Department of Information Engineering and Computer Science

Master's Degree in
Computer Science, Computational Foundations

final dissertation

EQDR: an Elite Quantum Diffusion Recombination procedure

Adapting quantum search algorithms for fitness-based optimization problems

Supervisor
Davide Pastorello

Student
Matteo Beltrame

Co-supervisors
Enrico Blanzieri
Enrico Zardini

Academic year 2022/2023

Acknowledgements

Without the shadow of a doubt, the best acknowledgements are the minimal and sincere ones.
I want to express my appreciation toward my supervisor Davide Pastorello who supported me in developing the idea I had for the thesis. A special thanks goes to my co-supervisors Enrico Blanzieri and Enrico Zardini for having helped me with valuable advice.

As an off-site student, I did not have the chance to always be there for my close friends. I want to thank you all for making my university period lighter.

The acknowledgement of a thousand thesis would not be enough to express the gratitude I have toward my family, always ready to support and encourage me despite my behavior. I extend special thanks to my mother Susanna for all the advice and persistence in helping me.
Again, to my brother Fabio I dedicate this work. Long thanks for the umpteenth time would be redundant, I just want to remind you that I will never stop considering you as a role model. Thank you for all the times you have put me back on the right path.

Index

Table of Figures

Abstract

With the recent development of quantum computing technology, the interest in understanding the actual advantages of quantum computers over classical ones has been steadily rising as well.
As folklore computer science teaches, without having substantive information about the modeling problem, there is no single model that will always do better than any other model (NFL, No Free Lunch Theorem).
It is not yet clear which classes of problems can be solved more efficiently in a quantum environment with respect to a classical one. The conjecture is that, given the quantum principles upon which quantum computers are based on, such as interference and superposition among others, quantum computers should have an intrinsic advantage over classical computers. However, there are numerous caveats in building, maintaining, and programming a quantum computer, especially in these initial years in which the technology is not yet mature enough. Noise, decoherence, initialization overhead and non-determinism of results constitute only a subset of the numerous challenges that quantum computing poses. This entails that, in the current era (NISQ, Noisy Intermediate Scale Quantum computers), the development of algorithms that work in synergy with classical machines is a critical point.
In the field of optimization, quantum computing has been widely used in all its paradigms: from quantum annealing to circuit-based algorithms. In particular, many solutions based on the well-known Grover's search algorithm, which can be used for efficient searches in unstructured databases, have been proposed. The fundamental element of the algorithm is the quantum oracle, an operator that is able to efficiently recognize a solution. Methods that make use of Grover's algorithm tend to show a clear advantage in search-based problems. However, this is often due to the possibility to delegate to the oracle itself the hard part of solving the problem. The truth is that oracles are very complicated to design and to implement in practice: they are problem-specific and, in almost every application, they are implemented with variational circuits.
In many real-world scenarios, the optimum quantum oracle is either too difficult to implement or so complicated that cannot be executed on a NISQ device with the required precision.
In these situations, it is possible to define a partial oracle that is able to identify a superset of elements among which only few of them are actual solutions. This characteristic can be seen as the oracle being able to recognize only certain features of the solutions. Executing the vanilla Grover Algorithm with this partial oracle basically means running a random sampling algorithm on the subspace of the solutions that are recognized by the oracle.
By combining Grover's algorithm with the theory of Evolutionary Computation (EC), it is possible to apply quantum search algorithms to fitness-based optimization problems even in situations in which only a partial oracle can be exploited. In other words, the goal is to ease the burden placed on quantum oracles and share it with well-established evolutionary computation techniques in order to efficiently drive the repeated applications of Grover's algorithm. The proposed algorithm, called EQDR (Elite Quantum Diffusion Recombination), gathers inter-generational knowledge regarding the solutions and the search space with the various measurements and exploits this knowledge to apply specific adaptive quantum operators. These operators, applied to the superposition of states returned by Grover's algorithm, can be seen as the quantum analogous of the classical recombination operators, namely, crossover and mutation. A key consideration is the fact that these quantum operators are applied to a quantum superposition, hence enabling recombination and mutation without the loss of information caused by the collapse of the wave function due to measurements.
As shown by the experimental results, the proposed solution can effectively enhance Grover's algorithm in particular situations and, more importantly, extend it for different applications in the field of fitness-based optimization. In the current era, the overhead of quantum initialization and quantum simulation is extremely high, for this reason, it does not make much sense to compare quantum and classical algorithm the means of the real running time. It is way more interesting to compare quantum and classical algorithms by means of subroutine calls. In the framework of evolutionary computation, this means analyzing the number of fitness calls required to find the optimum. A promising result of the proposed procedure is the fact that the number of times the fitness function is called is independent of the dimension of the population. In classical evolutionary computation, it is required to call the fitness function a number of times that is at least as large as the population size, in order to assign to each solution its fitness. The synergy of Grover's algorithm and the genetic

knowledge of selected generational elite individuals gathered during runs enables the EQDR algorithm to execute the fitness function exactly once in each generation. This result suggests that in the next years, with the further development of quantum hardware and the resolution of the challenges of quantum computing, the considerable reduction in the number of fitness calls could enable quantum optimization algorithms to find a solution faster with respect to classical ones, bringing unprecedented advantages to the quantum optimization procedures. However, the EQDR algorithm presents an advantage mainly in specific classes of problems. In case the oracle is very close to the optimum one, meaning it can recognize almost every wanted solution, the EQDR algorithm loses its advantages. In this scenario, the vanilla Grover's algorithm, extended to multiple solutions, randomly samples in a subspace that includes almost all the solutions, enabling to find one of them in an average number of iterations that is comparable if not smaller than the one required by the EQDR algorithm.

The classical methodologies used to exploit the gathered genetic knowledge play a key role in the EQDR algorithm and the best option depends on the problem at hand, specifically on the properties of the fitness function used. This entails that the algorithm must be adapted to the optimization problem in order to obtain acceptable results. Fortunately, in these situations, it is possible to exploit the well-established theory of evolutionary computation since the subroutine that exploits the genetic knowledge to generate the parameters for the variational quantum operators can be seen as a classical black box. This aspect is therefore a strong point of the EQDR algorithm with respect to purely quantum search optimization algorithms. The ability to adapt itself to various situations and problems thanks to the heuristic methodologies of evolutionary computation, allows to not offload all the responsibility to an oracle which is often extremely difficult to implement.

1. Quantum computing and quantum search

Quantum computing is the discipline that studies quantum computers, computational devices that exploit quantum phenomena such as interference, superposition and entanglement, among others, in order to carry out a computational job. To deeply understand the concept and the results of the proposed work, it is important to refresh the basic notions of quantum computing and its fundamental concepts.
In this chapter, an introduction to quantum computing is given. The first section deals with quantum bits and basic quantum computing notions, such as multiple qubits systems, quantum gates and quantum circuits, fundamental building blocks of any quantum algorithm.
The second section introduces the most famous quantum algorithms, in order to explain how quantum features can be exploited, with a particular emphasis on the problem of quantum search, Grover's algorithm, its application and its drawbacks.

1.1 Introduction to quantum computing

Before introducing quantum gates and quantum algorithm an important step is to deeply understand the quantum information units, that is, the quantum analogous of the classical bit. Bits are the fundamental concept of classical computation. Analogously, quantum computation is built on top of a similar concept, i.e., the quantum bit, or qubit.

1.1.1 Qubits

The qubit is a two-state (two-level) quantum-mechanical system; in other words, it is a vector in a two-dimensional complex vector space. The same way as classical bits have two possible states, 0 or 1, qubits have the two analogous states, $|0\rangle, |1\rangle$, expressed using Dirac's notation. However, quantum mechanics allows qubits to be in any linear combination of the two eigenstates $|0\rangle$ and $|1\rangle$. Any linear combination of the two eigenstates is called superposition. An arbitrary state of a qubit can be written as:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1.1}$$

$$|\alpha|^2 + |\beta|^2 = 1$$

$$\alpha, \beta \in \mathbb{C}$$

The square module of the coefficients α and β determines the probability of obtaining respectively the state $|0\rangle$ or the state $|1\rangle$ upon measuring the qubit and, due to the law of total probabilities, the partial probabilities must sum to 1.
It is fundamental to keep in mind that, despite the probabilistic interpretation, the physical system described above does not represent the situation in which the state is either $|0\rangle$ or $|1\rangle$ with uncertainty; instead, the state is both $|0\rangle$ and $|1\rangle$ at the same time. Upon measurement, the wave function collapses and gives a classical result with a probability equal to the square module of the corresponding coefficient. This concept is fundamental in quantum computing and, as it will be shown in later sections, it is the main reason for the quantum speedup in algorithms due to a phenomenon called quantum parallelization.

The Bloch Sphere

In order to visualize single qubit operations, which will be presented in the following sections, it can be useful to think about qubits in terms of a geometrical representation. Given the qubit's state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and due to the normalization condition $|\alpha|^2 + |\beta|^2 = 1$, the state can be rewritten as:

$$|\psi\rangle = e^{i\gamma}\left(cos\frac{\theta}{2}|0\rangle + e^{i\phi}\, sin\frac{\theta}{2}|1\rangle\right)$$

The factor $e^{i\gamma}$ can be ignored as it represents a global phase that has no observable effects; therefore, the state can be rewritten as:

$$|\psi\rangle = cos\frac{\theta}{2}|0\rangle + e^{i\phi}\, sin\frac{\theta}{2}|1\rangle$$

The values of θ and ϕ identify a point on the three-dimensional unit sphere, called the Bloch Sphere, shown in Figure 1.

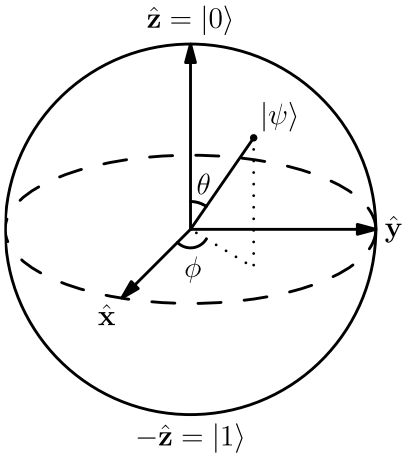

Figure 1: the Bloch Sphere, a geometrical representation of qubits [1]

The Bloch Sphere represents an effective tool to visualize the state of a qubit and to grasp the geometrical intuition of certain single qubit gates. Nonetheless, it must be kept in mind that the Bloch Sphere representation is limited. Quantum algorithms employ multiple qubits for their computation, and currently there is not a simple generalization of the Bloch Sphere to multiple qubits systems which can be used to effectively represent multi qubits systems [2].

Multiple qubits systems

In the same way classical computing exploits multiple bits to carry out computations, quantum algorithms exploit multiple qubits systems and the quantum features that derive from them. Let us consider a system composed of 2 bits, there are 4 possible states: $00, 01, 10, 11$. In the same way, the quantum system has the 4 computational basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$. A pair of qubits can therefore also exist in any superposition of the basis states. The most general state for a two-qubit system is:

$$|\psi\rangle = a_{00}|00\rangle + a_{01}|01\rangle + a_{10}|10\rangle + a_{11}|11\rangle$$

Similarly to the single qubit case, the measurement result $x = (00, 01, 10\ or\ 11)$ occurs with probability $|a_x|^2$. More generally, a system of n qubits can be described by the following Equation:

$$|\psi\rangle = \sum_{i=0}^{2^n-1} a_i|i\rangle \qquad 1.2$$

$$\sum_i |a_i|^2 = 1, \qquad a_i \in \mathbb{C}$$

where $|i\rangle$ is the state described by the binary string representation of i.
There are some states in multiple qubits systems that present some very peculiar characteristics. Let us consider the two-qubit system state:

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

The possible outcomes of a measurement process on the first qubit are $|0\rangle$ and $|1\rangle$, both with probability $\frac{1}{2}$.
If, upon measuring the first qubit, the outcome is $|0\rangle$, the post-measurement state will be $|\psi'\rangle = |00\rangle$. On the other hand, if the outcome is $|1\rangle$, the post-measurement state will be $|\psi'\rangle = |11\rangle$. This means that the measurement of the second qubit will always give the same result as the first qubit, meaning that measurements are correlated. This particular correlation is called quantum entanglement and it is a fundamental quantum computing feature, the main ingredient for quantum algorithms such as quantum teleportation and superdense coding.
An interesting question is: how much information can a qubit represent?
Naively, one could think that, since a qubit is a complex vector in a two-dimensional space, the information that can represent is infinite, since the number of possible combinations of the two coefficients α and β is infinite. However, this conclusion is misleading, due to the way qubits behave when observed. Whenever a qubit is measured, the output is either 1 or 0. Moreover, after the measurement, the qubit's state collapses to the outcome of the measurement: any successive measurement on the same qubit will yield the same identical result. The collapse of the wave function is a postulate of quantum mechanics, and, although widely accepted and proven, nobody knows why and exactly how this phenomenon happens.
During computation, until the qubit is measured, all its possible states are taken into consideration by quantum mechanics. This indeed suggests how nature keeps track of all possibilities during the quantum computation. Considering a system of n qubits, the number of amplitudes required to describe the system grows exponentially in the number of qubits. Recalling (1.2), the number of possible states, hence the number of coefficients (amplitudes) associated to the states, is 2^n. This is the main reason why simulating quantum computers with classical machines is hard. A classical computer must keep track of all the amplitudes of the simulated quantum system, having to deal with an exponential overhead with respect to a quantum computer, which is able to simply evolve the system according to fundamental quantum mechanical laws.

1.1.2 Quantum gates

The same way classical bits flow through an electrical circuit that applies transformations to them with logical gates, qubits flow through a quantum circuit which is composed of quantum gates.
Many quantum gates are generalizations of the classical gates, hence they maintain the same logical intuition. Quantum gates can be represented as matrices. In particular, single qubits gates are represented by 2x2 matrices. As an example, let us consider the NOT gate, which classically flips the state of the bit. It can be represented as the following matrix:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The matrix, acting on an arbitrary $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ state of the qubit, simply swaps the role of the two basis states, effectively swapping the two coefficients:

$$X|\psi\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix} \rightarrow |\psi'\rangle = \beta|0\rangle + \alpha|1\rangle$$

Another fundamental single qubit gate is the Hadamard gate. This single qubit gate enables to put a single qubit in a balanced superposition and represents the basic state preparation routine for most of the quantum algorithms. Its matrix form and its effects are the following:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \qquad H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

More generally, for a matrix to represent a valid quantum gate, there is a requirement that must be satisfied.

The normalization condition $|\alpha|^2 + |\beta|^2 = 1$ of an arbitrary qubit's state must hold true also after the application of the gate. This means that the matrix U describing the gate must be unitary:

$$UU^\dagger = \mathbb{I}$$

where $U^\dagger$ is the adjoint of U, obtained by transposing and complex conjugating U. Any matrix that satisfies this condition is a valid single quantum gate. This entails that there are many non-trivial quantum gates. The important notion of universality of the logical NAND gate in classical computation can be achieved also in quantum computation. It has been shown [2] that any 2x2 unitary matrix may be decomposed as:

$$U_g = e^{i\alpha}\begin{bmatrix} e^{-\frac{i\beta}{2}} & 0 \\ 0 & e^{\frac{i\beta}{2}} \end{bmatrix}\begin{bmatrix} cos(\gamma/2) & -sin(\gamma/2) \\ sin(\gamma/2) & cos(\gamma/2) \end{bmatrix}\begin{bmatrix} e^{-\frac{i\delta}{2}} & 0 \\ 0 & e^{\frac{i\delta}{2}} \end{bmatrix} \quad 1.3$$

$$\alpha, \beta, \gamma, \delta \in \mathbb{R}$$

The second matrix is a simple rotation, the first and the third matrices can also be understood as rotations but on different planes. Thanks to this decomposition, any single qubit gate can be expressed as a product of rotations. Introducing the C-NOT gate allows to achieve the concept of universality of quantum gates. The C-NOT gate is a two-qubit gate that applies the following transformations:

$$|00\rangle \to |00\rangle, |01\rangle \to |01\rangle, |10\rangle \to |11\rangle, |11\rangle \to |10\rangle$$

$$|a, b\rangle \to |a, b \oplus a\rangle$$

where $\oplus$ represents the addition modulo 2. In terms of matrix form, the C-NOT gate can be represented as:

$$U_{CN} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

In the same way as the NAND gate is universal in classical computing, it is proven [2] that the set composed of the C-NOT gate and single qubit rotational gates represent a universal set: any multiple qubits gate may be composed from C-NOT and single qubit gates, which in turn, can be decomposed into sequential rotation gates, as shown in Equation (1.3).

In the gate-based quantum computing paradigm, quantum gates are interconnected in order to form quantum circuits. Quantum circuits are models for quantum computation in which a computation is represented as a sequence of single or multiple qubits gates, states preparations and measurements. Figure 2 shows an extremely simple quantum circuit acting on a single qubit. The circuit puts the qubit, initially in the state $|0\rangle$, into the balanced superposition using the Hadamard gate and then measures the outcome.

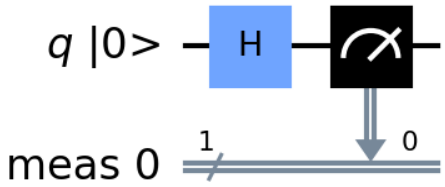

Figure 2: a simple quantum circuit putting a qubit in superposition and measuring the outcome.

Figure 3 shows the results of 1000 runs of the circuit that puts a qubit in superposition. As can be seen, the outcome of a measurement process on a qubit in a balanced superposition is half of the time the basis state $|0\rangle$ and the other half $|1\rangle$. The small difference in the number of times is due to the stochastic nature of the measurement process. Due to the law of large numbers, the higher the number of times the circuit is sampled, the closer the results will be.

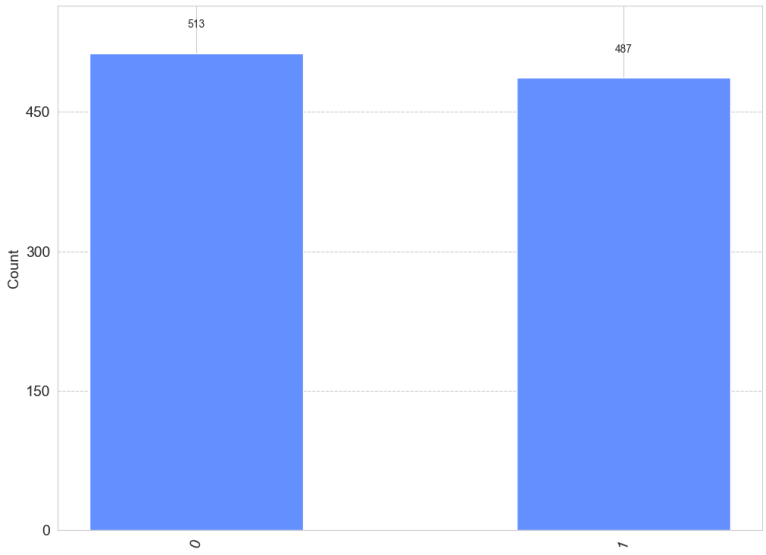

Figure 3: results of the circuit presented in Figure 2 sampled 1000 *times*

Many times, in algorithms, circuits depend on certain parameters that may vary during different iterations of the computation. In these situations, these parametric circuits are called Variational Quantum Circuits (VQC) and represent the fundamental building blocks of many algorithms. VQC can be described as a function $C(\theta)$ that depends on the parameters vector $\theta = \{\theta_0, \theta_1, \dots, \theta_n\}$. Variational Quantum Circuits are also called Adaptive Quantum Circuits, due to their capability of adapting their action during different executions of the circuit.

1.2 Quantum algorithms and quantum search

Quantum algorithms are computations that run on a realistic model of quantum computation. Although other quantum paradigms exist, such as adiabatic quantum computing, in this work the circuit-based model is the one considered. Just like classical algorithms are finite sequences of instructions, quantum algorithms can be decomposed as finite sets of instructions performed by quantum gates, grouped in one or more circuits.
An important finding that should be outlined is the fact that problems that are undecidable in classical computing remain as such also in quantum computing [2]. This entails that quantum computing is not intrinsically more powerful than classical computation. However, in specific cases [3], the algorithms are proven to achieve an exponential speedup, effectively proving the advantage of quantum computation on specific problems, in terms of running time.

Deutsch-Jozsa algorithm

To introduce the main quantum features exploited by quantum algorithms, the famous Deutsch-Jozsa algorithm is presented.
Let us consider a function $f: \{0,1\}^n \to \{0,1\}$, which is known to be either constant, $f(x)$ is either all 1 or all 0, or balanced, $f(x)$ is 0 for exactly half of the possible x and 1 for the other half. The goal is to determine whether the function is balanced or constant using a quantum circuit.

In the classical case, the complexity analysis shows that at worst $2^{n-1}+1$ queries of f are required. In fact, the number of possible x values is exactly 2^n, thus, at least half of the values must be fed to f in order to conclude whether the function is balanced or constant. The Deutsch-Jozsa algorithm shows that in the quantum realm the result can be achieved with $O(1)$ queries of the function f, a quite remarkable result.
All qubits are initialized in the state $|0\rangle$ and an ancilla qubit is initialized in the state $|1\rangle$:

$$|\psi_0\rangle = |0\rangle^{\otimes n} \otimes |1\rangle$$

Hadamard gates are applied to all qubits, resulting in the following state:

$$|\psi_1\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} |x\rangle \otimes (|0\rangle - |1\rangle)$$

A quantum oracle U_f is applied to $|\psi_1\rangle$. The oracle effectively implements f, hence, the action of the oracle U_f is $U_f|x,y\rangle = |x, y \oplus f(x)\rangle$. Applying the oracle transforms the state into:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} |x\rangle \otimes (|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle).$$

By noticing that:

$$|f(x)\rangle - |1 \oplus f(x)\rangle = \begin{cases} |0\rangle - |1\rangle, & f(x) = 0 \\ |1\rangle - |0\rangle, & f(x) = 1 \end{cases}$$

the state can be rewritten as

$$|\psi_2\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle \otimes (|0\rangle - |1\rangle).$$

Hadamard gates are again applied to the n qubits. In Equations below, the ancilla qubit is ignored as, at this point, it is not important for the computation anymore.

$$|\psi_3\rangle = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} \left[\sum_{y=0}^{2^n-1} (-1)^{x \cdot y} |y\rangle \right]$$

$$= \frac{1}{2^n} \sum_{x=0}^{2^n-1} \left[\sum_{y=0}^{2^n-1} (-1)^{f(x)+x \cdot y} \right] |y\rangle$$

where $x \cdot y$ is the sum modulo 2 of the bitwise product. The amplitude of the state $|0\rangle^{\otimes n}$ is $\sum_x \frac{(-1)^{f(x)}}{2^n}$.
In case f is constant, this value is either 1 or -1, depending on the constant value of f. Since $|\psi_3\rangle$ is of unit length, this means that all other states must have amplitudes equal to 0. On the other hand, if f is balanced, the contributions in the amplitude of the state $|0\rangle^{\otimes n}$ cancel out and the measurement will necessarily yield another state. Summarizing, if the measurement outcome is the state $|0\rangle^{\otimes n}$, then f is constant, if it is any other state, then f is balanced. Notice how, with respect to the classical case in which the required number of queries is $2^{n-1}+1$ in the worst case, the quantum version requires a single evaluation of f, carried out by a single application of the operator U_f. The application of an operator to a superposition of states is the concept behind the notion of quantum parallelism and it is a fundamental feature in quantum computation. Intuitively, it can be seen as applying a function to all possible states at once. Another remarkable result of the Deutsch-Jozsa algorithm is that the result is deterministic, meaning that the success is ensured in every run.
However, there are some caveats regarding the Deutsch-Jozsa algorithm. First of all, it has no known practical application. Secondly, the comparison with classical computation is unbalanced, as the method used to evaluate the function f is completely different. Nonetheless, it is an important algorithm that can be used to show quantum features such as interference, quantum parallelism and quantum oracles application.

