.

[3] Bao Yan et al. *Factoring integers with sublinear resources on a superconducting quantum processor*. 2022. arXiv: 2212.12372 [quant-ph]. URL: https://arxiv.org/abs/2212.12372.

[4] Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information*. eng. 10th anniversary ed. Cambridge: Cambridge University Press, 2010. ISBN: 0-511-99400-1.

[5] C.H. Papadimitriou. *Computational Complexity*. Theoretical computer science. Addison-Wesley, 1994. ISBN: 9780201530827. URL: https://books.google.es/books?id=JogZAQAAIAAJ.

[6] Marco Tesoro et al. *Quantum inspired factorization up to 100-bit RSA number in polynomial time*. 2024. arXiv: 2410.16355 [cs.CR]. URL: https://arxiv.org/abs/2410.16355.

[7] D-Wave. *Solving Problems with Quantum Samplers — D-Wave System Documentation*. Accessed: 2024-11-04. 2020. URL: https://docs.dwavesys.com/docs/latest/c_gs_3.html#ising-model.

[8] Andrew Lucas. "Ising formulations of many NP problems". In: *Frontiers in Physics* 2 (2014). ISSN: 2296-424X. DOI: 10.3389/fphy.2014.00005. URL: http://dx.doi.org/10.3389/fphy.2014.00005.

[9] Philipp Hauke et al. "Perspectives of quantum annealing: methods and implementations". In: *Reports on Progress in Physics* 83.5 (May 2020), p. 054401. ISSN: 1361-6633. DOI: 10.1088/1361-6633/ab85b8. URL: http://dx.doi.org/10.1088/1361-6633/ab85b8.

[10] D-Wave. *Get Started with D-Wave Solvers - What is Quantum Annealing?* Accessed: 2024-11-19. 2022. URL: https://docs.dwavesys.com/docs/latest/c_gs_2.html.

[11] Finley Alexander Quinton et al. *Quantum annealing versus classical solvers: Applications, challenges and limitations for optimisation problems*. 2024. arXiv: 2409.05542 [quant-ph]. URL: https://arxiv.org/abs/2409.05542.

[12] D-Wave. *Welcome to D-Wave - D-Wave's Quantum Computer Systems*. Accessed: 2024-11-19. 2022. URL: https://docs.dwavesys.com/docs/latest/c_gs_1.html.

[13] D-Wave. *Getting Started with D-Wave Solvers - D-Wave QPU Architecture: Topologies*. Accessed: 2024-11-19. 2022. URL: https://docs.dwavesys.com/docs/latest/c_gs_4.html#pegasusqubits.

[14] D-Wave Systems Inc. *D-Wave Ocean SDK*. Version 8.0.1. 2024. URL: https://docs.ocean.dwavesys.com/.

[15] D-Wave. *Solver Parameters — D-Wave System Documentation*. Accessed: 2024-11-04. 2022. URL: https://docs.dwavesys.com/docs/latest/c_solver_parameters.html.

[16] D-Wave. *Getting Started with D-Wave Solvers - Constraints Example: Minor-Embedding*. Accessed: 2024-11-19. 2022. URL: https://docs.dwavesys.com/docs/latest/c_gs_7.html.

[17] D-Wave. *QPU Solvers Datasheet - Annealing Implementation and Controls*. Accessed: 2024-11-19. 2022. URL: https://docs.dwavesys.com/docs/latest/c_qpu_annealing.html#qpu-annealprotocol-standard.

[18] Tanuj Khattar and Noureldin Yosri. *A comment on "Factoring integers with sublinear resources on a superconducting quantum processor"*. 2023. arXiv: 2307.09651 [quant-ph]. URL: https://arxiv.org/abs/2307.09651.

[19] Willie Aboumrad, Dominic Widdows, and Ananth Kaushik. *Quantum and Classical Combinatorial Optimizations Applied to Lattice-Based Factorization*. 2023. arXiv: 2308.07804 [quant-ph]. URL: https://arxiv.org/abs/2308.07804.

[20] Sergey V. Grebnev et al. "Pitfalls of the Sublinear QAOA-Based Factorization Algorithm". In: *IEEE Access* 11 (2023), pp. 134760–134768. DOI: 10.1109/ACCESS.2023.3336989.