1.2.1 Grover's algorithm

Diving into the world of quantum search, Grover's algorithm is the most known and famous algorithm for searching an item in an unstructured database. Grover's algorithm is presented in detail here as it constitutes the basic subroutine of the algorithm proposed in this work. In Appendix 6.1, the geometrical intuition behind Grover's algorithm is presented, allowing to better understand the actual functioning of the algorithm by analyzing its geometrical interpretation. Classically, in the worst case, in order to find a specific item in an unsorted database, $O(N)$ steps are required. In the quantum realm, the problem can be solved with $O(\sqrt{N})$ evaluations using Grover's algorithm, achieving a quadratic speedup in the input size. The original presentation of the algorithm supposes that there is a single x in the database such that x is a solution.
As input to the algorithm, suppose to have a function $f: \{0,1,\dots,N-1\} \to \{0,1\}$ where the indices represent the index of the items in the unstructured database and $f(x) = 1 \Leftrightarrow x$ satisfies the search criterion. Analogously to the Deutsch-Jozsa algorithm, a quantum oracle that implements f and is able to mark solutions is required. The goal of the algorithm is to find the unique solution w. The quantum oracle is defined as follows:

$$U_w|x\rangle = (-1)^{f(x)}|x\rangle \tag{1.4}$$

The oracle marks the solution by introducing a negative sign, hence shifting the phase of the solution. The algorithm begins with the register of n qubits set to $|0\rangle^{\otimes n}$. Notice that to address all the N elements of the database, only $n = \sqrt{N}$ qubits are required. The state is initialized in the balanced superpositions of all the indexes of the database:

$$|\psi_0\rangle = H^{\otimes n}|0\rangle^{\otimes n} = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$$

The core of Grover's algorithm consists in the repeated applications of a subroutine called Grover iteration, denoted as G.
The application of the operator G can be broken up into four steps:

1. Apply the oracle U_w;
2. Apply an array of Hadamard gates $H^{\otimes n}$;
3. Perform a conditional phase shift, with every computational basis state except $|0\rangle$ receiving a phase factor of -1. Each state is transformed in $|x\rangle \to (-1)^{\delta_{x0}}|x\rangle$ by applying the operator $2|0\rangle\langle 0| - \mathbb{I}$;
4. Apply a second array of Hadamard gates $H^{\otimes n}$.

It is useful to notice the combined effect of steps $2, 3$ and 4:

$$G_d = H^{\otimes n}(2|0\rangle\langle 0| - I)H^{\otimes n} = 2|\psi\rangle\langle\psi| - \mathbb{I}$$

where $|\psi\rangle$ is the equally weighted superposition of states. This combined operator is often called Grover diffusion operator and is the one responsible for creating constructive interference on the amplitude of the solution state. The complete Grover iteration can therefore be written simply as:

$$G = G_d U_f = (2|\psi\rangle\langle\psi| - \mathbb{I})U_f \tag{1.5}$$

It is proven [4] that the algorithm finds the solution in an expected number of evaluations equal to $O(\sqrt{N})$.
The logic behind the algorithm is to exploit the oracle application and the phase shifting effect in order to create constructive interference on the amplitude that corresponds to the solution. Grover's algorithm can be extended to cases in which multiple solutions are present in the unstructured database. In these cases, the expected number of required evaluations becomes $O\left(\sqrt{N/M}\right)$, where M is the number of solutions.
This algorithm is a fundamental building block of the method proposed in this work. As will be explained in detail, Grover's algorithm is effectively used as a subroutine and the number of times the Grover iteration is

applied is less than $O(\sqrt{N})$ as a certain level of stochasticity in the solutions will turn out to be very important to enable space exploration. In order to better understand the functioning and the complexity of Grover's algorithm, Appendix 6.1 presents the geometric visualization and the geometrical proof of the correctness of the algorithm. Furthermore, Appendix 6.1 shows that applying the Grover iteration a number of times that is greater than the optimum has a negative effect in terms of probabilities of measurements, meaning that the probability of obtaining a solution by measuring the final state, decreases for each Grover iteration applied after having reached the optimum number of applications.

BBHT algorithm

In case the number of solutions M is not known a priori, it is not possible to compute the exact number of iterations that must be applied. Nonetheless, a solution can still be found thanks to an algorithm called BBHT algorithm [5].
This procedure uses Grover's algorithm as a subroutine. The logic is to execute Grover's algorithm multiple times, each time with an increasing number of applications of the Grover iteration:

1. Initialize $m = 1$ and $\lambda \in \left[1, \frac{4}{3}\right]$;
2. Choose j uniformly at random among nonnegative integers smaller than m;
3. Execute Grover's algorithm applying j times the Grover iteration, starting from the initial state $|\psi_0\rangle = \sum_i \frac{1}{\sqrt{N}}|i\rangle$;
4. Measure the register, let i be the outcome;
5. If i is a solution, the problem is solved → exit;
6. Otherwise, set $m = min(\lambda m, \sqrt{N})$, go back to step 2.

The above algorithm is shown to run with a complexity of $O(\sqrt{N/M})$ yielding a solution that is likely to be within a factor $(1 + \pi/c)^2$ of the correct answer [5].

1.2.2 Quantum oracles

Both the Deutsch-Jozsa algorithm and Grover's algorithm assume the existence of specific quantum oracles. Specifically, Grover's algorithm requires an oracle that is able to distinguish a solution. It may seem puzzling to search for an item in an unstructured database with an oracle that seems to already know the solution. However, it is important to notice that there is a clear distinction between finding a solution and verifying a solution: it is possible to do the latter without necessarily being able to do the former.
A quantum oracle can be defined in the simplest way as a reversible quantum mechanical operator that effectively applies a phase flip to the target quantum state $|x\rangle$:

$$O_p|\psi\rangle = \begin{cases} -|\psi\rangle, & \psi = x \\ |\psi\rangle, & \psi \neq x \end{cases} \quad 1.6$$

The definition presented above describes the oracle in terms of a phase-flip oracle. It is also possible to define the oracle as a bit-flip oracle instead. In this case, the operator flips an ancilla qubit if the target state satisfies the condition:

$$O_b|\psi\rangle|y\rangle = \begin{cases} |\psi\rangle|\neg y\rangle, & \psi = x \\ |\psi\rangle|y\rangle, & \psi \neq x \end{cases} \quad 1.7$$

It is always possible to convert a bit flip oracle to a phase flip one by simply wrapping the ancilla qubit with X and H gates, an example of such transformation is presented below.
To understand how quantum oracles work, let us consider a system composed of 2 qubits. The goal is to create an oracle O that is able to recognize the state $|01\rangle$. This means that the action of the oracle is the following:

$$|00\rangle \xrightarrow{O} |00\rangle, \qquad |01\rangle \xrightarrow{O} -|01\rangle, \qquad |10\rangle \xrightarrow{O} |10\rangle, \qquad |11\rangle \xrightarrow{O} |11\rangle$$

It is possible to construct such an oracle with C-NOT, H and X gates.

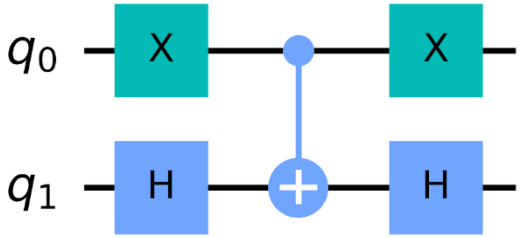

Figure 4: circuit implementing an oracle that recognizes and marks the state $|01\rangle$

The action of the circuit C presented in Figure 4 is:

$$|\psi_0\rangle = |q_0\rangle \otimes |q_1\rangle$$

$$C|\psi_0\rangle = |q_0\rangle \otimes HX_{\neg q_0}H|q_1\rangle$$

where the notation X_q indicates the bit flip conditioned on the state of the qubit q, in other words, the C-NOT controlled by q. To simplify the calculation, notice that:

$$HXH|\psi\rangle = \sigma_z|\psi\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}|\psi\rangle$$

$$H^2|\psi\rangle = \mathbb{I}|\psi\rangle$$

where σ_z is the Z Pauli matrix. The effect of the circuit on the four basis states is:

$$C|00\rangle = |0\rangle \otimes HXH|0\rangle = |0\rangle \otimes \sigma_z|0\rangle = |00\rangle$$

$$C|01\rangle = |0\rangle \otimes HXH|1\rangle = |0\rangle \otimes \sigma_z|1\rangle = -|01\rangle$$

$$C|10\rangle = |1\rangle \otimes H^2|0\rangle = |1\rangle \otimes \mathbb{I}|0\rangle = |10\rangle$$

$$C|11\rangle = |1\rangle \otimes H^2|1\rangle = |1\rangle \otimes \mathbb{I}|1\rangle = |11\rangle$$

The oracle is effectively able to recognize the target state by marking it with a phase factor of -1.
Now, let us consider the bit flip equivalent oracle that acts on a system composed of two target qubits and an ancilla one:

$$|00\rangle \otimes |a\rangle \xrightarrow{O_b} |00\rangle \otimes |a\rangle, \qquad |01\rangle \otimes |a\rangle \xrightarrow{O_b} |01\rangle \otimes |\neg a\rangle, \qquad |10\rangle \otimes |a\rangle \xrightarrow{O_b} |10\rangle \otimes |a\rangle, \qquad |11\rangle \xrightarrow{O_b} |11\rangle \otimes |a\rangle$$

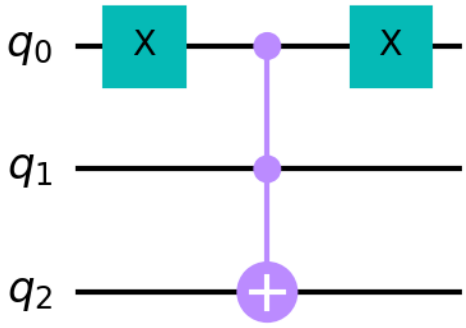

Figure 5: circuit implementing a bit flip oracle that flips the state of the ancilla qubit when the state of the two first qubit is $|01\rangle$

The action of the circuit in Figure 5 is to flip the state of the qubit q_2 when the state of the qubits q_0 and q_1 is $|01\rangle$. The circuit uses the Toffoli gate that is the analogous of the C-NOT gate generalized to 3 qubits: it flips the qubit q_2 when the two control qubits are equal to $|1\rangle$. The goal is to prove that this oracle can be converted into a phase flip oracle with the same effect as the one shown above.
To convert this oracle to a phase flip oracle, it is enough to wrap the ancilla qubit in X and H gates:

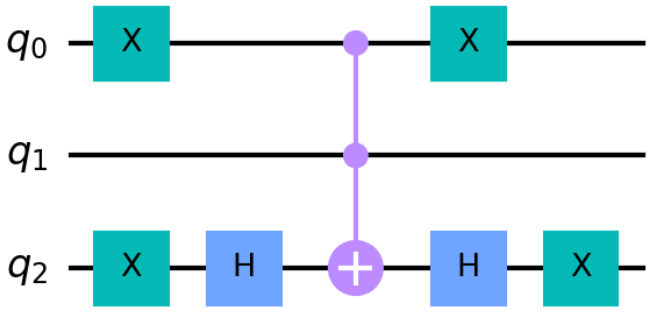

Figure 6: conversion of a bit flip oracle to a phase flip oracle

The q_2 qubit is always initialized in the state $|-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$. Now the circuit C_1 in Figure 6 has the following effect:

$$C_1|\psi_0\rangle = |q_0 q_1\rangle \otimes XHX_{\neg q_0, q_1} HX|-\rangle.$$

In this case the $X_{a,b}$ notation indicates the Toffoli gate where a and b are the control qubits. Noticing that:

$$(XH)^2 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \qquad (XH)^2|-\rangle = |+\rangle$$

$$XH^2X = \mathbb{I}$$

the actions on the basis states are:

$$C_1|00\rangle \otimes |-\rangle = |00\rangle \otimes XH^2X|-\rangle = |00\rangle \otimes |-\rangle$$

$$C_1|01\rangle \otimes |-\rangle = |01\rangle \otimes (XH)^2X|-\rangle = |01\rangle \otimes (-(XH)^2|-\rangle) = -|01\rangle \otimes |+\rangle$$

$$C_1|10\rangle \otimes |-\rangle = |10\rangle \otimes XH^2X|-\rangle = |10\rangle \otimes |-\rangle$$

$$C_1|11\rangle\otimes|-\rangle = |11\rangle\otimes XH^2X|-\rangle = |11\rangle\otimes|-\rangle$$

As can be noticed, the oracle has added a phase factor of -1 to the target state as well as flipping the ancilla qubit in its basis. Considering the subsystem composed only of the first two qubits, the action of this oracle and the one presented in Figure 4 are equivalent. Although in the above-mentioned case the oracle may seem easy to implement, it is worth noticing that the structure is completely problem-dependent. Quantum oracles may be very hard to implement in practice. The great number of recent publications regarding how to automatize the creation of quantum oracles [6] [7] proves that the task of devising quantum oracles for different problems is a difficult task. There are also tools [8] that allow synthetizing logical conditions into quantum oracles.

Grover's algorithm is completely dependent on the definition of its quantum oracle in order to find the desired solution. In real-world optimization problems, the classical function that can be used to recognize a solution can be extremely complicated. Considering for example classes of problems such as CFD (Computational Fluid Dynamics) and system optimization in which solutions are computed using complicated differential equations, it is clear how any attempt to translate a function of this kind into the quantum realm as a quantum oracle is extremely complicated, if not impossible in some cases.

Partial oracles

Identifying an oracle by the pattern of the different bit strings that it marks is very suitable for identifying the number of solutions that the oracle can recognize. Let us consider a system of dimension n, solutions are identified by bit strings of length n, thereby the number of possible strings is 2^n. Suppose now the number of solutions is $m < 2^n$. A partial oracle recognizes as solutions a superset S of elements that has a size larger than m: $|S| > m$. This means that the oracle may mark an element as solution even if it is not. The greater the difference $|S| - m$ is, the less precise the oracle is. Obviously, in the case in which $|S| = m$, being S a superset by definition, the set S coincides with the set of true solutions, making the oracle a perfect one.

Let us consider now the interesting and common situation in which there is a single solution s, hence $m = 1$. In this situation the perfect oracle $\tilde{O}$ would be the one that recognizes only the bit string $s = (s_0 s_1 \dots s_{n-1})$:

$$\tilde{O}|\psi\rangle = \begin{cases} -|\psi\rangle, & \psi = s \\ |\psi\rangle, & \psi \neq s \end{cases}$$

The symbol ψ is identifies the binary string represented by the quantum state in binary encoding. The equality $\psi = s$ means that the quantum state is represented by the binary string that is equal to the solution bit string. Defining as K the set of indexes the partial oracle is able to recognize, it is possible to define the partial oracle as:

$$O|\psi\rangle = \begin{cases} -|\psi\rangle, & \psi_i = s_i, \quad \forall i \in K \\ |\psi\rangle, & \psi_i \neq s_i, \text{ for some } i \in K \end{cases}$$

Reasoning in terms of bit strings, the quantum oracle marks all the states having the correct bit value in all the positions that the oracle recognizes. For example, consider a system of 6 qubits, in which the solution s is identified by the bit string 000111. A partial oracle could be defined as an operator that marks all states having the following shape:

$$00 ** 11$$

in which the symbol $*$ is used to identify any of the two possible basis states of the qubit. In this situation, it can be stated that the oracle recognizes only certain features, in particular the features in positions $\{0,1,4,5\}$.

More generally, when there is a single solution s, the pattern of solutions a partial oracle recognizes can be identified by its pattern string $\tilde{s}$:

$$\tilde{s}_i = \begin{cases} s_i, & i \in K \\ *, & i \notin K \end{cases} \qquad 1.8$$

where the set K is the set of the indexes of features the oracle recognizes.
A very simple but functional parameter that identifies the precision of a partial oracle, is the ratio:

$$\varepsilon = \frac{m}{2^{n-|K|}} \tag{1.9}$$

where m is the number of true solutions. It is easy to see how the oracle is optimal in case $\varepsilon = 1$. When there is a single solution:

$$\varepsilon = 1 \Leftrightarrow |K| = n$$

This means that an optimal oracle recognizes all features of solutions. In other words, an optimal oracle does not have any unknown symbol in its pattern representation. Note that the number of solutions recognized by a partial oracle is the denominator of Equation (1.9). Basically, the precision of an oracle is identified by the number of true solutions divided by the total number of bit strings that can be generated using all the indexes that are not in K.

2 Elite quantum diffusion recombination

As shown in Section 1.2.2, quantum oracles can be very difficult to create in a real-world scenario. The circuits that represent them are often variational circuits, as oracles may need to adapt to different stages of the computation and may depend on parameters that can change during runtime. The goal of this work is to ease the burden placed on quantum oracles due to their difficult practical implementation. Exploiting notions from evolutionary computation, it is possible to devise an algorithm that can optimize systems even in cases in which oracles recognize only a subset of the features of the solutions. The framework that enables to extend quantum search algorithms to fitness-based optimization problems has been called EQDR (Elite Quantum Diffusion Recombination) procedure. The name of the algorithm refers to the various modules used:

- Elite: genetic knowledge of best individuals from various iterations is gathered during the repeated runs of Grover's algorithm;
- Quantum: the algorithm uses the quantum Grover search algorithm as a subroutine;
- Diffusion: as will be explained better in later sections, the procedure effectively builds a vector of bits that is "diffused" into the superposition returned by Grover's algorithm, biasing the search toward the solution represented by the vector;
- Recombination: the procedure composed of a variational quantum operator and a classical computation to calculate the parameters of the operator, acts as a classical recombination operator that comprehends both crossover and mutation. The computation of the diffusion vector is carried out in the classical world taking inspiration from classical crossover operators, while mutation is applied with a series of rotational variational gates to each single qubit.

Although the EQDR procedure is not properly an EA (Evolutionary Algorithm), there are some important concepts that are shared between the two worlds. In Section 2.2, analogies and differences between the two frameworks are explained, highlighting how and why EQDR borrows certain features of EAs. To begin with, a problem solved with EAs needs to be modeled considering some requirements and fundamental concepts:

- Individuals: an individual represents a candidate solution to the optimization problem. Individuals can be modeled in different ways, depending on the problem. An individual is represented by its chromosome, a set of parameters $\{x_i\}_i$. These parameters enable the construction of the phenotype intended as the concretization of a solution of the optimization problems. The phenotype is then evaluated by means of a certain function.
- Population: the population is the set of individuals considered at a certain point in time during the computation. It represents a subset of the possible solutions to the problem.
- Generation: a generation consists of a population created at a given point of the evolutionary algorithm. Generations proceed and older populations generate new populations by altering, recombining and introducing new individuals that substitute others, refining the quality of future solutions.

Individuals are evaluated by means of a fitness function. This function is a general mathematical function $f: \mathbb{R}^n \to \mathbb{R}$ that, given a chromosome g of length n, calculates how close the solution is to achieve the set of predefined aims. Fitness functions can have any shape and, guiding the evolution of the solutions during the generations, have the greatest impact on how good the final solutions are and how the populations evolve. The most general structure of an evolutionary algorithm is depicted in Figure 7.

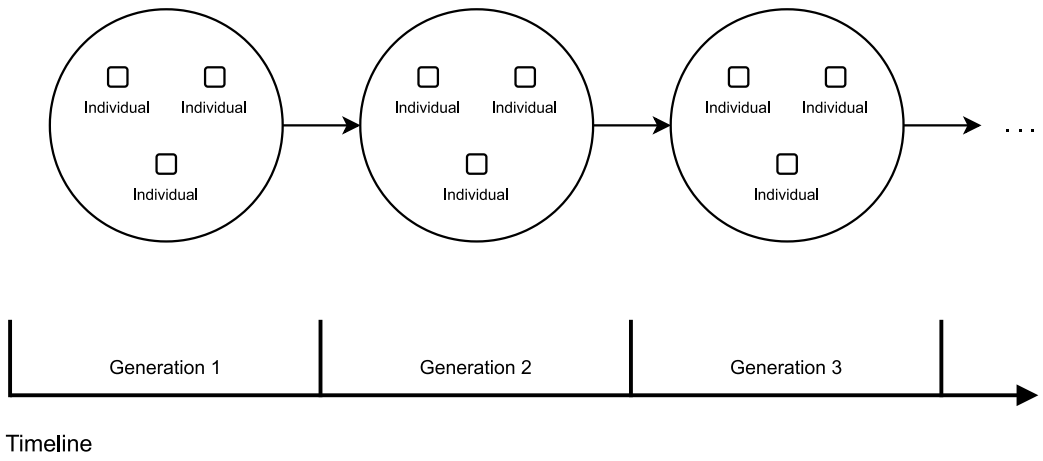

Figure 7: the most general structure of an evolutionary algorithm. Individuals are grouped in populations. Each generation, a population is evaluated by means of the fitness function f and a new population is generated by recombining individuals.

2.1 EQDR algorithm

The algorithm proposed in this work can be effectively deconstructed in two main parts. The first part involves the application of a vanilla Grover's algorithm. The Grover quantum oracle U_f has the only requirement to recognize a superset of all the solutions of a certain problem. The state $|\psi\rangle$ that Grover's algorithm outputs is supposed to be a quantum superposition in which amplitudes that correspond to the superset of the recognized solutions have a greater magnitude. The state $|\psi\rangle$ returned by Grover's algorithm is not measured, instead it is passed to the second part. This second module is comprised of the recombination and the mutation step. The final step is the measurement of the state and the evaluation of the obtained result with the classical fitness function. The genetic knowledge gathered with measurements of different generations is used to guide the evolution by altering the amplitudes of the superposition. The process of altering the amplitudes of the superposition is called diffusion in the EQDR framework. This nomenclature comes from the fact that in order to enforce a known state to a certain qubit in an unknown state, it is possible to use a single Grover iteration on a two-dimensional subsystem spawned by the target qubit and an ancilla one, as will be explained in Section 2.3. The name is borrowed from the Grover Diffusion Operator which, in this case, is applied only once.
The job of the EQDR procedure is to exploit the genetic knowledge of the best individuals measured in previous generations to compute the vectors $\hat{b}$ and $\hat{a}$ and use them to alter the amplitudes of the superposition. The first is a binary vector in which the element $\hat{b}_i$ identifies the classical state that must be diffused into the i-th qubit of the superposition $|\psi\rangle$, while the latter is the accuracy vector. The values $\hat{a}_i$ identify the certainty levels of the bits $\hat{b}_i$. The accuracy vector plays an important role in avoiding the problem caused by the determinism of measurement of a diffused superposition, presented in detail in Appendix 6.2. In fact, each bit $\hat{b}_i$ is not deterministically enforced into the corresponding qubit, but instead the application of the operator is stochastic, with a probability proportional to the accuracy value $\hat{a}_i$.
Finally, the modified state undergoes a mutation step composed of a series of random variational rotational gates. The state is measured and the fitness $f(x)$ of the measured classical chromosome contributes to the inter-generational genetic knowledge. The structure of the whole algorithm is shown in Figure 8 below.

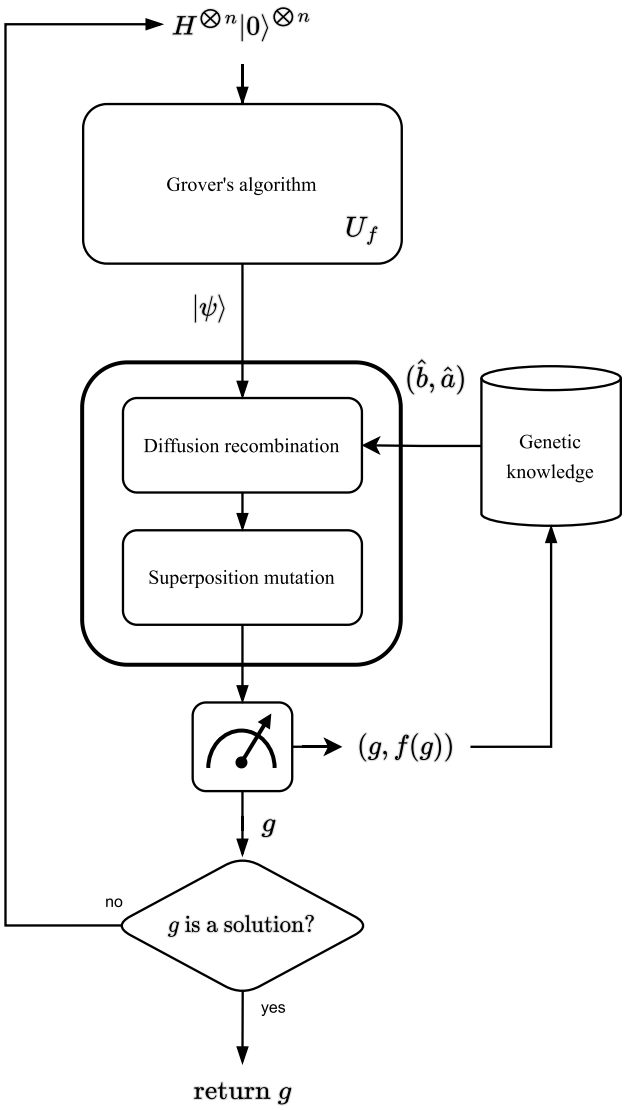

Figure 8: the structure of the proposed algorithm. The EQDR procedure is responsible for altering the state $|\psi\rangle$ returned by Grover's algorithm, exploiting the genetic knowledge acquired during different iterations. After the diffusion step, the EQDR procedure introduces stochasticity by applying the rotational variational gates that act as the classical mutation operator in evolutionary computation.

As explained in Section 1.2.1, the number of solutions M determines how many Grover iterations must be executed to obtain a solution with high probability. However, being U_f a partial oracle, the value of M is not known a priori. Recall, from Section 1.2.1, that the BBHT algorithm can be used in order to find a solution with relative high probability using Grover's algorithm as a subroutine, maintaining the complexity as $O\left(\sqrt{N/M}\right)$. Therefore, in the proposed algorithm the exact same procedure of the BBHT algorithm can be used, with the only difference that before each measurement the EQDR applies the diffusion recombination and the superposition mutation.

Notice that the EQDR algorithm does not require Grover's algorithm to output an exact solution, even though it may happen. The oracle used is partial, hence it recognizes a superset of the solutions. In order to enhance space exploration, a higher degree of non-determinism during Grover's algorithm step may be beneficial in the EQDR procedure. This translates in the fact that, the hyperparameter $\lambda \in [1, 4/3]$ that describes the rate of growth of the number of Grover iteration after each run, can be set very close to 1. This means that the number of Grover iterations will increase very slowly each iteration, effectively speeding up the framework. The modification of this parameter is shown not to alter the complexity of the BBHT algorithm, provided it is inside the boundaries [5].

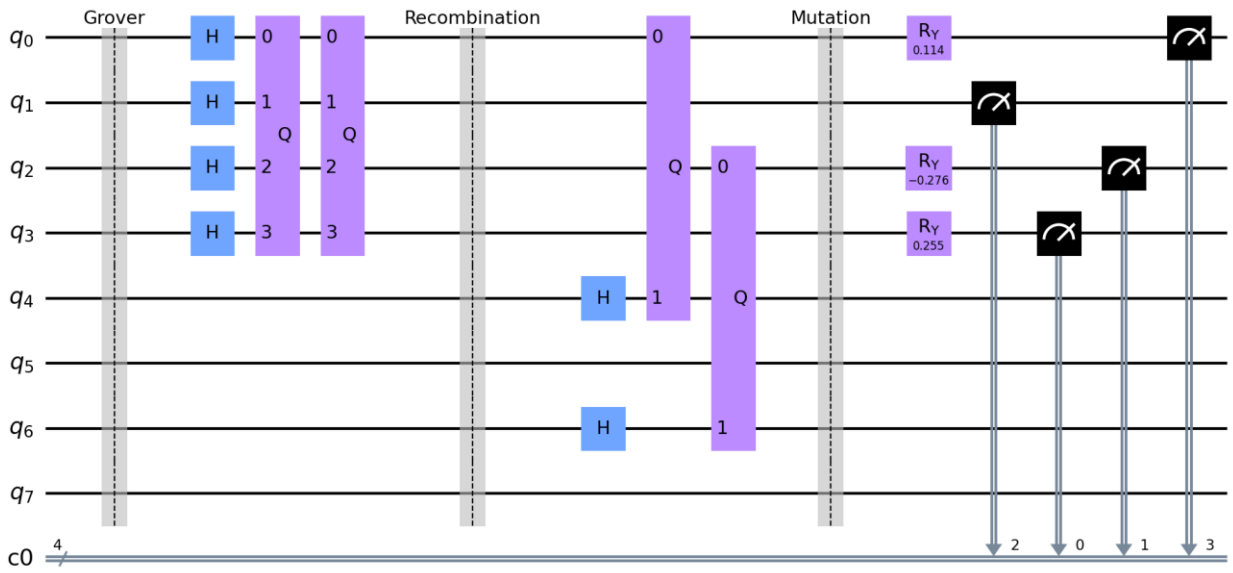

Figure 9: the circuit representation of the quantum section of the EQDR algorithm in case of chromosomes of length 4. The number used in the mutation step are random and do not represent fixed parameters. Random values have been used for displaying purposes.

Figure 9 displays the circuit representation of the quantum section of the EQDR algorithm in case of chromosomes of length 4. Notice how, in the recombination step, not all qubits have undergone the alteration process. This is due to the fact that, as explained in detail in Section 2.3, each qubit has a certain probability of being altered by the process, proportional to the accuracy vector element $\hat{a}_i$.
The number of Grover iterations is dictated by the procedure of the BBHT algorithm. In this case the circuit is representing a Grover's algorithm with 2 iterations. Figure 10 shows the decomposed gates of the vanilla Grover iteration. As can be seen, the oracle is applied to the first 4 qubits, which are the ones that represent the chromosome. The remaining ones are the ancillas used for the diffusion recombination.

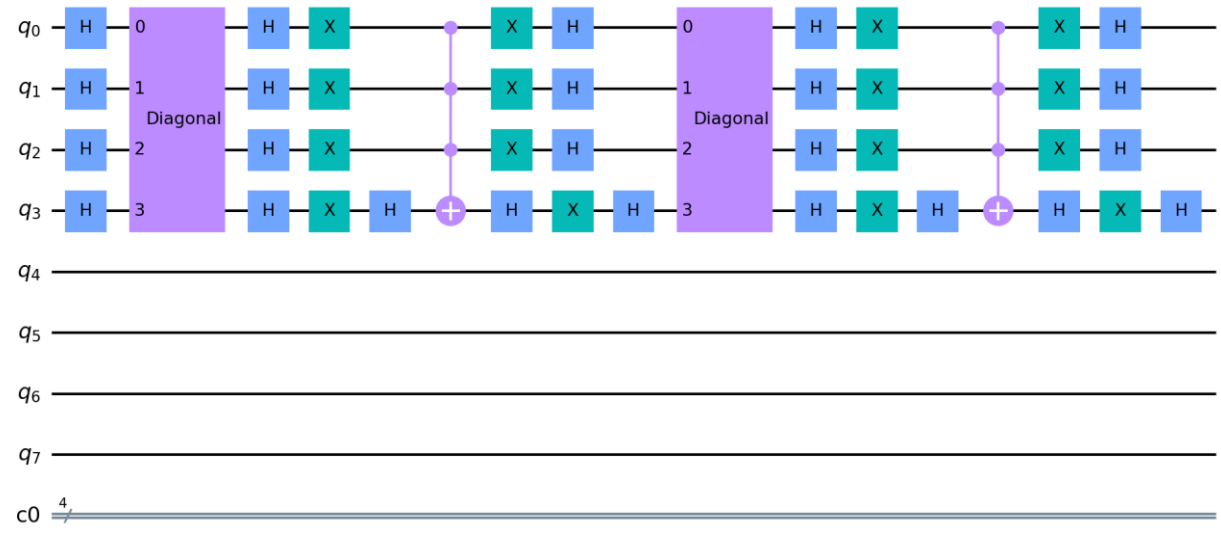

Figure 10: insight of the vanilla Grover's algorithm subcircuit. In this example there are two Grover iteration in the circuit.

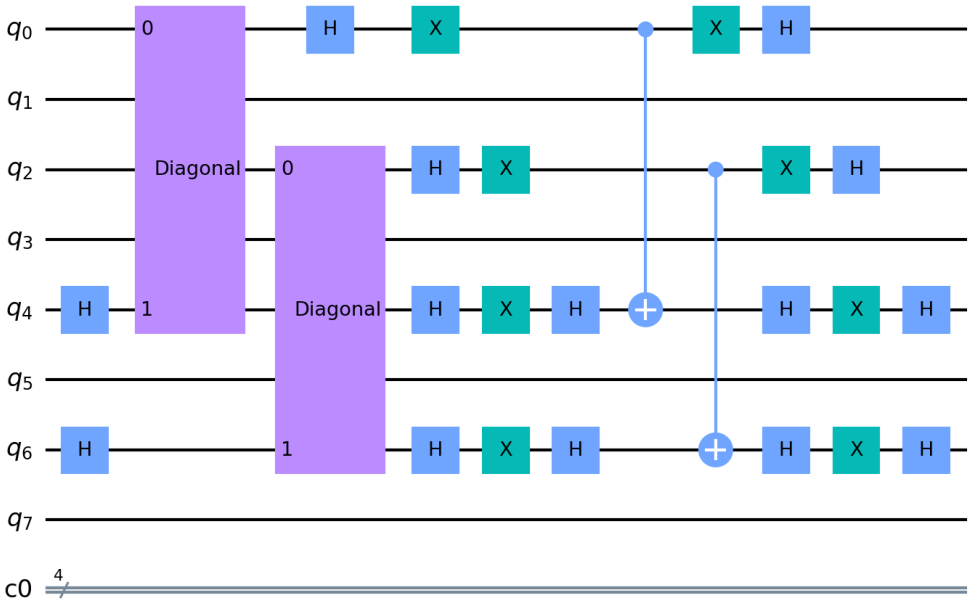

Figure 11: details of the recombination procedure of the EQDR algorithm. The adaptive diffusion recombination has been applied to qubit 0 and 2. Ancillas are therefore qubits 4 and 6 and are initialized in the state $|+\rangle$.

In Figure 11, it is shown in detail the decomposition of the adaptive diffusion recombination step. In this case, the example is showing the recombination step applied exclusively to qubits 0 and 2, with the corresponding ancillary qubits 4 and 6. Each gate labeled as Q represents an application of a Grover iteration to the indicated qubits. The ancillas are being initialized in the state $H|0\rangle = |+\rangle$. Finally, the mutation step slightly rotates some of the qubits with a certain mutation probability and with an angle proportional to the mutation amplitude. In the recombination step, it is possible to notice that the Grover iterations involve only two qubits.
How the vectors $\hat{b}$ and $\hat{a}$ are computed, is explained in detail in Section 2.5. The logic behind the vector $\hat{b}$ is to effectively represent a state derived from the genetic knowledge acquired. In this sense, the classical state represented by $\hat{b}$ can be seen as an intermediate offspring state generated from all the classical states stored in the genetic knowledge. All the solutions in the inter-generational set can be associated to the classical parents in an evolutionary algorithm. The way in which the vectors $\hat{b}$ and $\hat{a}$ are computed, is transparent to the EQDR procedure. Any classical black box that takes as input the genetic knowledge and outputs a classical state $\hat{b}$ and an accuracy vector $\hat{a}$, both of length n, can be used. This high level of separation between the quantum section of the algorithm and the classical one allows the proposed framework to adapt to various problems. Adapting either the quantum part and its oracle or modifying how the diffusion vector is classically computed, it is possible to change the way the genetic knowledge regarding the inter-generational elites is used. An important consideration is the fact that, while classically the offspring generated actually represents a solution that may or may not undergo a mutation step, in the quantum world the offspring is a state in which each of the n elements $\hat{b}_i$ corresponds to the single qubit state that will be probabilistically enforced to some of the qubits in the quantum register.

2.2 Evolutionary computation analogies

In the EQDR procedure, the population is identified by all the possible solutions in terms of bit strings. Let us consider a set of n bits, the number of possible bit strings is $2^n = N$. In the quantum world, creating a superposition between only certain elements of all the 2^n elements is extremely difficult. Oppositely, creating the balanced superposition of all the possible bit strings can be efficiently carried out by applying n Hadamard gates, one for each qubit. This entails that, at any given generation, the population is constituted by all the possible individuals. In a classical environment, this means losing the effectiveness of the parameter that

identifies the population size. Larger populations usually involve faster exploration capabilities at the cost of higher computational power requirements as the initial genetic material is composed of many of the possible chromosomes and the fitness function must be calculated for each of them.
In the EQDR procedure, the problem related to the increased computational complexity is mitigated by the fact that the search is carried out by Grover's algorithm. Thanks to quantum features, Grover's algorithm effectively weights the amplitudes of the complete superposition, reducing the probability of obtaining an extremely bad solution as the outcome of the measurement process. Due to this consideration, it would be more correct to associate the classical population with the biased superposition of states that Grover's algorithm outputs. In this case, the generation can be seen as formed by some kind of probabilistic population. The classical analogous would be to have the pool of all possible candidates, but only probabilistically select some of them by means of some probability distribution, to carry out the recombination process in order to generate a new population. One of the strengths of the EQDR algorithm lies in this concept. The quantum oracle is the quantum analogous of the classical probability distribution presented above. A quantum oracle that is able to identify a superset of the solutions, can be seen as a probability distribution biased towards the elements in that set.

In classical EAs, the alteration of the solutions is carried out by crossover and mutation. Crossover identifies the process of recombining existing solutions to create a new one, while mutation is the process that introduces new genetic material into the population. There cannot exist an EA that uses solely the crossover step to generate new solutions. In that case, in fact, the genetic material, intended as all the combinations of possible parameters, would depend only on the initial population, as the crossover step simply shuffles the existing solutions. On the other hand, EAs that employ only mutation are possible. Nonetheless, crossover and mutation are complementary and better if used in synergy. Related to crossover and mutation are the two concepts of exploitation and exploration. The first refers to the capability of a certain algorithm to exploit the good solutions already found, attempting to refine them. The latter identifies how fast an algorithm can look for other possible solutions in the whole search space. Exploration and exploitation are two opposite concepts: whenever an algorithm is focusing on exploitation, it means that resources are being employed to refine existing solutions, at the cost of a worse ability to look around for other solutions. An efficient EA must balance the two concepts and, possibly, adapt to the various stages of the search to avoid getting stuck in local optima. Usually, whenever the population is converging too fast, exploration must be enhanced at the cost of exploitation, in order to avoid all individuals to converge to the exact same chromosome.
Classical operations to perform crossover and mutation cannot be applied to a superposition state $|\psi\rangle$ without measuring the state, hence destroying the information. Quantum mechanics is inherently probabilistic, meaning that, to implement mutation, one simple possibility is to directly act on the amplitudes of the superposition, altering them in order to introduce a controlled degree of stochasticity towards a biased result. From a bigger point of view, the effect of mutation in the quantum realm is analogous to the classical mutation, with the only difference being that in the quantum world the stochasticity is being introduced in an already probabilistic environment. Mutation in the EQDR procedure is explained in detail in Section 2.4.
Regarding crossover, a novel and completely different approach must be developed. As explained before, crossover step involves recombining certain existing solutions to generate a new chromosome. In the EQDR procedure, Grover's algorithm outputs a single state $|\psi\rangle$ that identifies the superposition of all possible solutions, weighted differently. It is not possible to know a priori which solutions are more probable without applying repeated measurements to different copies of the state $|\psi\rangle$. The EQDR procedure approaches the problem of recombination under a different point of view, borrowing the concept of elitism from EAs.
Classically, this term identifies a strategy employed when a new population is generated and individuals must be replaced. An EA that employs elitism will allow certain individuals, called elites, to be inserted directly into the next population, without undergoing any recombination procedure. The idea is to allow very good solutions of older populations to be inserted into new ones without altering the effectiveness of their chromosomes.
Similarly, the EQDR procedure gathers genetic knowledge at each iteration by measuring the superposition returned from Grover's algorithm. During subsequent iterations of the algorithm, this global genetic knowledge is used to design and apply a variational quantum operator, presented in Section 2.3. This operator

biases the superposition of states towards an ensemble of the best genetic material gathered. In this sense, the term elitism is referred to the global genetic knowledge, hence the best solutions, gathered during runtime.

2.3 Adaptive diffusion recombination

Grover's algorithm outputs a superposition state $|\psi\rangle$ in the diffusion recombination step the amplitudes are modified exploiting the diffusion vector $\hat{b}$. This step can be effectively recognized as the quantum version of the classical crossover operator. As explained in Section 2.2, the population in the EQDR procedure is identified by the superposition $|\psi\rangle$, in which each basis state represents a possible individual. The main difference between a classical crossover and the proposed diffusion recombination, is the fact that the latter does not exploit the individuals of the current population. Instead, the parents are selected from the genetic pool in which best individuals are inserted during different generations.
The goal of the diffusion recombination step is to act on the different amplitudes of the superposition $|\psi\rangle$ in such a way that the measurement will output a good individual with high probability. The diffusion vector $\hat{b}$ can be seen as an encoding of a mixture of the good solutions present in the genetic pool. Hence, the goal is to modify the amplitudes of the quantum state $|\psi\rangle$ by biasing the value of the i-th qubit in the quantum register towards the value $\hat{b}_i$.

The problem of rotational gates for state biasing

To modify the amplitude of a single qubit, rotational gates are often used. However, the effect of rotational gates on a certain qubit, depends on the relative phase of the qubit's state. This is a problem when the state of the qubit is unknown.

As an example, let us consider a qubit in the state $|\phi\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle$. The probabilities of the different outcomes are $P(0) = \frac{3}{4}$, $P(1) = \frac{1}{4}$. Applying a rotational gate $R_y(\theta)$ with $\theta = \frac{2\pi}{3}$, causes the qubit state to become the basis state $|1\rangle$, effectively changing the probabilities to $P(0) = 0$, $P(1) = 1$. The rotational gate can be expressed in the following matrix form:

$$R_y(\theta) = \begin{bmatrix} cos\left(\frac{\theta}{2}\right) & -sin\left(\frac{\theta}{2}\right) \\ sin\left(\frac{\theta}{2}\right) & cos\left(\frac{\theta}{2}\right) \end{bmatrix}. \quad 2.1$$

The application of the gate proceeds as:

$$R_y\left(\frac{2\pi}{3}\right)|\phi\rangle = \begin{bmatrix} \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{2} \\ \frac{1}{2} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle.$$

In this case, it is possible to modify the state by simply applying a rotational gate. It is worth noticing that, with a smaller angle θ, it is possible to maintain a certain degree of stochasticity in the measurement outcomes. This means that the final state will not coincide with $|1\rangle$, but instead it will yield it with high probability.
Let us consider the situation in which the initial state $|\phi\rangle$ of the qubit is not known. The goal is still to move the state toward the basis state $|1\rangle$. The sign of the angle θ identifies the direction of rotation. In this case, it is not possible to know in which direction to rotate the qubit, since the state is unknown. In the example presented above, a positive θ rotates the qubit towards the basis state $|1\rangle$ while a negative θ pushes the qubit towards $|0\rangle$. However, this is not always true, as the relative phase of an arbitrary qubit state determines the sign of the θ required to move the qubit toward the basis state $|1\rangle$.

In case the initial state of the qubit is $|\phi\rangle = \frac{\sqrt{3}}{2}|0\rangle - \frac{1}{2}|1\rangle$, applying the same rotational gate as the one in the example above, causes a rotation in the wrong direction:

$$R_y\left(\frac{2\pi}{3}\right)|\phi\rangle = \begin{bmatrix} \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & \frac{1}{2} \end{bmatrix}\begin{bmatrix} \frac{\sqrt{3}}{2} \\ -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}}{2} \\ \frac{1}{2} \end{bmatrix} = \frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle.$$

The coefficients obtained, demonstrate that the initial state has been biased toward the wrong basis state. The goal was to push the initial qubit state $|\phi\rangle$ towards the basis state $|1\rangle$. However, both the states $|\phi\rangle$ and $R_y\left(\frac{2\pi}{3}\right)|\phi\rangle$ have the same probabilities of measurement outcomes, meaning the rotated state did not result in the correct state. Inverting the sign of the angle θ and thus applying the rotational gate $R_y\left(-\frac{2\pi}{3}\right)$, causes instead the desired result:

$$R_y\left(-\frac{2\pi}{3}\right)|\phi\rangle = \begin{bmatrix} \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} \end{bmatrix}\begin{bmatrix} \frac{\sqrt{3}}{2} \\ -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix} = -|1\rangle \equiv |1\rangle.$$

This shows how the sign of the angle θ is fundamental when dealing with unknown states. In this case, if the angle θ is positive, the qubit is not biased toward the state $|1\rangle$. This causes the measurement probabilities to be the same as the ones before applying the rotational gate. Geometrically, this can be explained by considering the state $|\phi\rangle = \frac{\sqrt{3}}{2}|0\rangle - \frac{1}{2}|1\rangle$ on the Bloch's sphere. This state has a 60° angle from the basis state $|0\rangle$. Applying an anticlockwise rotation of 60° around the y axis, transforms the state into another one having 60° angle from the basis state $|0\rangle$ but on the other side of the axis. Figure 12 below presents the geometrical interpretation on the Bloch's sphere of the two states.

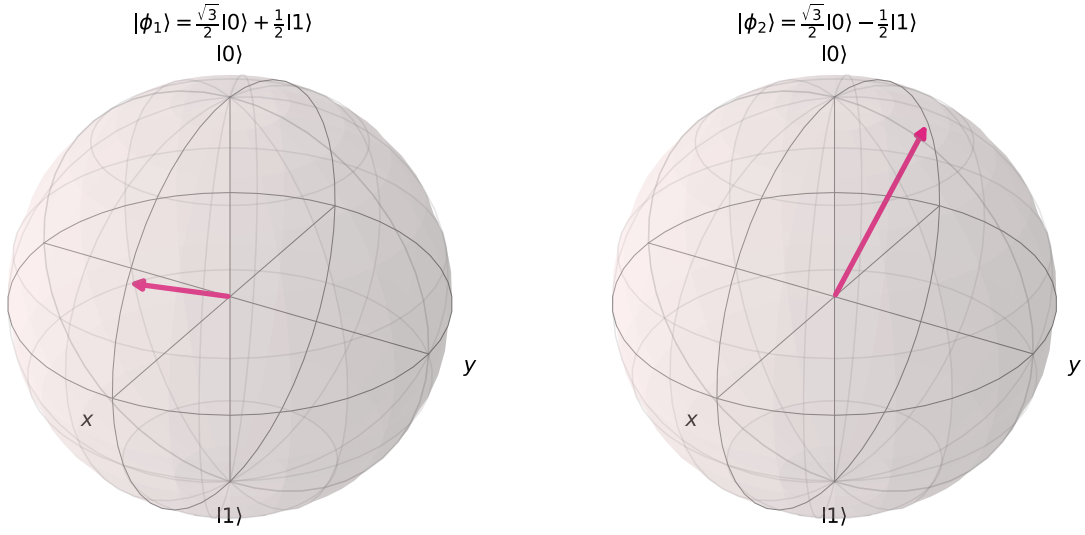

Figure 12: the relation between two qubits states. On the left the state $|\phi_1\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle$, on the right the state $|\phi_2\rangle = \frac{\sqrt{3}}{2}|0\rangle - \frac{1}{2}|1\rangle$ plotted on the Bloch's sphere. The state $|\phi_1\rangle$ can be obtained by rotating the state $|\phi_2\rangle$, applying the gate $R_y\left(\frac{\pi}{3}\right)$. Applying the same gate to the first state, causes the qubit to become the basis state $|1\rangle$. Hence, the effect of the rotational gate depends on the initial state of the qubit and depends on the sign of the angle θ.

The situation described above, highlights how the effect of the rotational gate depends on the initial state of the qubit. In the EQDR algorithm, the state of each qubit is the output of a Grover's algorithm. Each state is

unknown and cannot be measured without collapsing the superposition of states $|\psi\rangle$. There is the need for another method to push the qubit toward a target state independently of its initial state.
To introduce the diffusion recombination procedure, notice that the goal of a generic Grover's algorithm is to modify the amplitudes of a uniform superposition of states to amplify the probability of measuring a certain target state. The Grover's algorithm is therefore the perfect candidate to push a certain element toward a target state, independently of the initial state. The diffusion recombination operator applies a single Grover iteration to a two-dimensional subsystem composed of a qubit of the quantum register in an arbitrary state and an ancilla qubit. There are n different instances of Grover's algorithms each executed with a single iteration. Each instance runs on one of the n two dimensional subsystems. Since there are n qubits in the system, meaning the chromosome is of length n, the number of required qubits is $2n$.
However, to run a Grover iteration, there is the need of an oracle able to distinguish a solution. A two-dimensional system can be biased toward one of the four basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$. An oracle able to distinguish one of the basis states can be defined as:

$$O = \mathbb{I} - 2|t\rangle\langle t|$$

where $|t\rangle \in \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ is one of the four basis states.
Depending on which state is selected as target state, the oracle O_i of the i-th subsystem changes. The value of the target state $|t\rangle$ depends on the value of the i-th element of the diffusion vector $\hat{b}$. As a convention, the oracle is defined as follows:

$$O_i = \mathbb{I} - 2\left(|\hat{b}_i\rangle\otimes|\hat{b}_i\rangle\langle\hat{b}_i|\otimes\langle\hat{b}_i|\right) = \begin{cases} \mathbb{I} - 2|00\rangle\langle 00|, & \hat{b}_i = 0 \\ \mathbb{I} - 2|11\rangle\langle 11|, & \hat{b}_i = 1 \end{cases}. \qquad 2.2$$

Analyzing the diffusion recombination for the single i-th two-dimensional subsystem, the ancilla qubit is initialized in the state $H|0\rangle = |+\rangle$ while the qubit does not undergo any initialization procedure, maintaining the unknown initial state $|\phi\rangle$. The initial state of the system is:

$$|\psi_0\rangle = |\phi\rangle\otimes|+\rangle.$$

A single Grover iteration $G_i = (2|\psi\rangle\langle\psi| - \mathbb{I})O_i$, where $|\psi\rangle$ is the balanced superposition of a two-qubit system, is applied to the initial state:

$$|\psi_1\rangle = G_i|\psi_0\rangle = (2|\psi\rangle\langle\psi| - \mathbb{I})O_i|\psi_0\rangle.$$

At this point, if the state $|\psi_1\rangle$ is measured, the outcomes will depend on the oracle O_i used. Writing the initial state of the i-th qubit as $|\phi_i\rangle = \alpha|0\rangle + \beta|1\rangle$, the probabilities are:

- If $O_i = \mathbb{I} - 2|00\rangle\langle 00|$

$$P(00) = \left|\frac{2\alpha + 2\beta}{2\sqrt{2}}\right|^2, \qquad P(01) = \left|\frac{-2\alpha + 2\beta}{2\sqrt{2}}\right|^2, \qquad P(10) = P(11) = 0$$

- If $O_i = \mathbb{I} - 2|11\rangle\langle 11|$

$$P(10) = \left|\frac{2\alpha - 2\beta}{2\sqrt{2}}\right|^2, \qquad P(11) = \left|\frac{2\alpha + 2\beta}{2\sqrt{2}}\right|^2, \qquad P(00) = P(01) = 0.$$

The in-depth explanation of the state evolution during the recombination is presented in Appendix 6.2.
Leaving aside the ancilla and considering the state of the first qubit in the two-dimensional subsystem, it is possible to notice how the measurement process is deterministic. The application of a single Grover iteration to a two-dimensional subsystem is enough to push the qubit's state towards a target state with probability 1. From a certain point of view this feature is positive as ensures that the i-th element of the diffusion vector $\hat{b}$ is enforced into the qubit state. On the other hand, the total lack of stochasticity is not optimal, as the goal is to

probabilistically generate good solutions. Deterministically enforcing the diffusion vector into the quantum register would cause the same state to be measured at each generation.
This problem can be resolved by exploiting the accuracy vector $\hat{a}$, presented in the previous section, to prevent deterministically diffusing each element of the diffusion vector $\hat{b}$. Each element of the diffusion vector has a certain probability $p = \hat{a}_i$ of being enforced into the corresponding qubit. Being a vector of probabilities, the accuracy vector has the following constraints:

$$\hat{a}_i \in \mathbb{R}, \hat{a}_i \in [0,1] \; \forall i.$$

Each element $\hat{a}_i$ represents the probability of diffusing the bit $\hat{b}_i$ into the i-th qubit of the register. These probabilities must be somehow connected to how the diffusion vector $\hat{b}$ is computed. Some values $\hat{b}_i$ may have a higher accuracy. This might happen, for example, in case multiple chromosomes in the elites have the same value at the i-th position in the chromosome. In Section 2.5, various methodologies to compute the diffusion vector are proposed, along with the corresponding procedure to obtain the accuracy vector.

2.4 Superposition mutation

One of the fundamental purposes of the EQDR algorithm is to enable the use of partial quantum oracles. A partial quantum oracle is an oracle that can recognize a superset of solutions of a problem. Running a pure BBHT with a partial oracle is equivalent to running an algorithm that is comparable to a random sampling on a superset of solutions. While stochastically the algorithm may find one of the solutions, the search is not guided in any way. Due to the intrinsic stochasticity of the quantum search algorithm, it may be beneficial to add an additional layer that acts as a classical mutation layer.
As explained in the previous section, applying the adaptive diffusion recombination step of the EQDR algorithm, causes the alteration of the qubits' amplitudes, deterministically setting some of the qubits' states in the register equal to the corresponding classical basis states represented by the diffusion vector elements $\hat{b}_i$. However, in heuristic optimization problems, determinism is not always a good feature. The complete lack of stochasticity may result in the same state being measured at each iteration. This is due to the fact that, as proven in detail in Appendix 6.2, applying a single Grover iteration to a two-dimensional system enables to deterministically push a qubit's state toward a wanted result. Upon measuring the quantum register, the solution is evaluated and added to the genetic knowledge. In case of complete determinism in the EQDR procedure, the measurements will converge into a situation in which the computed diffusion vector is the same as the measured one, effectively preventing the algorithm to move away from this bad equilibrium.
For this reason, adding a mutation layer after the adaptive diffusion recombination can reintroduce a certain level of non-determinism, required by the heuristic nature of the algorithm. Usually, this can be achieved by applying rotational gates to the qubit. In this situation, the qubit's state can be either known, as the adaptive diffusion recombination has pushed it toward a wanted state, or unknown, for those qubits which have not been altered by the diffusion procedure. In case the qubit's state is known, the sign of the angle that drives the rotational gates is not a problem. In fact, the probabilities of the state obtained by rotating one of the two basis states are the same in case the angle is positive or negative. Instead, if the qubit's state is unknown, the rotational gate acts effectively as a minimal perturbation toward a random direction, as the rotation may move the qubit in any direction, depending on its relative phase. However, in the general scheme of the algorithm this is not a problem. The goal here is exactly to perturbate the qubit in a random direction, instead of pushing the qubit toward a predefined basis state.
Let us consider the following i-th two-dimensional subsystem composed of a register qubit that has been altered by the adaptive diffusion recombination and its ancilla. Since the value $\hat{b}_i$ has been diffused into the qubit, the initial state of the system is:

$$|\psi_0\rangle = |\hat{b}_i\rangle \otimes |a\rangle$$

where |a⟩ is the unknown state of the ancilla, not important for the computation. At this point, applying a rotational gate $R_y(\theta)$ to the register qubit $|\hat{b}_i\rangle$ enables to slightly move it away from the deterministic state, perturbating it with an amplitude that depends on the modulus of the angle θ.
Notice how, in this case, the sign of the angle θ is not important. The probabilities associated with the two basis states $|0\rangle$ and $|1\rangle$ do not depend on the sign of the angle. From this point on, the ancilla qubit can be ignored as it is not important for the computation. Expressing the register qubit's state as $|\hat{b}_i\rangle = \alpha|0\rangle + \beta|1\rangle$ and using the matrix form for rotational gates defined in Equation (2.1), the application of the rotational gates proceeds as:

$$R_y(\theta)|\psi_0\rangle = \begin{bmatrix} cos\left(\frac{\theta}{2}\right) & -sin\left(\frac{\theta}{2}\right) \\ sin\left(\frac{\theta}{2}\right) & cos\left(\frac{\theta}{2}\right) \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha\, cos\left(\frac{\theta}{2}\right) - \beta\, sin\left(\frac{\theta}{2}\right) \\ \alpha\, sin\left(\frac{\theta}{2}\right) + \beta\, cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

$$R_y(\theta)|\psi_0\rangle = \left(\alpha\, cos\left(\frac{\theta}{2}\right) - \beta\, sin\left(\frac{\theta}{2}\right)\right)|0\rangle + \left(\alpha\, sin\left(\frac{\theta}{2}\right) + \beta\, cos\left(\frac{\theta}{2}\right)\right)|1\rangle. \quad 2.3$$

Although the state $|\hat{b}_i\rangle$ is expressed in terms of linear combination of the basis states, it is either $\alpha = 0, \beta = 1$ or $\alpha = 1, \beta = 0$, as the state has been pushed toward one of the two basis states by the adaptive diffusion recombination step. In simple words, this means that either $|\hat{b}_i\rangle = |0\rangle$ or $|\hat{b}_i\rangle = |1\rangle$. Let us consider the two different cases:

- If $\alpha = 0, \beta = 1, |\hat{b}_i\rangle = |1\rangle$:

$$R_y(\theta)|\psi_0\rangle = sin\left(\frac{\theta}{2}\right)|0\rangle + cos\left(\frac{\theta}{2}\right)|1\rangle$$

$$P(0) = \left|sin\left(\frac{\theta}{2}\right)\right|^2, \qquad P(1) = \left|cos\left(\frac{\theta}{2}\right)\right|^2;$$

- If $\alpha = 1, \beta = 0, |\hat{b}_i\rangle = |0\rangle$:

$$R_y(\theta)|\psi_0\rangle = cos\left(\frac{\theta}{2}\right)|0\rangle - sin\left(\frac{\theta}{2}\right)|1\rangle$$

$$P(0) = \left|cos\left(\frac{\theta}{2}\right)\right|^2, \qquad P(1) = \left|-sin\left(\frac{\theta}{2}\right)\right|^2.$$

The only constraint on the angle θ is that $|\theta| \in \left[0, \frac{\pi}{2}\right]$, as applying rotations that are greater than $\frac{\pi}{2}$ in modulus would cause the total loss of the genetic knowledge encoded into the state, transforming the procedure in a random sampling. To understand the effect of the application of rotational gates to the basis states, let us consider the two basis states $|0\rangle$ and $|1\rangle$ on the Bloch Sphere as identified by the angles $\frac{\pi}{2}$ and $-\frac{\pi}{2}$ from the horizontal y axis respectively. Figure 13 shows the geometrical representation of the two basis states.

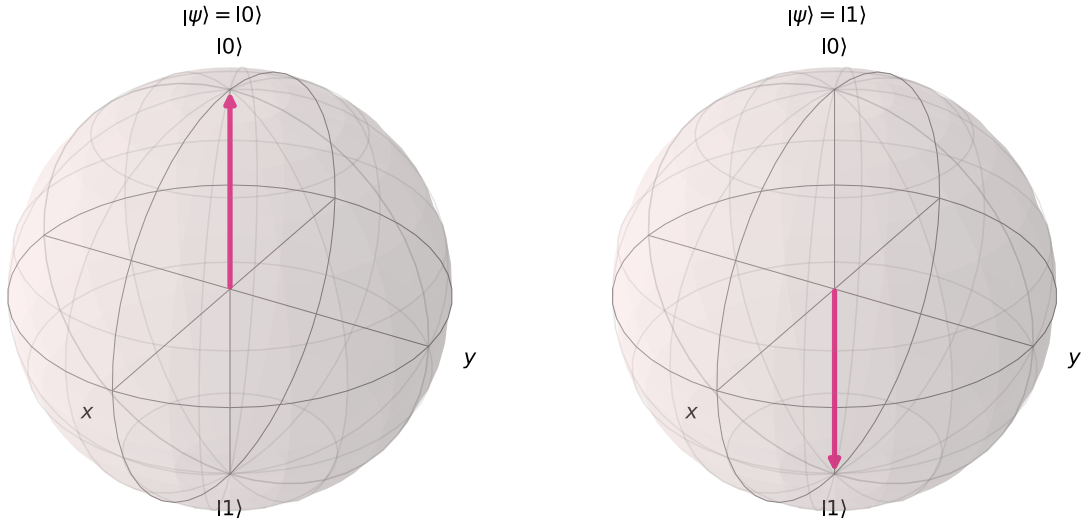

Figure 13: the two basis states $|0\rangle$ and $|1\rangle$ can be obtained with a rotation of respectively $\frac{\pi}{2}$ and $-\frac{\pi}{2}$ from the positive horizontal y axis. The following representation is given to better understand the action of the variational rotational gates on the two basis states.

If a rotation on the y axis with an angle θ is applied to one of the two basis states, the following relations hold:

$$sin\left(-\frac{\theta}{2}\right) = -\,sin\left(\frac{\theta}{2}\right), \qquad cos\left(-\frac{\theta}{2}\right) = cos\left(\frac{\theta}{2}\right)$$

$$\left|sin\left(-\frac{\theta}{2}\right)\right|^2 = \left|-\,sin\left(\frac{\theta}{2}\right)\right|^2, \qquad \left|cos\left(-\frac{\theta}{2}\right)\right|^2 = \left|cos\left(\frac{\theta}{2}\right)\right|^2. \tag{2.4}$$

Hence, the probabilities presented above do not depend on the sign of the angle θ:

- If $\alpha = 0, \beta = 1$:

$$P(0) = \left|sin\left(\frac{\theta}{2}\right)\right|^2 = \left|sin\left(-\frac{\theta}{2}\right)\right|^2, \qquad P(1) = \left|cos\left(\frac{\theta}{2}\right)\right|^2 = \left|-\,cos\left(\frac{\theta}{2}\right)\right|^2$$

- If $\alpha = 1, \beta = 0$:

$$P(0) = \left|cos\left(\frac{\theta}{2}\right)\right|^2 = \left|-\,cos\left(\frac{\theta}{2}\right)\right|^2, \qquad P(1) = \left|sin\left(\frac{\theta}{2}\right)\right|^2 = \left|sin\left(-\frac{\theta}{2}\right)\right|^2.$$

This property entails that it is possible to apply a rotational gate to each qubit with any angle θ in order to maintain a certain degree of stochasticity in the superposition and therefore in the measurement process. The modulus of the angle θ plays the same role of the mutation rate in classical genetic algorithms. The greater the modulus, the greater the level of stochasticity introduced, the smaller the modulus, the closer to the diffusion vector $\hat{b}$ the measurement outcome will be. Notice how, in case the qubit has been altered by the adaptive diffusion recombination, its state corresponds to one of the two basis states $|0\rangle$ and $|1\rangle$. For this reason, the probabilities of the different outcomes are exactly the same as the ones obtained by applying the rotational gate directly to the basis states.

In the situation in which the qubit has not been altered by the adaptive diffusion recombination, its state is unknown and the derivation of the measurement outcomes probabilities changes. In this case, the initial state of the system is:

$$|\psi_0\rangle = (\alpha|0\rangle + \beta|1\rangle)\otimes|a\rangle, \qquad |\alpha|^2 + |\beta|^2 = 1, \qquad \alpha, \beta \in \mathbb{C}.$$

As before, the ancilla qubit is not important as it does not influence the calculation, hence it can be safely ignored from this point on. The effect of the application of the rotational gate with an angle θ is shown in Equation (2.3) and is reported for convenience:

$$R_y(\theta)|\psi_0\rangle = \left(\alpha\, cos\left(\frac{\theta}{2}\right) - sin\left(\frac{\theta}{2}\right)\beta\right)|0\rangle + \left(\alpha\, sin\left(\frac{\theta}{2}\right) + cos\left(\frac{\theta}{2}\right)\beta\right)|1\rangle = \begin{bmatrix} \alpha\, cos\left(\frac{\theta}{2}\right) - sin\left(\frac{\theta}{2}\right)\beta \\ \alpha\, sin\left(\frac{\theta}{2}\right) + cos\left(\frac{\theta}{2}\right)\beta \end{bmatrix}.$$

Clearly, the probabilities depend on both the coefficients α and β of the initial state. Applying a rotational gate to an unknown state can have unpredictable results, depending on the sign of the angle θ and on the relative phase of the system's state. While this situation is a problem in case of the adaptive diffusion recombination, in this case, the goal is to exactly introduce a certain level of stochasticity into the system. Thus, the impossibility to predict the actual effect of the rotational gate on the unknown state is not a limitation.
Analogously to classical evolutionary algorithms, the application of the superposition mutation presented in this section depends on the mutation rate η and mutation amplitude m_a. The mutation rate defines the probability of applying a mutation to each qubit in the quantum register, while the mutation amplitude represents the minimum and maximum values of a random mutation amplitude selected during the operation. In other words, each qubit has a probability $\eta \in [0,1]$ of being mutated with a rotational gate $R_y(\theta)$ in which the value of the angle θ is picked from a uniform distribution U:

$$\theta \sim U(-m_a, m_a).$$

Notice that the angle is picked from a distribution symmetric around the zero value. This means that the probability of picking a positive value is the same as the probability of picking a negative one. In case the state of the qubit is known, rotating the qubit in a positive or negative direction has the same effect in terms of probabilities of outcomes, as shown in Equation (2.4). Whenever the state is unknown, rotating in both directions may result beneficial as the state may have a positive or negative relative phase. Rotating in a single direction may cause the superposition mutation to be unbalanced whenever the distribution of relative phases of the states of the different qubits is biased towards one sign.
The state evolution of a single two-dimensional subsystem, formed by one qubit of the register and its ancilla, during the EQDR recombination procedure is presented in Appendix 6.2. The computation includes both steps of the recombination procedure the adaptive diffusion recombination and the superposition mutation.

2.5 Diffusion and accuracy vectors computation

The core concept of the EQDR procedure is to adapt the repeated applications of Grover's algorithm by exploiting the genetic knowledge acquired during different iterations to guide the search.
The genetic knowledge can be defined as a set $B = \{(g_i, f_i)\}_{i\in[0,N_b]}$ where the pairs (g_i, f_i) are composed of the genome and its corresponding fitness value. The set is updated each iteration. Every time the superposition is measured, the fitness of the measured genome is computed classically and the pair is inserted into the set B, replacing the least fit element if the set has reached a predefined maximum size.
In the evolutionary computation theory, this set can be associated to an elite pool, in which the best individual of each generation is inserted. The quantum oracle used in Grover's algorithm is not required to perfectly represent the target fitness function as the actual fitness must be computed classically on the single measured genome. This entails that the fitness function is evaluated only once per generation.

As introduced in the previous chapter, the EQDR procedure requires the diffusion vector $\hat{b}$ in order to execute the subroutine that enforces the state represented by $\hat{b}$ by acting on the superposition. The accuracy vector $\hat{a}$ is also needed to express the probabilities of enforcing the corresponding diffusion bit into the i-th qubit. The calculation of the vectors $\hat{b}$ and $\hat{a}$ is transparent to the EQDR procedure: any black box operation that takes as input the genetic knowledge acquired and outputs the diffusion bits vector and the accuracy vector can be used. In this chapter, different ways of extracting the diffusion bits and calculating the corresponding probabilities are presented. It is worth noticing the fact that, acting now in a classical environment, it is possible to use any kind of algorithm and knowledge borrowed from the evolutionary computation algorithms in order to complete the task.
With $B = \{(g_i, f_i)\}_{i\in[0,N_b]}$ being the set composed of pairs of chromosomes in a bit string format and their corresponding fitnesses, the goal is to calculate the diffusion vector $\hat{b}$ and the accuracy vector $\hat{a}$, both of length $n = log_2 N$, to be passed to the EQDR procedure to perform the recombination step.

2.5.1 Ranked contribution diffusion (RCD)

In this method, each genome in the set contributes proportionally to its rank which is based on the fitness. To begin with, the set B is ordered having the fittest individual at position 0. The contributions are calculated as:

$$c_i = \sum_{k=0}^{|B|-1} (2g_i - 1) \cdot \gamma(k) \qquad 2.5$$

where k is the rank of the individual and $\gamma: \mathbb{N} \to [0,1]$ is a scaling function that determines the contribution factor of each individual based on its fitness rank. Having calculated the contributions, the diffusion bits are then computed as:

$$\hat{b}_i = \begin{cases} 0, & c_i < 0 \\ 1, & c_i \geq 0 \end{cases}.$$

The function γ can have any shape, provided that the output is in the interval $[0,1]$. Two different scaling functions are presented below. To compute the accuracy vector $\hat{a}$, it is possible to exploit the values c_i computed according to Equation (2.5). The element c_i intrinsically represents how strongly the i-th element of the diffusion vector $\hat{b}_i$ must be enforced into the qubit's state. In case $c_i \ll 0$, the genetic knowledge is enforcing the bit 0 with a good accuracy, as multiple elements in the pool have the bit 0 in the i-th position, effectively contributing to the negative value of c_i. On the other hand, having the value $c_i \gg 0$ means the opposite: the elements in the genetic pool are enforcing the bit 1 into the i-th position. The values c_i are therefore perfect candidates to represent the probabilities of diffusion. The requirement that needs to be satisfied is that $c_i \in [0,1], \forall i$. This means that the values must be remapped into the correct domain. How to achieve this condition depends on the scaling function γ used and is explained for both scaling functions in the next subsections. Notice that the sign of the elements can be removed without any loss of generality. The i-th element of the vector $\hat{a}$ is associated with the i-th element of the vector $\hat{b}$, therefore which value must be diffused into the qubit's state is identified by the element $\hat{b}_i$, while the element $\hat{a}_i$ simply identifies the probability. In other words, the sign of c_i identifies the classical bit value that must be diffused into the i-th qubit.

Polynomial scaling function

The first scaling function proposal is a polynomial scaling function defined as:

$$\gamma_p(k) = \alpha^k, \qquad \alpha \in [0,1]. \qquad 2.6$$

In this situation, the constant α can be adjusted to modify the weight of the contributions of individuals based on their rank. Setting α close to 1 will result in similar contributions for all individuals, effectively reducing

the importance of the rank. On the other hand, setting the value close to 0 will make contributions highly asymmetrical, biasing them towards higher ranked individuals.
The fittest individual will have the maximum scaling value. The least fit individual, that is the one ranked last, will have a scaling value equal to $\alpha^{|B|-1}$, where $|B|$ is the number of individuals in the genetic pool:

$$\gamma_p(0) = 1, \qquad \gamma_p(|B| - 1) = \alpha^{|B|-1}.$$

To compute the elements of the accuracy vector $\hat{a}$, it is important to note that, due to Equations (2.5) and (2.6), the maximum value ζ that each of the contribution elements c_i can obtain is:

$$\zeta = \sum_{i=0}^{|B|-1} \alpha^i.$$

To normalize the elements c_i, it is possible to divide the modulus of each value by the maximum value achievable ζ. The accuracy vector is therefore computed as:

$$\hat{a}_i = \frac{|c_i|}{\zeta}.$$

Let us consider the following situation in which the fitness is supposed to be minimized and the polynomial scaling γ_p is used:

$$B = \{(000, 0), (001,1), (111, 3), (011, 5)\}$$

$$|\mathrm{B}| = 4, \qquad n = 3 \Rightarrow N = 8, \quad \alpha = \frac{2}{3}$$

$$\gamma(k) = \gamma_p(k) = \left(\frac{2}{3}\right)^k$$

$$c_0 = -1 \cdot \alpha^0 - 1 \cdot \alpha^1 + 1 \cdot \alpha^2 - 1 \cdot \alpha^3 = -1 - \frac{2}{3} + \frac{4}{9} - \frac{8}{27} = -\frac{41}{27}$$

$$c_1 = -1 \cdot \alpha^0 - 1 \cdot \alpha^1 + 1 \cdot \alpha^2 + 1 \cdot \alpha^3 = -1 - \frac{2}{3} + \frac{4}{9} + \frac{8}{27} = -\frac{25}{27}$$

$$c_2 = -1 \cdot \alpha^0 + 1 \cdot \alpha^1 + 1 \cdot \alpha^2 + 1 \cdot \alpha^3 = -1 + \frac{2}{3} + \frac{4}{9} + \frac{8}{27} = \frac{11}{27}.$$

With these contributions, the diffusion bits are:

$$c_0 < 0 \Rightarrow \hat{b}_0 = 0, \qquad c_1 < 0 \Rightarrow \hat{b}_1 = 0, \qquad c_2 > 0 \Rightarrow \hat{b}_2 = 1$$

$$\hat{b} = (001).$$

The computation of the accuracy vector $\hat{a}$ is:

$$\zeta = \sum_{i=0}^{|B|-1} \alpha^i = \alpha^0 + \alpha^1 + \alpha^2 + \alpha^3 = 1 + \frac{2}{3} + \frac{4}{9} + \frac{8}{27} = \frac{65}{27}$$

$$\hat{a}_0 = \frac{|c_0|}{\zeta} = \left|-\frac{41}{27}\right| \cdot \frac{27}{65} = \frac{41}{65}$$

$$\hat{a}_1 = \frac{|c_1|}{\zeta} = \left|-\frac{25}{27}\right| \cdot \frac{27}{65} = \frac{25}{65}$$

$$\hat{a}_2 = \frac{|c_2|}{\zeta} = \frac{11}{27} \cdot \frac{27}{65} = \frac{11}{65}$$

$$\hat{a} = \left(\frac{41}{65}, \frac{25}{65}, \frac{11}{65}\right) \cong (0.63, 0.38, 0.17).$$

The accuracy vector defines that the first bit 0 will be enforced with a 63% chance, the second bit 0 will have a 38% chance of being diffused into the corresponding bit and finally the last bit 1 will be enforced into the last qubit with a 17% chance.

Gaussian scaling function

The second proposal is a Gaussian-shaped scaling function:

$$\gamma_g(k) = e^{\frac{k^2 \, ln(\alpha)}{(|B|-1)^2}}, \qquad \alpha \in [0,1]. \tag{2.7}$$

In this situation, the parameter α determines the minimum contribution value for the individual ranked last:

$$\gamma_g(0) = 1, \qquad \gamma_g(|B| - 1) = \alpha.$$

In the polynomial scaling function, the scaling value associated with the least fit individual is $\alpha^{|B|-1}$. In order to weight the contribution of the individual ranked last in the same way in both situations for comparison purposes, defining α_p as the parameter related to the polynomial function and as α_g the parameter relative to the gaussian function, the following relation must hold:

$$\alpha_g = (\alpha_p)^{|B|-1}. \tag{2.8}$$

As shown in Figure 14, the Gaussian function γ_g allows distributing the contributions more equally. This results in better "exploration" among the generational knowledge, since a higher contribution of genetic material is computed for each individual. On the other hand, the polynomial function γ_p tends to enforce exploitation of the highest ranked solutions in the genetic pool. Although the parameter α can be adjusted to obtain different steepness for both functions, their shape is different, and thus, impacts the overall behavior of the algorithm in a different way.

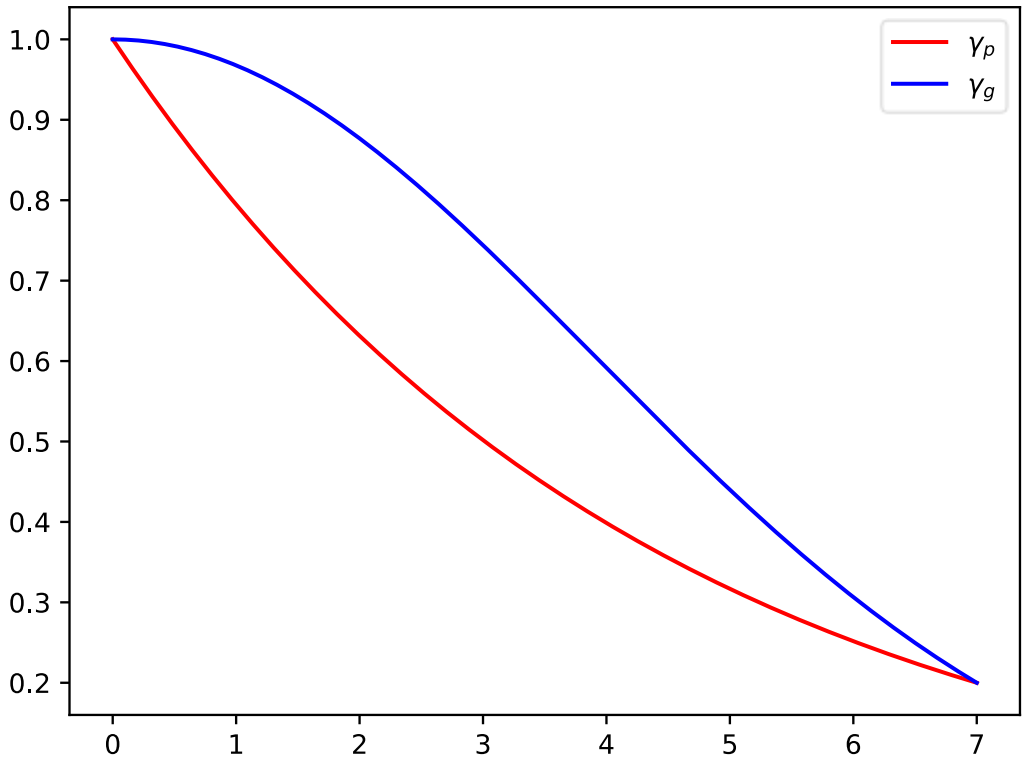

Figure 14: different shapes of the polynomial and gaussian γ functions when $|B| = 8$, $\alpha_g = \frac{1}{5}$, $\alpha_p = \left(\frac{1}{5}\right)^{1/3}$. The relation presented in Equation (2.8) allows to weight in the same way the least fit individual, that is the one ranked last, effectively enabling a comparation between the shapes of the two different functions.

In the case of the Gaussian scaling function, the maximum value ζ that an element c_i can obtain is equal to:

$$\zeta = \sum_{i=0}^{|B|-1} e^{\frac{i^2 \, ln(\alpha)}{(|B|-1)^2}}.$$

The computation of the accuracy vector elements proceeds in the same way as for the polynomial function:

$$\hat{a}_i = \frac{|c_i|}{\zeta}.$$

To better understand the computation of the diffusion bits and the accuracy vector, and the differences with the polynomial gamma, let us consider the same situation as before, but a parameter α set to a value consistent with Equation (2.8):

$$B = \{(000, 0), (001,1), (111, 3), (011, 5)\}$$

$$N_b = 4, \qquad n = 3 \Rightarrow N = 8, \qquad \alpha = \left(\frac{2}{3}\right)^3 = \frac{8}{27}$$

$$\gamma(k) = \gamma_g(k) = e^{\left(\frac{k^2}{9} \, ln(\alpha)\right)}.$$

Noticing that:

$$e^{\left(\frac{k^2}{9}\ln(\alpha)\right)} = e^{ln\left[\alpha^{k^2/9}\right]} = \alpha^{k^2/9},$$

the computation proceeds as:

$$c_0 = -1 \cdot \alpha^0 - 1 \cdot \alpha^{\frac{1}{9}} + 1 \cdot \alpha^{\frac{4}{9}} - 1 \cdot \alpha^{\frac{9}{9}} \cong -1.587$$

$$c_1 = -1 \cdot \alpha^0 - 1 \cdot \alpha^{\frac{1}{9}} + 1 \cdot \alpha^{\frac{4}{9}} + 1 \cdot \alpha^{\frac{9}{9}} \cong -0.995$$

$$c_2 = -1 \cdot \alpha^0 + 1 \cdot \alpha^{\frac{1}{9}} + 1 \cdot \alpha^{\frac{4}{9}} + 1 \cdot \alpha^{\frac{9}{9}} \cong 0.752.$$

Finally, the diffusion bits are:

$$c_0 < 0 \Rightarrow \hat{b}_0 = 0, \qquad c_1 < 0 \Rightarrow \hat{b}_1 = 0, \qquad c_2 > 0 \Rightarrow \hat{b}_2 = 1$$

$$\hat{b} = (001)$$

and the computation of the accuracy vector $\hat{a}$ is:

$$\zeta = \sum_{i=0}^{|B|-1} \alpha^{i^2/9} = \alpha^0 + \alpha^{1/9} + \alpha^{4/9} + \alpha^{9/9}$$

$$\hat{a}_0 = \frac{|c_0|}{\zeta} = \frac{1.587}{\zeta} \cong 0.577$$

$$\hat{a}_1 = \frac{|c_1|}{\zeta} = \frac{0.995}{\zeta} \cong 0.362$$

$$\hat{a}_2 = \frac{|c_2|}{\zeta} = \frac{0.752}{\zeta} \cong 0.273$$

$$\hat{a} = (0.58, 0.36, 0.27).$$

Respectively, the first bit will be diffused with a 58% chance, the second one with a 36% chance and the last bit will be enforced with a 27% chance. Notice how the relationship between the elements in the accuracy vector is different with respect to the vector computed in the case of a polynomial scaling function. Of course, in case of a gaussian scaling function, the parameter α plays a different role: it represents the minimum value that is associated with the least fit individual. In case of the polynomial scaling function, the gamma associated with the least fit individuals is equal to $\alpha^{|\mathrm{B}|-1}$, where $|\mathrm{B}|$ is the dimension of the genetic pool. Consequently, the parameter must be adjusted and in case of the gaussian function, it is normally smaller. The numerical examples have been introduced in order to explicitly show the calculation of the diffusion and the accuracy vectors, along with the differences that arise using a Gaussian gamma rather than a polynomial one.

2.5.2 Stochastic parent diffusion (SPD)

The stochastic parent diffusion selects a single parent from the genetic pool and enforces the classical state represented by the chromosome into the quantum register. The diffusion bits are set to the genome of the parent selected. Parents are sampled from the set B by means of a certain distribution Ω: $[0, |\mathrm{B}|] \rightarrow [0,1]$:

$$\hat{b} = g_k, \qquad k \sim \Omega.$$

In case the distribution Ω is a uniform distribution, the operator effectively acts as a classical single parent uniform crossover operator where the parents are selected from the set B.
It is also possible to introduce the dependency on the fitness rank. In this case the distribution Ω can be shifted towards fitter individuals. The difference with the ranked contribution diffusion method is the fact that bits are not calculated by mixing the values of the different genomes, but instead just drawn from a distribution.
Note that it is possible to introduce any kind of dependency in the distribution Ω. As an example, varying the distribution based on the index of the qubits that is being recombined can be useful in cases in which solutions are encoded in a particular way such that, for certain bits, exploration is preferred, while for others, exploitation of the existing solutions is more important.
The distribution used in the experiments is proportional to the fitnesses. Each individual is associated with a probability p_i of being picked that depends on the fitness of the individual. If the fitness function is supposed to be minimized, individuals with lower fitness must be associated with a higher probability of being selected. To remap the fitness values in the genetic pool B, each individual's fitness f_i can be associated with a value f_i' computed as:

$$m = \max_i \{f_i\}$$

$$f_i' = (\epsilon + 1)m - f_i.$$

If instead the fitness needs to be maximized, the remapping can be computed as:

$$f_i' = (\epsilon + 1)m + f_i.$$

The parameter ϵ is responsible for determining the minimum possible probability of being picked, preventing probabilities equal to zero. The probabilities are therefore computed as:

$$p_i = \frac{f_i'}{\sum_i f_i'}.$$

To better understand the computation, let us consider the same situation as the one presented in the previous subsection with the fitness function being minimized:

$$B = \{(000, 0), (001,1), (111, 3), (011, 5)\}$$

$$|B| = 4, \qquad n = 3 \Rightarrow N = 8, \qquad \epsilon = 0.1.$$

The first possibility is to use a uniform distribution to pick the elements. In this case, each individual in B will have the same probability of being picked and the corresponding accuracy vector will be set a priori as an hyperparameter ρ.
In case the fitnesses are taken into account, each element has a probability of being picked proportional to its fitness. First, the fitnesses must be remapped:

$$m = \max_i \{f_i\} = f_3 = 5$$

$$f_i' = m(\epsilon + 1) - f_i,$$

$$f_0' = 5 \cdot 1.1 - f_0 = 5.5, \qquad f_1' = 5 \cdot 1.1 - f_1 = 4.5,$$

$$f_2' = 5 \cdot 1.1 - f_2 = 2.5, \qquad f_3' = 5 \cdot 1.1 - f_3 = 0.5$$

$$\sum_i f_i' = 13.$$

The probabilities are computed as:

$$p_i = \frac{f_i'}{\sum_i f_i'}$$

$$p_0 = \frac{5.5}{13}, \qquad p_1 = \frac{4.5}{13}, \qquad p_2 = \frac{5.5}{13}, \qquad p_3 = \frac{0.5}{13}.$$

Note that, due to the computation used, the probabilities sum to one. To compute the accuracy vector $\hat{a}$, a fixed diffusion recombination probability $\rho \in [0,1]$ can be used to set the accuracy vector as:

$$\hat{a}_i = \rho, \qquad \forall i.$$

In this case, it is possible to have control over the stochastic application of the adaptive diffusion recombination step. Having direct control over the parameter ρ allows adapting it a priori depending on the problem at hand. This entails that a certain knowledge regarding the problem that is being solved is required. It is of course possible to define different recombination probabilities for each qubit. This enables to drive the evolution of the different alleles in different ways by setting different hyperparameters a priori. In this situation, different alleles will contribute differently to the overall fitness of the individual. This requires the fitness function to be able to weight these different contributions of different alleles in the chromosome.

2.5.3 Uniform diffusion (UD)

To exploit the genetic knowledge acquired during the runs of the algorithm, it is possible to take inspiration from the classical crossover operators. The uniform diffusion methodology is effectively the analogous of a uniform crossover operator. In this scenario, each individual in the genetic pool has the same probability of donating its alleles to the diffusion vector. Each diffusion bit is taken at random from one of the individuals in the genetic pool. Individuals are sampled from the set B by means of a uniform distribution $U: [0, |B| - 1] \rightarrow [0,1]$:

$$\hat{b}_i = g_{ki}, \qquad k \sim U.$$

As in the stochastic parent diffusion, the accuracy vector $\hat{a}$ can be set to a fixed probability $\rho \in [0,1]$. Let us consider the same situation as in the examples above:

$$B = \{(000, 0), (001,1), (111, 3), (011, 5)\}$$

$$|B| = 4, \qquad n = 3 \Rightarrow N = 8.$$

The computation is straightforward. Each element of the diffusion vector $\hat{b}$ can be chosen from one of the four parents with an equal probability. The accuracy vector is set to a predefined value. As will be presented in Section 3, this method can be useful in case there is no prior knowledge regarding the fitness function that is being optimized. In this case, sine each genome contributes with a single allele, the uniform diffusion can enhance exploration instead of exploitation, enabling the optimization process to explore the search space faster. The UD methodology allows the generation of more heterogeneous solutions with respect to the SPD methodology.

A final consideration regarding the methodologies presented, is that, given a certain genetic set B, the UD and the SPD are able to generate diverse diffusion and accuracy vectors on different computations. This depends on the fact that both methodologies use parameters drawn from random distributions. This cannot happen in

case of the RCD methodology. The computation of the diffusion and accuracy vectors in the case of the RCD methodology solely depends on the current genetic set B. Although the RCD generates an offspring that is effectively a stochastic ensemble of the solutions, the UD and the SPD can enhance exploration at the cost of exploitation, due to the parameters drawn from the random distribution.

3 Experiments

In this chapter, experimental validation results for the EQDR algorithm are presented. The goal is to show in which situations the algorithm can bring a benefit in the optimization task. Moreover, the interactions between the various modules and hyperparameters are analyzed. It is important to keep in mind that, due to the quantum computing challenges of the NISQ era, the results are not analyzed in terms of real running time. Instead, various experiments are studied by taking into consideration the number of generations, hence fitness function calls, required to find the optimum. Results obtained under these circumstances can be automatically transferred to a hypothetical optimal quantum device that may in future exploit the benefits of the EQDR algorithm.
The first experiment is a validation of the adaptive diffusion recombination and its determinism explained in Section 2.3. Subsequently, a comparison between a vanilla BBHT using a partial oracle and the corresponding EQDR version of the algorithm is presented, with the goal of identifying in which situations and classes of problems the EQDR may have an advantage. Section 3.4 analyzes the classical part of the EQDR algorithm, in particular, the differences between the different methods for computing the diffusion vector $\hat{b}$ and the corresponding accuracy vector $\hat{a}$. Finally, the last experiment has the objective of identifying how the different hyperparameters of the EQDR algorithm can affect the computation, attempting to define heuristic rules for devising the best combination depending on the problem at hand.

3.1 Environment

All the experiments have been carried out with the Qiskit [9] framework, running on the Aer simulation backend. Qiskit Aer is a high-performance quantum computing simulator with realistic noise models. It provides interfaces to run quantum circuits with or without noise using multiple different simulation methods. Qiskit Aer supports running on GPUs to improve the performance of simulation [10]. The exclusive use of simulators to run the experiments is due to the type of experiments run and the circuit shapes.
The experiments have not been run on a real quantum device due to two main reasons. The first is the hybrid nature of the algorithm proposed. In fact, hybrid algorithms require an efficient alternation between quantum and classical subroutines. This entails the ability to efficiently execute quantum circuits with little to no overhead. In the current era, NISQ devices lack of this ability and require a lot of initialization work to be done. Secondly, at the time of writing, the average waiting time to execute a single circuit on a real IBM quantum device, using the Qiskit backend, is in the order of multiple hours. In the Aer simulator, it is possible to easily simulate with noise models up to 10 qubits systems with a running time of the order of minutes.
The EQDR algorithm structure, presented in Section 2.1, highlights how the algorithm is composed of repeated execution of different instances of Grover's algorithm as subroutines both for the initial quantum search, and for the adaptive diffusion recombination. Running such an algorithm on a real quantum device would mean scheduling and awaiting the completion of each subroutine call before running the next one. This is clearly unfeasible to execute on real quantum devices in the current era. Nonetheless, thanks to the modularity of the Qiskit framework, changing the backend to use a different simulator or a real quantum device is straightforward in terms of practical implementation in the code, allowing future work to also test the different backend providers.
It is important to remind that the different circuits that correspond to different iterations, cannot be concatenated in order to feed the whole block to the quantum device at once. This is again due to the hybrid nature of the EQDR algorithm. Exploiting variational quantum circuits with varying parameters, the definition of the quantum gates depends on the previous computation results. As an example, let us consider the adaptive diffusion recombination step. The definitions of the various oracles O_i depend on the diffusion vector $\hat{b}$ computed from the genetic knowledge gathered from previous iterations. Moreover, their application is not deterministic: the application of the i-th oracle to the corresponding qubit's subsystem depends on the value of the accuracy vector $\hat{a}_i$, also computed exploiting the genetic knowledge gathered. These reasons motivate the use of the Qiskit Aer backend to execute the experiments, simulating the circuits on a classical device.

Despite the theoretical exponential overhead required to simulate quantum devices on a classical device, thanks to Qiskit's optimizations, the Aer backend is able to manage 8 qubits systems on the computing device used, effectively providing solid and realistic simulation results. Moreover, as mentioned in the official documentation, the Aer backend can internally simulate and take into consideration the realistic noise models of a real quantum device. This entails that the Aer backend does not simulate a theoretical perfect quantum device, instead it simulates a noisy quantum computer, allowing to execute the circuits in the most realistic setting possible.

3.2 Diffusion recombination experimental validation

The experiment has the goal to validate the capability of the diffusion recombination to push any unknown qubit's state towards a certain basis state. Recall that, having a quantum register of n qubits, the goal is to enforce a certain classical state represented by a bitstring of length n into the quantum register. In order to do this, n single Grover iterations are applied to two-dimensional subsystems composed of the qubits and their ancillas. The ancilla qubits are used to make each subsystem two-dimensional. This is important as, in bidimensional systems, a single Grover iteration is enough to deterministically enforce a certain classical state into a qubit. Defining as $\hat{b}$ the diffusion vector, representing the classical state that must be enforced into the quantum register, the n oracles used into the various subsystems are defined according to Equation (2.2), reported here for convenience:

$$O_i = \mathbb{I} - 2\left(|\hat{b}_i\rangle\otimes|\hat{b}_i\rangle\langle\hat{b}_i|\otimes\langle\hat{b}_i|\right) = \begin{cases}\mathbb{I} - 2|00\rangle\langle 00|, & \hat{b}_i = 0\\ \mathbb{I} - 2|11\rangle\langle 11|, & \hat{b}_i = 1\end{cases}.$$

In the example below, the diffusion recombination is applied to all qubits, without taking into consideration the accuracy vector $\hat{a}$, to show the capability of introducing the determinism of measurement of the diffusion recombination step. Figure 15 depicts the diffusion circuit that is built to enforce the diffusion vector $\hat{b} = (0101)$ into a quantum register of 4 qubits. Qubits from 0 to 3 are the qubits of the quantum register, while the remaining ones are the ancillas.
Each ancilla qubit is associated with its register qubit using the indexing:

$$j = i + n,$$

where i is the index of the register qubit, n the dimension of the system and j the index of the ancilla qubit. In Figure 15 it is possible to notice that the Grover iterations are applied to qubits in pairs, following the indexing presented above. The total state of the quantum registers is initialized to a random state by applying a rotational gate with random θ to each qubit in the register. Numbers shown may change at different runs of the algorithm and have been included only for displaying purposes. The ancillas are instead initialized in the $|+\rangle = H|0\rangle$ state.

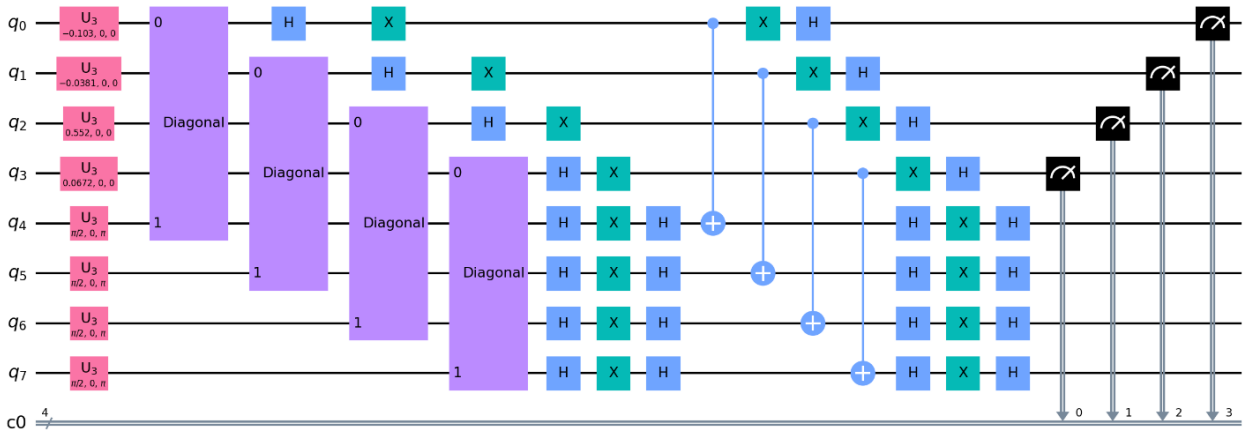

Figure 15: the circuit of the diffusion recombination procedure. The example shows the circuit that can be used to enforce any 4 qubits state into the register. The first 4 qubits are the target qubits, the last 4 qubits are the ancillas, used to form 4 two-dimensional subsystems. Differently to the vanilla Grover Algorithm, only the ancillas are initialized in the state $H|0\rangle = |+\rangle$.

The circuit is executed and measured 10000 times to prevent any statistical bias in the results. The histogram shown in Figure 16 proves that the diffusion recombination operator is able to deterministically enforce a certain classical state into a quantum register, independently of the initial quantum state.

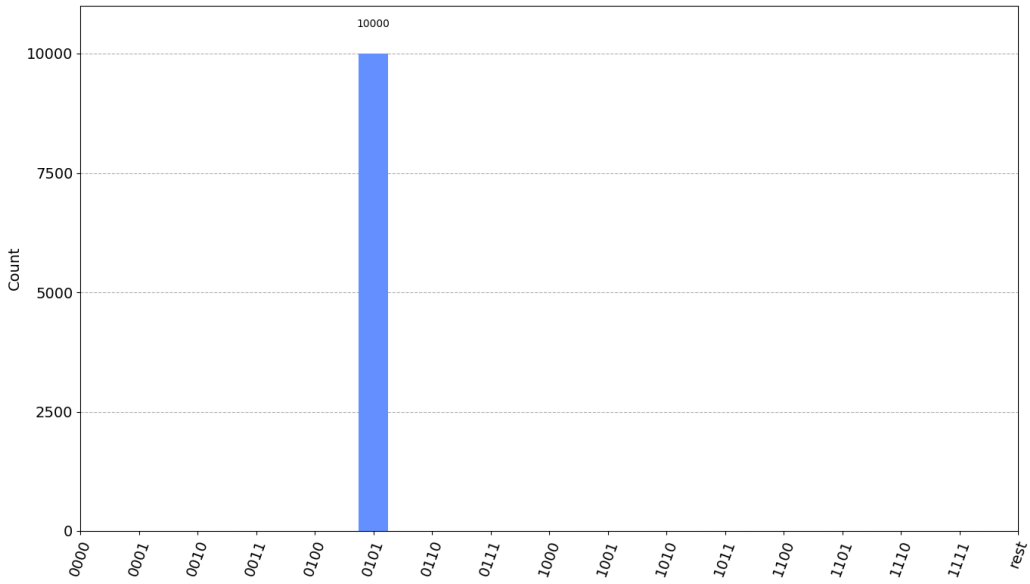

Figure 16: results of the diffusion recombination. The diffusion recombination procedure can deterministically enforce a certain state into the quantum register.

One may ask why there is the need of using a quantum operator to enforce a classical state into a quantum register when the register can be initialized in the target classical state instead. The answer lies in the non-determinism of the application of such an operator. In a situation in which the operator is applied deterministically to all the qubits in the quantum register, the output coincides with the classical state. Instead of applying a complicated quantum operator, a faster solution is to simply initialize the quantum register to the target state. However, if the diffusion operator is applied only to certain qubits, the output state will maintain a certain degree of stochasticity. In the EQDR framework, this allows to preserve some information of the quantum state that is outputted from the global Grover search. At the same time, the probabilistic application of the diffusion recombination allows to modify the state, biasing it towards the inter-generational knowledge acquired during different runs.

Note that it is impossible to initialize a new register to a quantum state by setting some qubits to a certain classical state while copying others. Due to the well-known no cloning theorem of quantum computation, it is not possible to clone arbitrary quantum states using a quantum operator. This entails that, apart from special situations, the qubits in an unknown state cannot be copied into a newly initialized quantum register. Using the same diffusion vector $\hat{b}$ presented above, let us consider the circuit shown in Figure 17 in which the diffusion recombination is applied only to qubits q_1 and q_3.

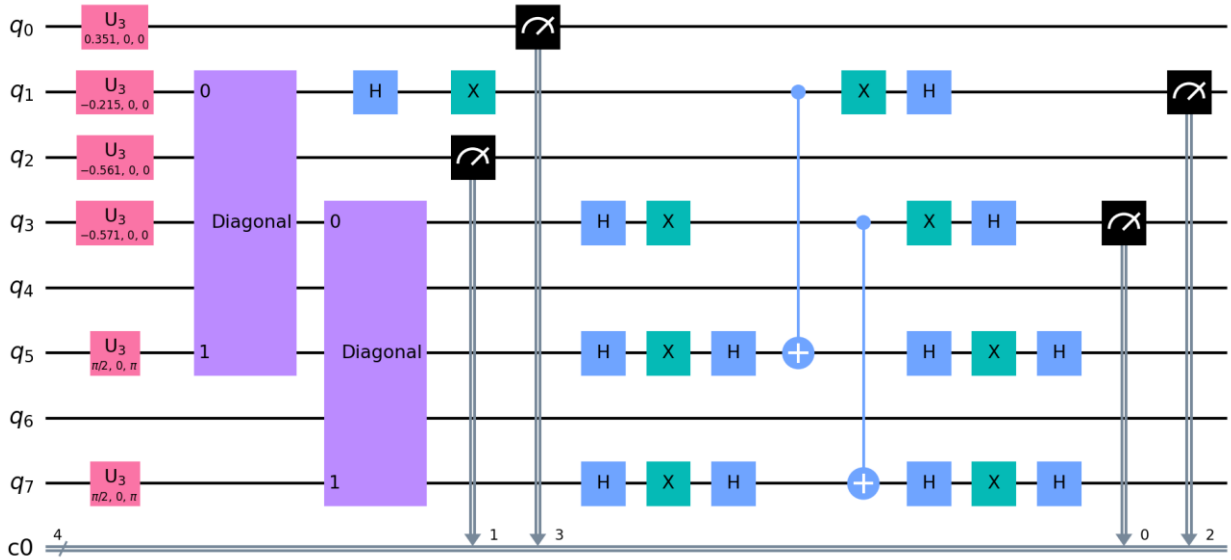

Figure 17: partial diffusion recombination operator applied to the qubits q_1 and q_3. The measurement operators are placed automatically by the Qiskit renderer and are displayed before for qubits q_0 and q_2, that do not undergo the diffusion step.

This partial diffusion recombination circuit leaves untouched qubits q_0 and q_2, randomly initializing their state and directly measuring the output. The interesting capabilities of the partial quantum diffusion recombination can be seen in Figure 18. The histogram shows that, although the target state is the one measured with the highest probability, there are other states that have a non-zero probability of being obtained as outcomes of the measurement process. In particular, the classical state 0111 has the second highest probability of being measured. Indeed, this state differs from the target state only for the bit in third position.

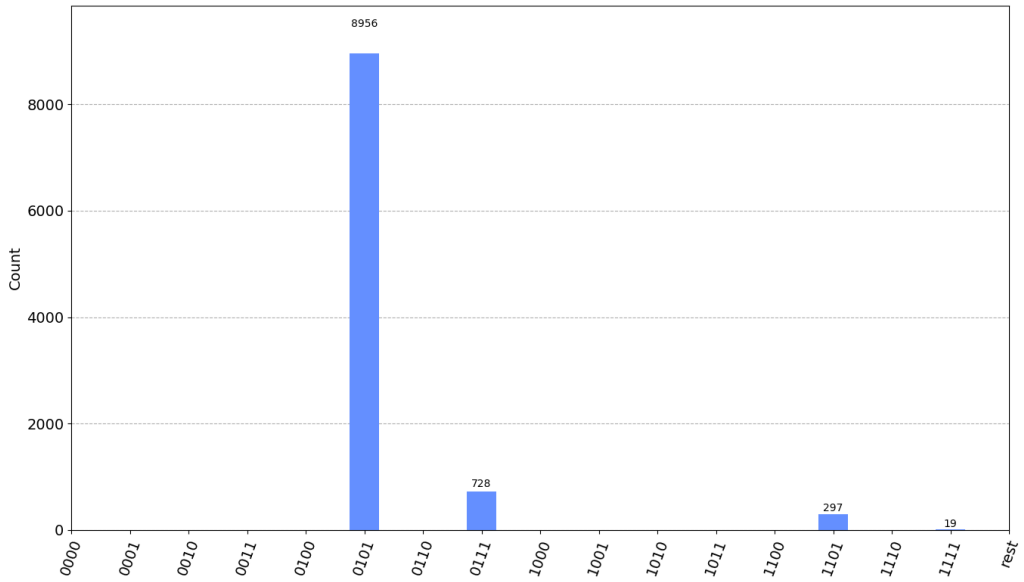

Figure 18: results of the partial diffusion recombination. Although the target classical state (0101) *is the one measured with the highest probability, there are other states with a non-zero probability of being obtained as a measurement outcome.*

The probabilities of the outcomes naturally depend on the initial state of the quantum register. To show how some of the prior knowledge regarding the state is conserved, let us consider the same circuit displayed in Figure 17 but with a different initialization. While ancillas are still initialized in the state $|+\rangle$, suppose now that the quantum register is initialized in the state in which all the qubits are rotated on the y axis by an angle $\theta = \frac{2}{3}\pi$. In this situation each qubit in the initial state $|\psi_0\rangle$ will have a high probability of returning the basis state $|1\rangle$ upon measurement:

$$|\psi_0\rangle = \bigotimes R_y\left(\frac{2}{3}\pi\right)|0\rangle.$$

Figure 19 shows that the probabilities change substantially with respect to the previous situation. The determinism of the diffusion recombination is lost when applying a partial operator and the histogram shows how the initial state information is maintained after the application of the operator, effectively enabling a stochastic recombination of the possible states.

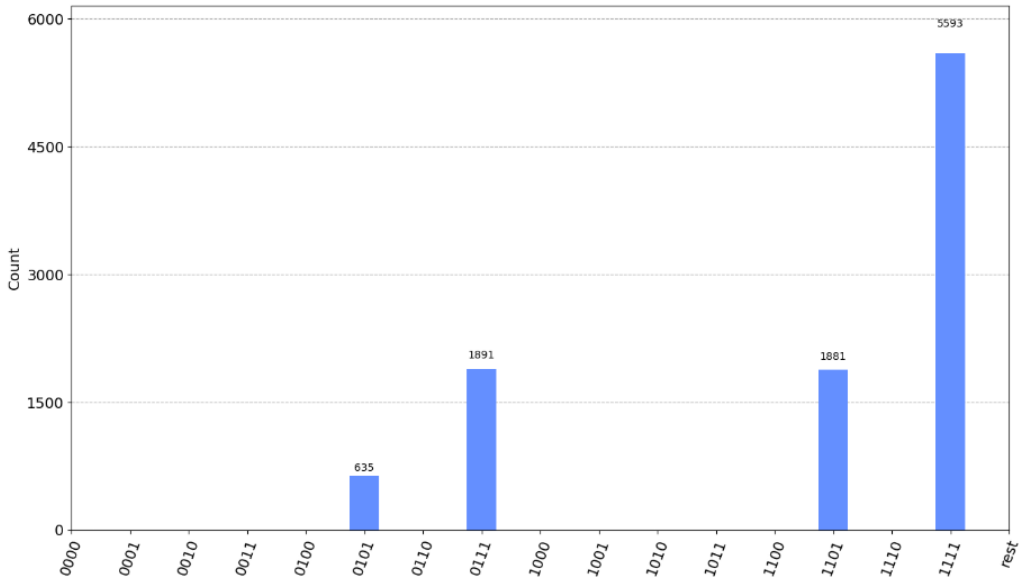

Figure 19: partial diffusion recombination with state initialization. The state is initialized in the state $|\psi_0\rangle$ in which each qubit has a greater probability of outputting the basis state $|1\rangle$ rather than the basis state $|0\rangle$ as the outcome of the measurement.

Finally, let us consider the case in which the diffusion recombination is applied to all qubits except for the first one. Hence, the circuit used is similar to the one presented in Figure 17 but with a Grover iteration applied also to qubit q_2. Moreover, all the states are initialized to $|\psi_0\rangle = R_y\left(\frac{2}{3}\pi\right)|0\rangle$. The classical state being enforced into the quantum register is the same as in the previous examples, that is 0101. In this situation, the results are exactly what one would expect. As shown in Figure 20, there are only two possible states that can be measured. These states are the ones that differ only for the value of the first qubit, the one left untouched by the diffusion recombination operator. The probabilities depend on the initial state. In the case presented, the state 1101 has a greater probability of being measures as the register is initialized in the state $|\psi_0\rangle$ presented above, in which each qubit is closer to the basis state $|1\rangle$.

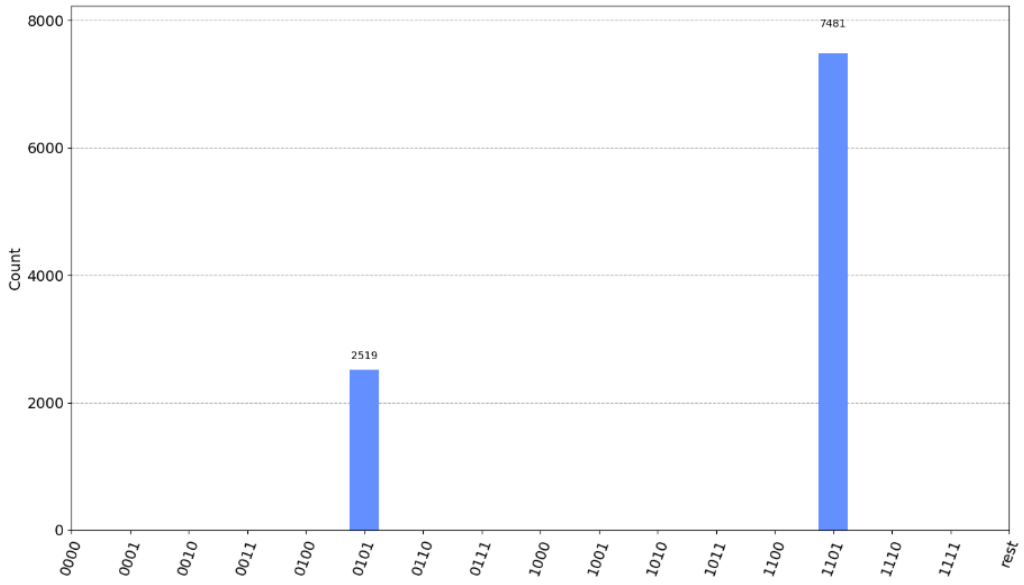

Figure 20: diffusion recombination experimental results in case of the first qubit being untouched by the diffusion operator.

3.3 Comparing partial BBHT with EQDR

Understanding the different situations in which the EQDR algorithm can have an advantage with respect to a pure BBHT is fundamental in order to identify the problems in which the proposed solution can bring a real benefit. The experiment presented compares the BBHT algorithm and the EQDR algorithm. Both the algorithms are executed with a partial oracle. The BBHT algorithm that uses a partial oracle can be effectively associated with a random sampling algorithm in which the sampled solutions are inside the superset recognized by the partial oracle. Finding a real solution means to run the random sampling a number of times high enough to obtain a solution that can be validated and accepted by the classical fitness function.
Differently, the EQDR algorithm attempts to guide the evolution towards the optimum, even when using a partial oracle. The algorithm gathers inter-generational knowledge by exploiting repeated measurements to store elite individuals. The diffusion recombination step is then adapted and modified exploiting the elite pool to compute its parameters. In a situation in which the partial oracle is close to an optimum one, the advantage of the EQDR algorithm should be limited with respect to a pure BBHT. This is due to the fact that, when a quantum oracle that is close to the optimal one is used, the BBHT samples solutions from a subspace that basically coincides with the space spanned by valid solutions, allowing to probabilistically obtain a valid solution in few calls. On the other hand, if the partial oracle used is very distant from the optimum oracle, meaning that it recognizes many invalid solutions, the EQDR should be able to effectively guide the search toward an optimum faster than the pure BBHT.

3.3.1 Rastrigin multimodal fitness function

The first fitness function taken into consideration is the well-known Rastrigin function. In its most general form, it is defined as:

$$f(x) = 10\kappa + \sum_{i=0}^{\kappa} x_i^2 - 10\,cos(2\pi x_i). \tag{3.1}$$

Figure 21 displays the shape of the Rastrigin fitness function in case of two target variables. In its general form, the function is highly multimodal, with multiple local optima and a single global minimum.

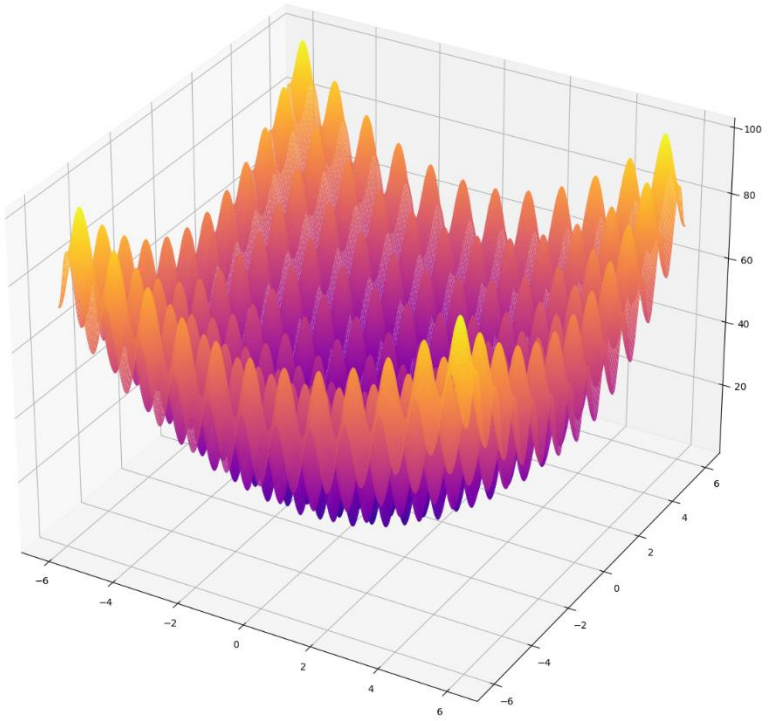

Figure 21: Rastrigin multimodal fitness function plot in 3D in case of two variables.

In the experiments, each genome can be considered as a single objective variable to optimize. This means that each solution is identified by the integer value represented by the bit string. In this situation, the function becomes:

$$f(x) = 10 + x^2 - 10\,cos(2\pi x).$$

With $\kappa = |g|$ being the size of the genome, the function can be moved and shaped in such a way such that it has the minimum in $x = \kappa$:

$$f(x) = 10 + (x - \kappa)^2 - 10\,cos\big(2\pi(x - \kappa)\big). \qquad 3.2$$

In this particular situation, since the phenotype of each genome is simply the integer representation, the function being optimized is effectively a quadratic parabola, as shown in Figure 22 below. Hence, the fitness function can effectively be considered a unimodal function. The reason for using the Rastrigin function is that, by simply changing the way the phenotype is constructed, it is possible to optimize a highly multimodal fitness function.

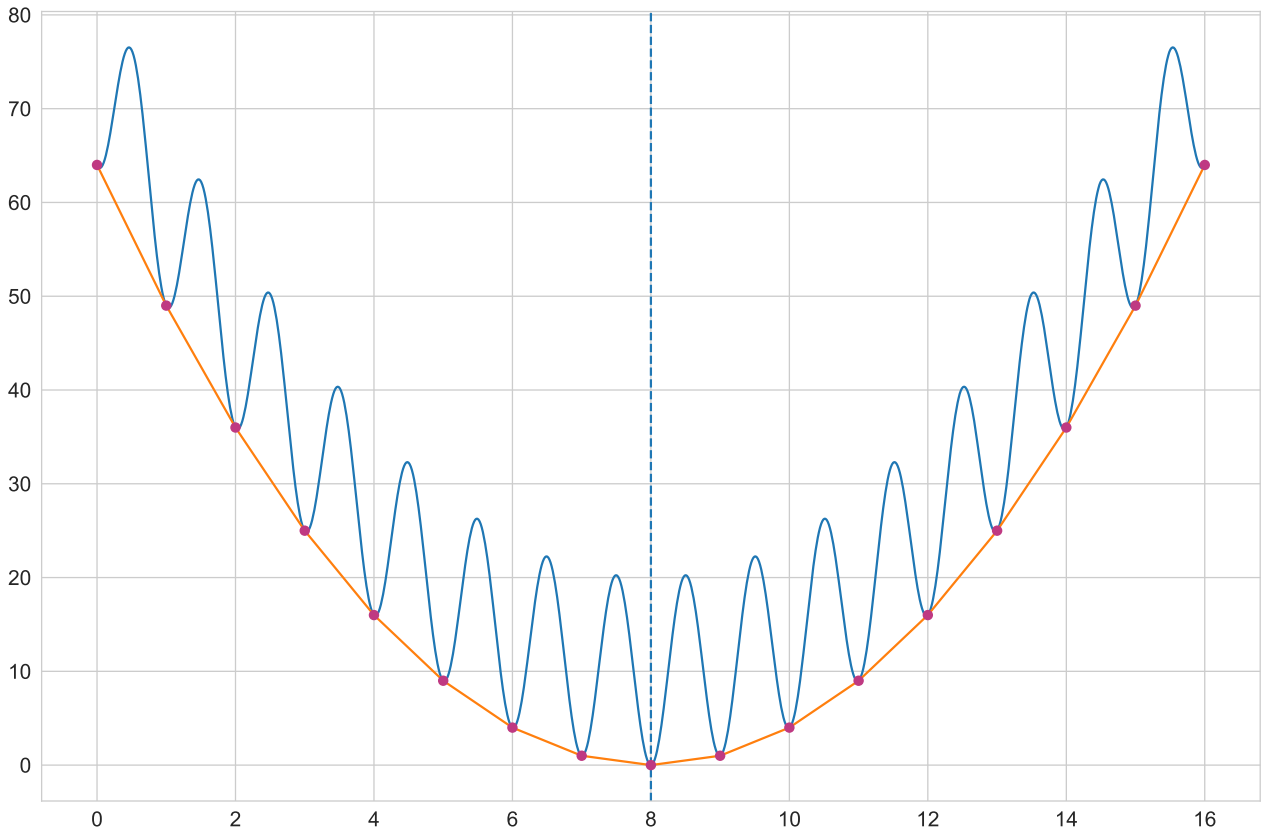

Figure 22: the discretized version of the Rastrigin fitness function. Constructing the phenotype of each individual as the integer represented by its bit string, it is possible to consider the Rastrigin fitness function as a discretized quadratic function..

System of 4 qubits

In this experiment, the system is composed of 4 qubits. A direct comparison between the BBHT and the EQDR is given. Using the fitness function defined in Equation (3.2), the optimum is $x = 4$, meaning the solution in binary form is $x_b = 0100$.
The oracle used in both algorithms is the partial oracle that is able to recognize all the solutions that end with two zeros. This means that the oracle marks a greater set of solutions as valid. The pattern of the partial oracle can be defined using Equation (1.8) from Section 1.2.2 and its precision with the Equation (1.9):

$$\tilde{s} = (** 00), \qquad \epsilon = \frac{1}{2^2} = \frac{1}{4}$$

where the symbol $* \in \{0,1\}$ indicates any binary value. In particular, the oracle recognizes as solutions the four states: $0000, 0100, 1000, 1100$. The only common parameter between the BBHT and the EQDR is the value of λ, that dictates the rate of growth of successive applications of the main Grover algorithm. Tables below present the parameters used in the simulation.

Common parameters

Grover iteration scaling λ	$\lambda = 6/5 = 1.2$

EQDR common parameters

Genetic pool size	$\lvert B \rvert = 8$
Recombination probability	$p_r = 0.75$
Mutation probability	$\eta = 0.25$
Mutation amplitude	$m_a = \pi/8$

EQDR(RCD)

Diffusion recombination	Ranked contribution diffusion (RCD)
Scaling function	Gaussian: γ_g
Scaling function α	$\alpha = 0.1$

EQDR(SPD)

Diffusion recombination	Stochastic parent diffusion (SPD)
Recombination accuracy	$\rho = 0.75$

EQDR(UD)

Diffusion recombination	Uniform diffusion (UD)
Recombination accuracy	$\rho = 0.75$

Each instance of the algorithm is run until the optimum is found and for a total of 500 times in order to reduce the statistical bias of the results. The boxplot in Figure 23 collects the different number of iterations required to find the optimum. It is possible to notice that there is basically no difference in terms of performance between the two algorithms when the dimensionality of the system is low. The dashed line identifies the mean value of the number of generations required to find the optimum. The same statistical datum is represented by the label annotation.
In case of 4 qubits, the pure BBHT algorithm works as well as the EQDR algorithm since there are relatively few candidate solutions. The EQDR algorithm requires some generations to gather inter-generational genetic knowledge regarding the search space. Before this knowledge is acquired, the algorithm effectively behaves as the pure BBHT in every aspect. Hence, by the time the EQDR algorithm has gathered useful genetic knowledge, the probability of having obtained the valid solution by chance is quite high. In other words, it can be stated that the EQDR has a response that is delayed with respect to the number of generations. The algorithm needs to gather enough inter-generational knowledge in order to start exploiting the methodologies for computing the diffusion and accuracy vectors. During this period of time, the EQDR algorithm relies solely on the application of the main Grover iteration with the partial oracle to sample possible solutions.

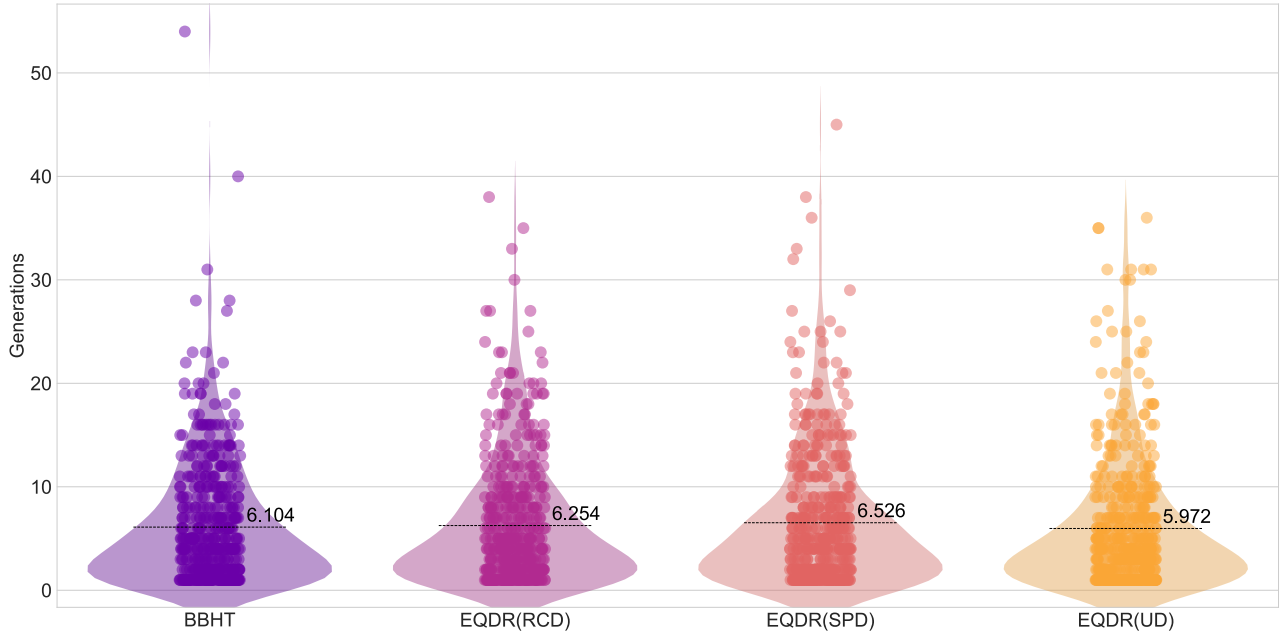

Figure 23: results in case of 4 qubits. There is no significant difference in terms of generations required to find the optimum.

System of 8 qubits

Scaling the system to a dimensionality of 8 qubits shows some promising results. In this situation, the optimum of the fitness function presented in Equation (3.1) is $x = 8$, hence the binary string that represents the solution is 0001000. The first experiment is run with an oracle that has a low precision ε. The pattern of the oracle used can be defined, using the convention presented above, as:

$$\tilde{s} = (****** 00), \qquad \varepsilon = \frac{1}{2^6} = \frac{1}{64}.$$

Common parameters

Grover iteration scaling λ	$\lambda = 6/5 = 1.2$

The tables below present the various parameters used for each instance of the EQDR algorithm with different diffusion and accuracy vectors computation methodologies.

EQDR common parameters

Genetic pool size	$\lvert B\rvert = 12$
Recombination probability	$p_r = 0.6$
Mutation probability	$\eta = 0.3$
Mutation amplitude	$m_a = \pi/5$

EQDR (RCD)

Diffusion recombination	Ranked contribution diffusion (RCD)
Scaling function	Gaussian: γ_g
Scaling function α	$\alpha = 0.2$

EQDR (SPD)

Diffusion recombination	Stochastic parent diffusion (SPD)
Recombination accuracy	$\rho = 0.75$

EQDR (UD)

Diffusion recombination	Uniform diffusion (UD)
Recombination accuracy	$\rho = 0.75$

Due to the higher system dimensionality, the effective running times of the algorithms are substantially increased. For this reason, each instance of the algorithm has been run for a total of 100 iterations instead of 500. As shown in Figure 24, the EQDR shows a clear advantage. In particular, the average number of generations required to find the global optimum is 61.83 for the EQDR algorithm using the RCD recombination against the 139.7 of the pure BBHT. The EQDR algorithm is able to obtain a speedup of more than two times in terms of the number of generations required to find the global optimum.
An important remark is the fact that reducing the generations also reduces the number of fitness calls. In a real-world scenario, in which the fitness function is extremely complex and computationally intensive, reducing the number of fitness function calls by such a factor can result in a significant speed up of the whole optimization process. The EQDR algorithm consistently requires less fitness function evaluations to obtain the optimum. The standard deviation of the number of generations required is lower in the EQDR algorithm as shown by the boxplots. This means that the improvement of the EQDR algorithm is reproducible among runs. This strengthens the results found, proving that the EQDR algorithm can consistently exploit the gathered genetical knowledge to guide the evolution towards the optimum.

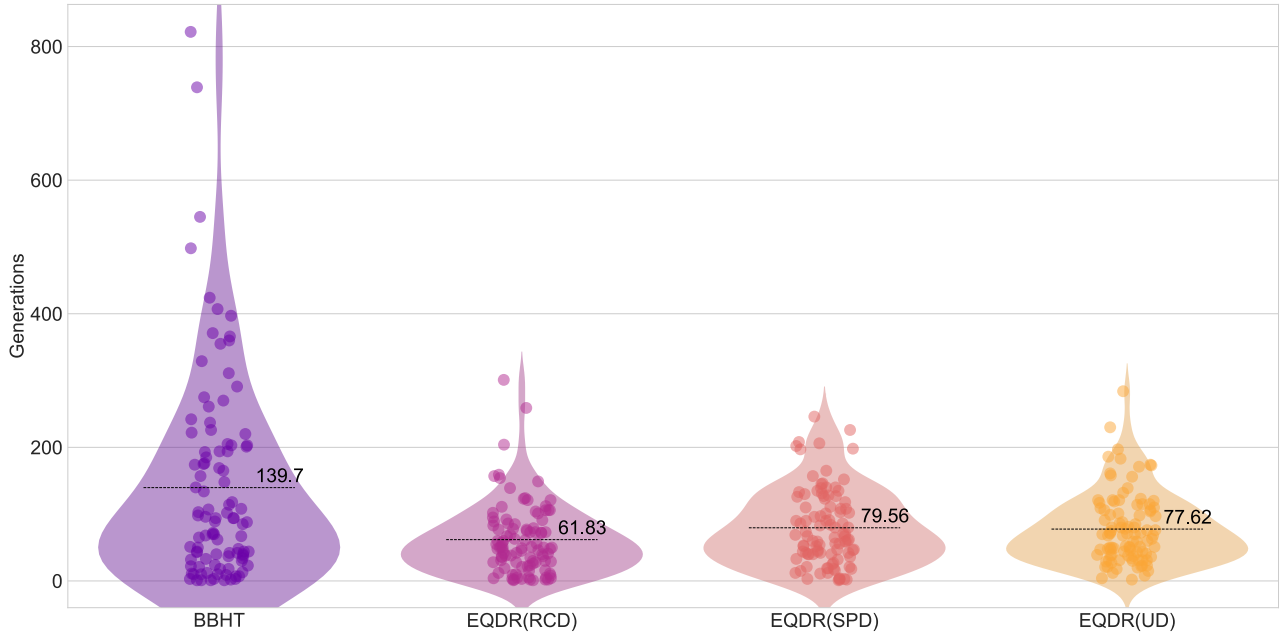

Figure 24: results in case of 8 qubits. Increasing the dimensionality of the system, the EQDR obtains a net advantage in finding the global optimum.

High precision oracle

Let us consider now an oracle with a much higher precision, described by the pattern below:

$$\tilde{s} = (** 001000), \qquad \varepsilon = \frac{1}{2^2} = \frac{1}{4}.$$

In this situation, the pure BBHT random samples solutions in a space similar to the actual solution space. Other parameters remain the same as the ones from the previous experiment.
As shown in Figure 25, the EQDR loses its advantage if the oracle used has a high precision. An oracle with high precision is able to recognize a set of solutions that almost coincides with the set of exact solutions. The oracle used in this experiment recognizes 4 elements as possible solutions: 00001000, 10001000, 10001000 and 11001000. By stochastically sampling these elements, the probability of obtaining the true solution is quite high. During this sampling process, the EQDR is not able to gather enough inter-generational genetic knowledge and exploit that information to guide the search process. In the few generations required to find the optimum, the EQDR effectively operates as the pure BBHT but with a small computational overhead.

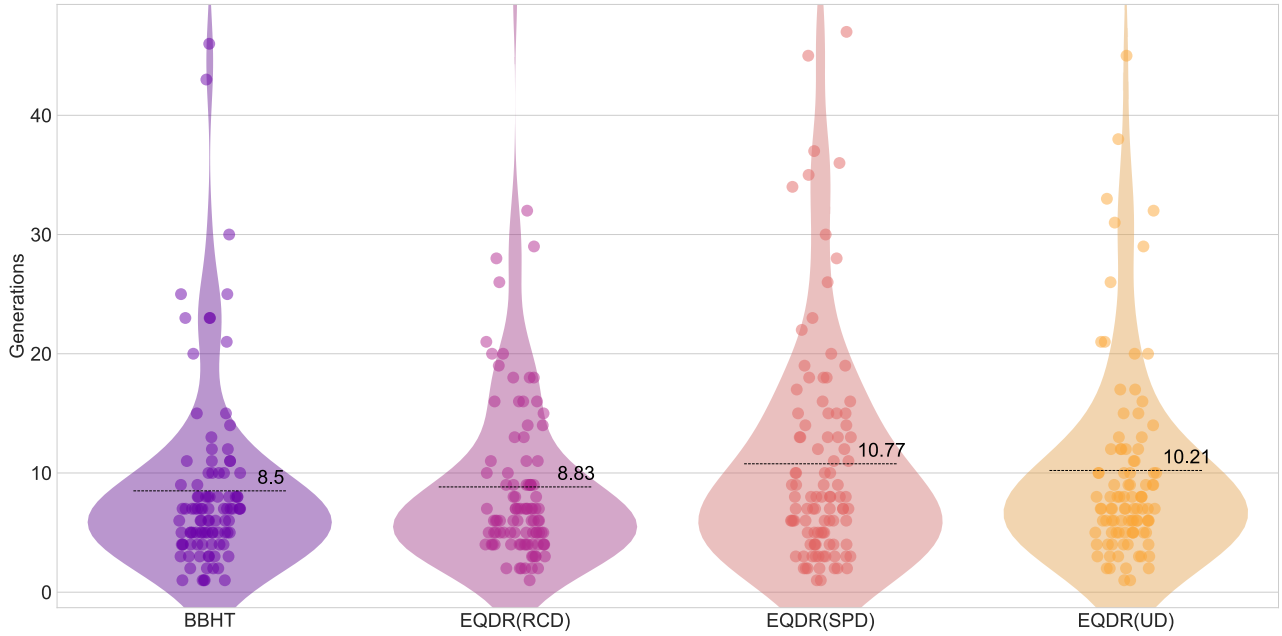

Figure 25: results in case of 8 qubits with an oracle with higher precision. Increasing the precision of the oracle makes the EQDR lose its advantage.

The only difference that can be noted is among the three different diffusion and accuracy vectors computation methodologies used. In both experiments analyzed, the EQDR with the RCD recombination methodology seems to have a better consistency. In fact, the RCD basically shows no outliers and is able to consistently find the optimum among different runs with a low standard deviation. The next section aims to clarify the pro and cons of the three different methodologies presented in this work.

3.4 Comparing diffusion and accuracy vectors computations

The experiment aims at analyzing the different performances achieved by the EQDR algorithm using the three different methodologies for computing the diffusion and accuracy vectors presented in Section 2.5. Recall that the computation of the diffusion and accuracy vectors is a classical procedure. As such, it is possible to use any kind of black-box procedure that is able to output two vectors of the correct length. Thanks to this modularity, implementing new methods for classically computing these vectors is straightforward. The experiment is executed in two different scenarios, with different oracles and fitness functions.

Square function

Let us consider a system having genomes of size k, the first function tested is a very simple unimodal square function having the minimum $x = k$:

$$f(x) = (x - k)^2$$

In this experiment, the dimension of the system is set to $k = 8$. The oracle implemented can be represented by its pattern $\tilde{s}$:

$$\tilde{s} = (*** 01 ***), \qquad \varepsilon = \frac{1}{2^6} = \frac{1}{64}.$$

The following table present the common parameters used for the EQDR algorithm.

Grover iteration scaling λ	$\lambda = 6/5 = 1.2$
Genetic pool size	$\lvert B \rvert = 14$
Recombination probability	$p_r = 0.65$
Mutation probability	$\eta = 0.5$

Mutation amplitude	$m_a = \pi/8$

The algorithm that uses the RCD recombination procedure uses the Gaussian scaling function γ_g with the hyperparameter α set to 0.125. The two other procedures do not entail the use of any scaling function and use a recombination accuracy $\rho = 0.45$.
Figure 26 shows the result of the experiment run. Each EQDR instance has been run for a total of 100 times.

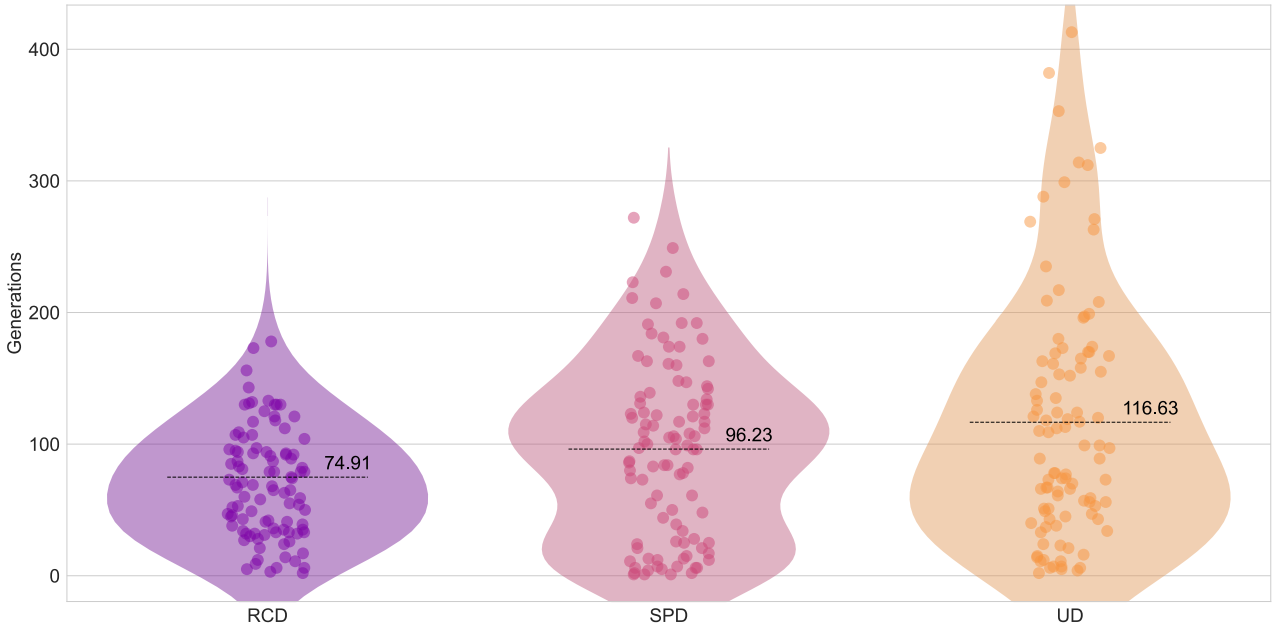

Figure 26: results in case of a unimodal fitness function. The three different methodologies for computing the diffusion and accuracy vectors are presented.

The RCD procedure has an advantage of around 50% over the UD and around 30% over the SPD. This is due to the fact that the RCD is better at exploiting good solutions. In the case of a unimodal fitness function, exploiting good solutions is always a good choice. Whenever a solution has a lower fitness, it is surely closer to the global optimum since the fitness function is unimodal. However, the SPD presents an improvement of around 20% with respect to the UD. Indeed, the SPD has greater exploitation capabilities compared to the UD, since individuals in the genetic pool are selected based on a probability distribution that depends on their fitness. This further strengthens the consideration that exploitation is always a good choice when optimizing unimodal fitness functions. In case of a fitness function that is not highly multimodal, the RCD can achieve the highest degree of exploitation, bringing a noticeable improvement in terms of generations required to find the optimum solution.

Binary knapsack

The second experiment is run on an instance of the binary knapsack problem. The problem can be described as follows: given a set of items, each associated with a weight and a value, determine which items to include in the collection so that the total weight is less than or equal to a given limit and the value is as large as possible. More formally, given a set of n items, numbered from 1 to n, each associated with a weight w_i and a value v_i, along with a maximum weight w_{max}:

$$\text{maximize} \sum_{i=1}^{n} v_i x_i$$

$$\text{subject to } \begin{cases} x_i \in \{0,1\}\ \forall i \\ \sum_{i=1}^{n} w_i x_i \leq w_{max} \end{cases}.$$

The problem is binary in the sense that each variable x_i can be either 0 or 1. This means that, differently from the non-binary knapsack problem, this version requires each item to be either taken or not.
The system is composed of 7 qubits in which the i-th element determines whether the i-th object is included in the solution. The optimum is identified by a sum of the values equal to 107 and the binary string that represents the solution is 1001000, meaning the solution corresponds to taking the first and the fourth elements. The table below shows the price for each element and the corresponding weight. The maximum weight w_{max} is set to 50.

Weight	Price	Index
31	70	0
10	20	1
20	39	2
19	37	3
4	7	4
3	5	5
6	10	6

The partial oracle used is described by the following pattern:

$$\tilde{s} = (1 ****\ 00), \qquad \varepsilon = \frac{1}{2^4} = \frac{1}{16}.$$

The Grover iteration scaling factor λ is set to 1.2 as in previous experiments.

EQDR common parameters

Genetic pool size	$\lvert B\rvert = 12$
Recombination probability	$p_r = 0.8$
Mutation probability	$\eta = 0.3$
Mutation amplitude	$m_a = \pi \cdot 0.15$

EQDR (RCD)

Diffusion recombination	Ranked contribution diffusion (RCD)
Scaling function	Polynomial: γ_p
Scaling function α	$\alpha = 0.7$

EQDR (SPD)

Diffusion recombination	Stochastic parent diffusion (SPD)
Recombination accuracy	$\rho = 0.8$

EQDR (UD)

Diffusion recombination	Uniform diffusion (UD)
Recombination accuracy	$\rho = 0.8$

Figure 27 shows the results of the experiment over a series of 100 runs. The advantage that the RCD showed in optimizing a unimodal fitness function is missing in this case. In particular, both the UD and the SPD show an overall average performance improvement of around 60% compared to the RCD. This can be reconducted to the fact that the binary knapsack problem is characterized by a highly multimodal fitness function. In fact,

differently from the Rastrigin function, the binary knapsack problem can have various local minima not evenly distributed into the search space.

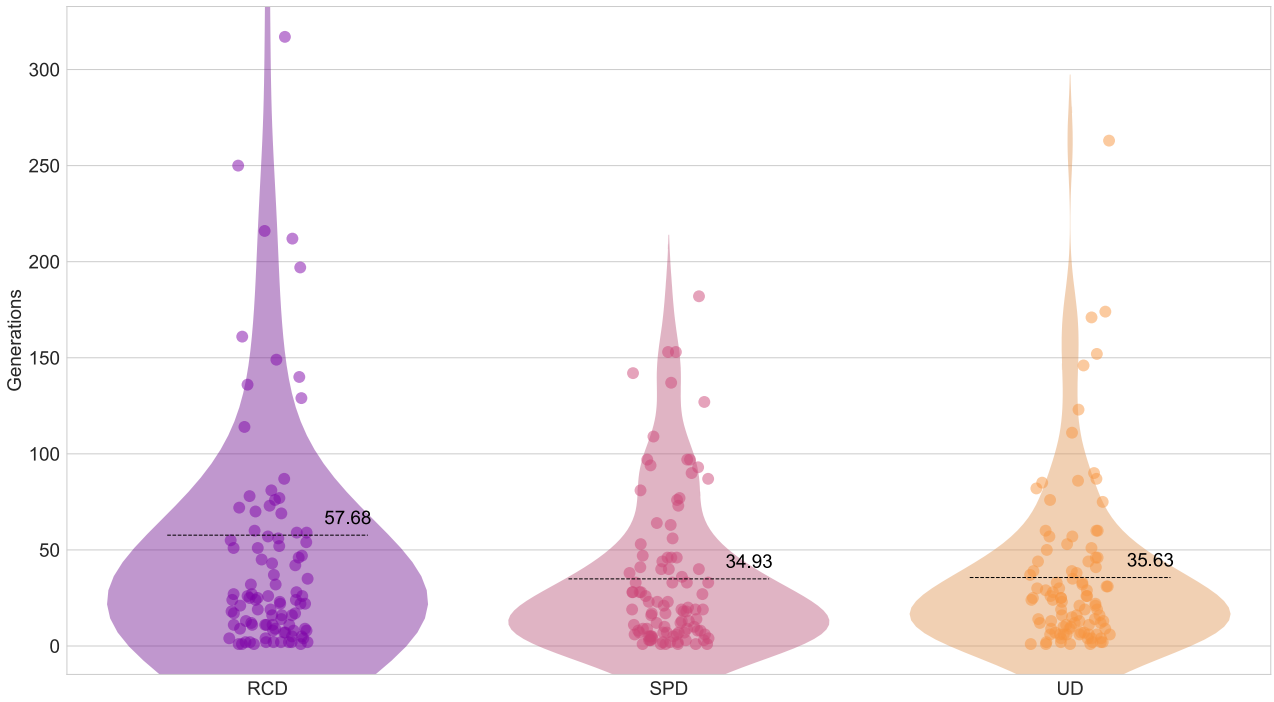

Figure 27: results in case of the binary knapsack problem. The plot shows that there is no advantage of the RCD in this case. In particular, the UD was able to achieve a performance improvement or around 16%.

It is possible that a solution that has a good fitness is far away from the global optimum, as local optima may be distributed unevenly in the whole search space. In this situation, focusing on exploitation may not be the best strategy. Exploiting good solutions at the cost of exploration may cause the algorithm to remain stuck around the current local optimum. For this reason, the RCD effectively performs worse compared to the SPD and UD. Although the average numbers of generations required to optimize the problem are similar in the SPD and the UD versions of the EQDR, it is important to notice that the two methods are very different. The SPD bases the selection of the elites in the genetic pool on a probability distribution based on the fitness. This necessarily means that the SPD maintains a relatively high degree of exploitation at the cost of exploration capabilities. Moreover, once a parent has been selected with the SPD, the single individual is responsible for determining the entirety of the diffusion vector $\hat{b}$, while in the UD each allele can be taken from different parents with equal probabilities.

The results found in this experiment highlight that, although the RCD methodology may bring performance improvements in certain cases, these benefits highly depend on the fitness function being optimized and on the hyperparameters selected. In a sense, the RCD does lack of genetic diversity and tends to remain stuck in a local optimum. This characteristic of the RCD may seem counterintuitive: the RCD effectively mixes the genomes of the elites inside the genetic pool in order to compute the diffusion vector. For this reason, one may think the RCD is able to generate diverse solutions at each run. However, it is important to notice that, in case the genetic pool is not updated, the diffusion vector computed will be always the same, with the mutation layer being the only responsible for introducing new genetic material. On the other hand, the SPD and UD methodologies allow to generate different solutions during different iterations even if the genetic pool remains unchanged. This is due to the probabilistic nature of the two methodologies and enables to explain the greater exploration capabilities of the UD and the SPD computation methodologies.

It must be taken into consideration that enhancing the mutation amplitude and probability, along with increasing the hyperparameter α of the scaling function, may bring some benefits to the RCD methodology. In fact, increasing the α value of the scaling function, enhances exploration among the elite individuals. As shown in the next section, hyperparameters play a fundamental role in adapting the different methodologies to

different situations. This requires a prior step of analyzing the problem at hand in order to select the best methodology.

3.5 Exploring relations

The EQDR algorithm has great number of hyperparameters that contribute to the overall performance of the procedure. Although these parameters allow the algorithm to adapt to different fitness functions and quantum oracles, it is undeniable that finding the optimum combination of values is extremely difficult and, as usual, it is done by trial and error. This section does not aim to find an automatic methodology to compute the best combination of parameters for any given problem. Instead, the goal is to clarify the fundamental role of the most important hyperparameters of the EQDR algorithm by analyzing how the performance varies depending on the hyperparameters values.

3.5.1 Gamma function and genetic pool size

The first hyperparameters analyzed are the scaling gamma function γ presented in Section 2.5.1 and the size of the genetic pool B. The gamma function is used exclusively in the framework of the ranked contribution diffusion. In the RCD methodology, each individual in the genetic pool contributes with a factor to enforce the value of its i-th allele into the i-th qubit of the quantum register. The function has the role to scale these contributions based on the rank of the different individuals. Recall that the scaling gamma function can be defined as a function $\gamma: \mathbb{N} \to [0,1]$ in which the input is the rank of the individual and the output is its relative contribution factor.
The two proposed gamma functions that are analyzed in this experiment are polynomial scaling function γ_p introduced in Equation (2.6)

$$\gamma_p(k) = \alpha^k, \qquad \alpha \in [0,1].$$

and the Gaussian gamma function γ_g presented in Equation (2.7)

$$\gamma_g(k) = e^{\frac{k^2 \, ln(\alpha)}{(|B|-1)^2}}, \qquad \alpha \in [0,1]$$

The parameter α defines the behavior of the function. In the case of the Gaussian gamma, the value of this parameter identifies the contribution associated to the individual with the lowest rank. In other words, it determines the minimum possible contribution. In the case of the polynomial function, the parameter defines the shape of the function and the rapidity with which the contributions decrease. Defining with α_g the hyperparameter of the gaussian function and with α_p the parameter of the polynomial function, it is important to remind the relation presented in Equation (2.8):

$$\alpha_g = (\alpha_p)^{|B|-1}.$$

If the alpha parameters are set based on this relation, the contributions corresponding to the least fit individual will match in both cases. This allows to fairly compare the two functions.
The experiment compares how the two different scaling functions react to variations in the parameter α and genetic pool size $|B|$. The problem selected to compare the two functions is the instance of the binary knapsack presented in the previous section. The reason for selecting the binary knapsack as the problem for this experiment is the fact that the problem is solved a bit more efficiently by the UD and SPD recombination methodologies due to their higher exploration capabilities. The goal of the experiment is to show that higher values of the α parameter in synergy with a higher genetic pool size does allow enhancing the exploration capabilities of the algorithm equipped with the RCD. The dimensionality of the system is set to 7 qubits and the pattern of the oracle used is:

$$\tilde{s} = (1 \ast\ast\ast\ast \, 00), \qquad \varepsilon = \frac{1}{2^4} = \frac{1}{16}.$$

The experiment consists of two instances of the EQDR algorithm, one with the Gaussian gamma function and one with the polynomial function. Each instance is executed a total of 100 times with a high α value and a high genetic pool size and other 100 times with a small α value and a smaller genetic pool size.
The table below presents the fixed hyperparameters used. Notice that Equation (2.8) is satisfied in the values used for the hyperparameter α.

Grover iteration scaling λ	$\lambda = 6/5 = 1.2$
Recombination probability	$p_r = 0.8$
Diffusion Recombination	Ranked contribution diffusion (RCD)
Mutation probability	$\eta = 0.4$
Mutation amplitude	$m_a = \pi/5$

Gaussian γ_g

Low exploration	$\alpha_g = (0.5)^3 = 0.125, \quad \lvert B\rvert = 4$
High exploration	$\alpha_g = (0.975)^{13} \cong 0.719, \quad \lvert B\rvert = 14$

Polynomial γ_p

Low exploration	$\alpha_p = 0.5, \quad \lvert B\rvert = 4$
High exploration	$\alpha_p = 0.975, \quad \lvert B\rvert = 14$

In both instances, the EQDR was able to achieve good performance on the binary knapsack problem by enhancing exploration capabilities. Figure 28 and Figure 29 show the results plotted for both the instances of the EQDR. The performance difference between the gaussian gamma function and the polynomial gamma function are negligible. The results achieved by the EQDR equipped with the RCD with boosted exploration are slightly better than the results obtained by the UD and SPD in the previous experiment on the binary knapsack problem. This entails that it is possible to use the RCD methodology in highy multimodal fitness functions by enhancing its exploration capabilities.

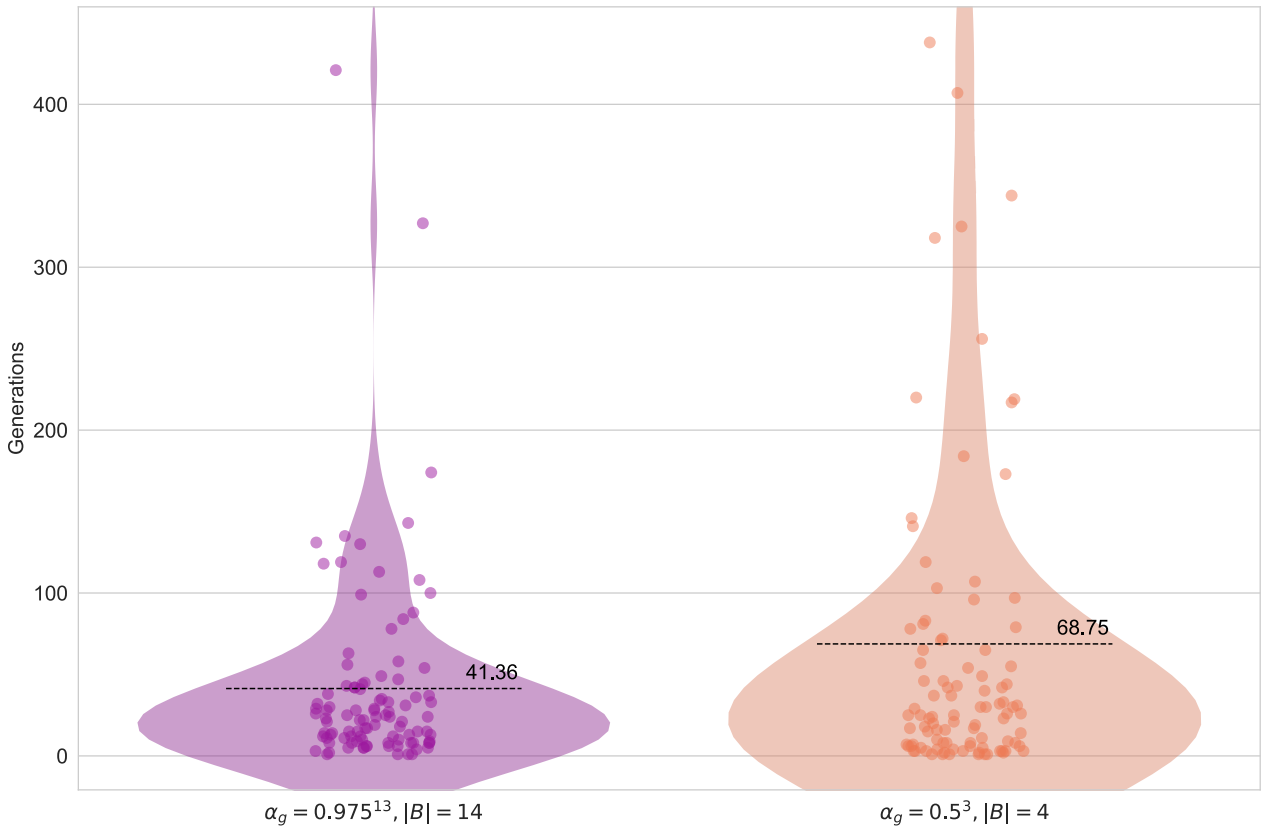

Figure 28: gaussian gamma function results with different hyperparameters profiles. Increasing both the parameter α_g and the dimension of the genetic pool $|B|$, enables the EQDR with the RDC recombination methodology to enhance exploration.

Increasing both the hyperparameters α and $|B|$, enables the EQDR equipped with the RCD recombination methodology to enhance the performance on the binary knapsack problem. A greater genetic pool B, enables the algorithm to maintain a greater genetic diversity during different oracle applications. This results in a larger genetic pool of alleles that can be used to form the offspring during the recombination procedure. In synergy, increasing the parameter α of the scaling function, enables the individuals to be considered more equally, effectively reducing the impact of their rank during the recombination procedure. These two characteristics allow the EQDR equipped with the RCD to enhance exploration. In this case, the EQDR using the RCD with enhanced exploration obtained a remarkable performance improvement with respect to the previous version presented in Section 3.4.

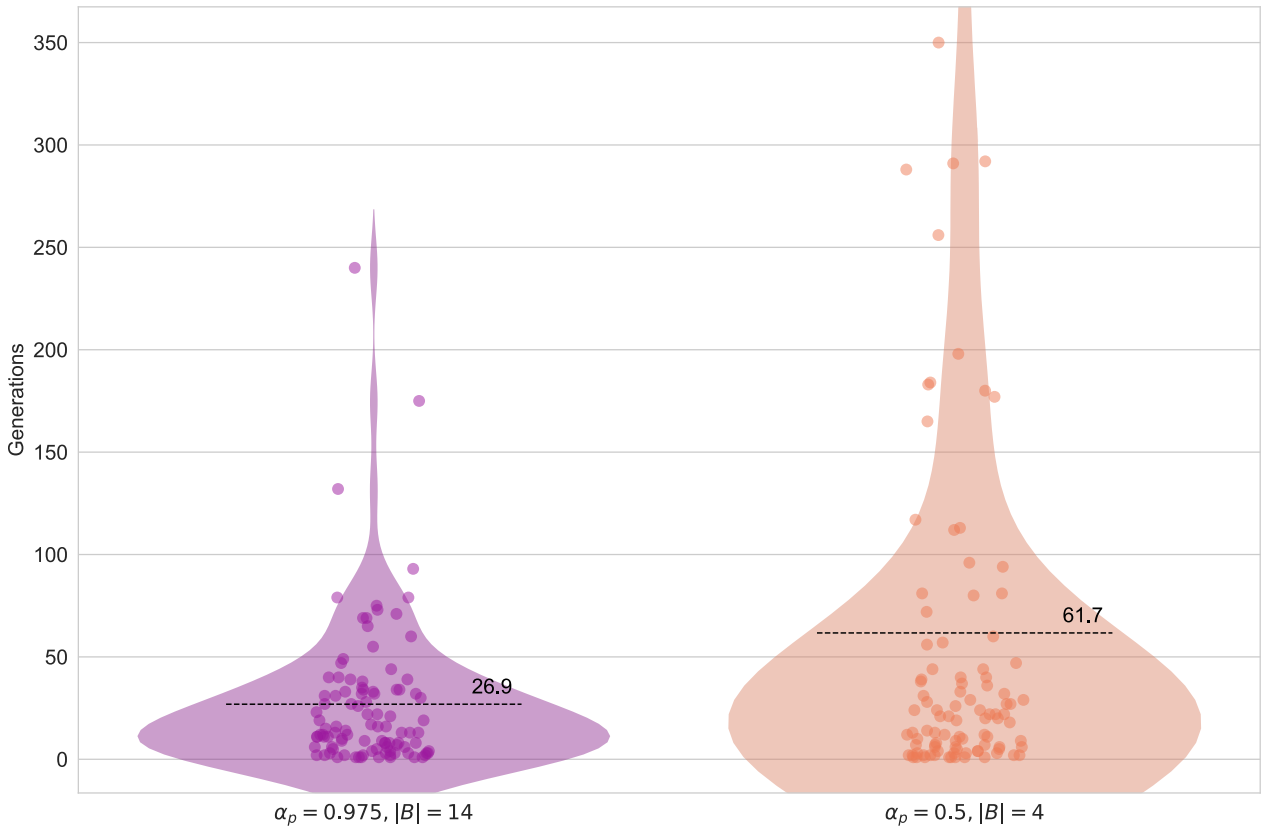

Figure 29: polynomial gamma function results with different hyperparameters profiles. Increasing both the parameter α_p and the dimension of the genetic pool $|B|$, enables the EQDR with the RDC recombination methodology to enhance exploration.

The instance of the EQDR that used the polynomial scaling function, a higher value of the alpha parameter and a larger genetic pool, performed better than the EQDR equipped with the UD recombination methodology presented in the previous section. The performance improvement is around 30%. This result shows that the RCD is highly subjective to the hyperparameters selected. This sensitivity can be considered a double-edged sword: on one hand it requires important values to be set a priori, on the other hand it enables the RCD to adapt to different situations. However, some knowledge regarding the problem at hand is always useful in order to select the best version of the EQDR algorithm.

3.5.2 Quantum oracle influence

The quantum oracle is the main component of a quantum search algorithm. In Grover's algorithm, the oracle needs to recognize all the true solutions of the problem. In the EQDR framework, it is possible to exploit also partial oracles, quantum operators that are able to recognize a superset of solutions that includes both true solutions and solutions. As shown in Section 1.2.2, partial quantum oracles can be associated with a precision. Under a theoretical viewpoint, substantially decreasing the oracle precision corresponds to increasing the number of solutions that are considered valid. In particular, an oracle with the lowest possible precision, equal to $\varepsilon = 1/2^n$, is described by the pattern:

$$\tilde{s} = (*)^n,$$

where the power represents the repetition of the symbol for n times. The goal of this experiment is to quantify the influence of the oracle precision in solving a certain optimization problem. The problem selected is the multimodal Rastrigin fitness function presented in Section 3.3.1 with a system of dimension $\kappa = 8$.
Three instances of the EQDR algorithm are run with three different oracles with different precisions:

$$\tilde{s}_1 = (******* 0), \qquad \varepsilon = \frac{1}{2^7} = \frac{1}{128}$$

$$\tilde{s}_2 = (0 ** 10 ** 0), \qquad \varepsilon = \frac{1}{2^4} = \frac{1}{16}$$

$$\tilde{s}_3 = (0 * 0100 * 0), \qquad \varepsilon = \frac{1}{2^2} = \frac{1}{4}.$$

Common parameters for the three instances are presented in the table below.

Grover iteration scaling λ	$\lambda = 6/5 = 1.2$
Genetic pool size	$\lvert B \rvert = 12$
Recombination probability	$p_r = 0.8$
Diffusion Recombination	Ranked contribution diffusion (RCD)
Scaling function	Gaussian: γ_g
Scaling function α	$\alpha = 0.3$
Mutation probability	$\eta = 0.35$
Mutation amplitude	$m_a = \pi/5$

The results displayed in Figure 30 show that, as expected, the quantum oracle plays a fundamental role in the EQDR algorithm. Increasing the precision by a power of 3 massively improves performance. The graph below suggests that the performance improvement obtained by increasing the precision follows an exponential trend. Quantum oracles are very complicated to design and implement. However, the presented trend confirms that, whenever possible, increasing the precision by a single power of two can indeed be enough to obtain remarkable speedups.

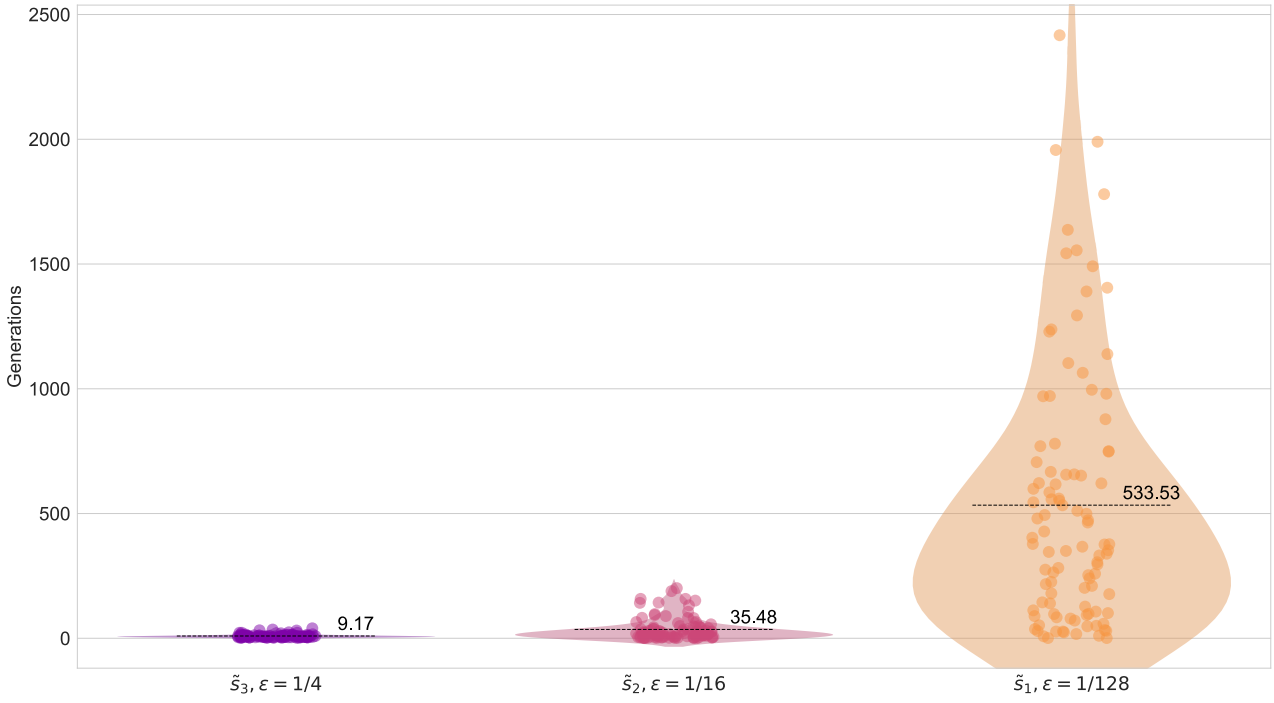

Figure 30: results of different oracles with different precisions. The oracle is the main component that influences the performance of the algorithm.

In order to identify the actual trend of the number of generations based on the oracle precision, an exponential function with three free parameters has been fit on the data using a simple curve fitting algorithm. The function has the form:

$$g(x) = a \cdot x^b + c.$$

It is important to note that the plot does not use the precision values on the x axis. Instead, the number of unrecognized indexes in the system of dimension $\kappa = 8$ is used. Recalling from Section 1.2.2 that the number of indexes recognized by an oracle is identified by the symbol $|K|$, the number of unrecognized indexes is defined as $n - |K|$. The x axis in Figure 31 represents different values of $x = n - |K|$. The results prove that there is a clearly defined exponential trend. Thus, it is possible to obtain an exponential performance improvement by optimizing and improving the quantum oracle.

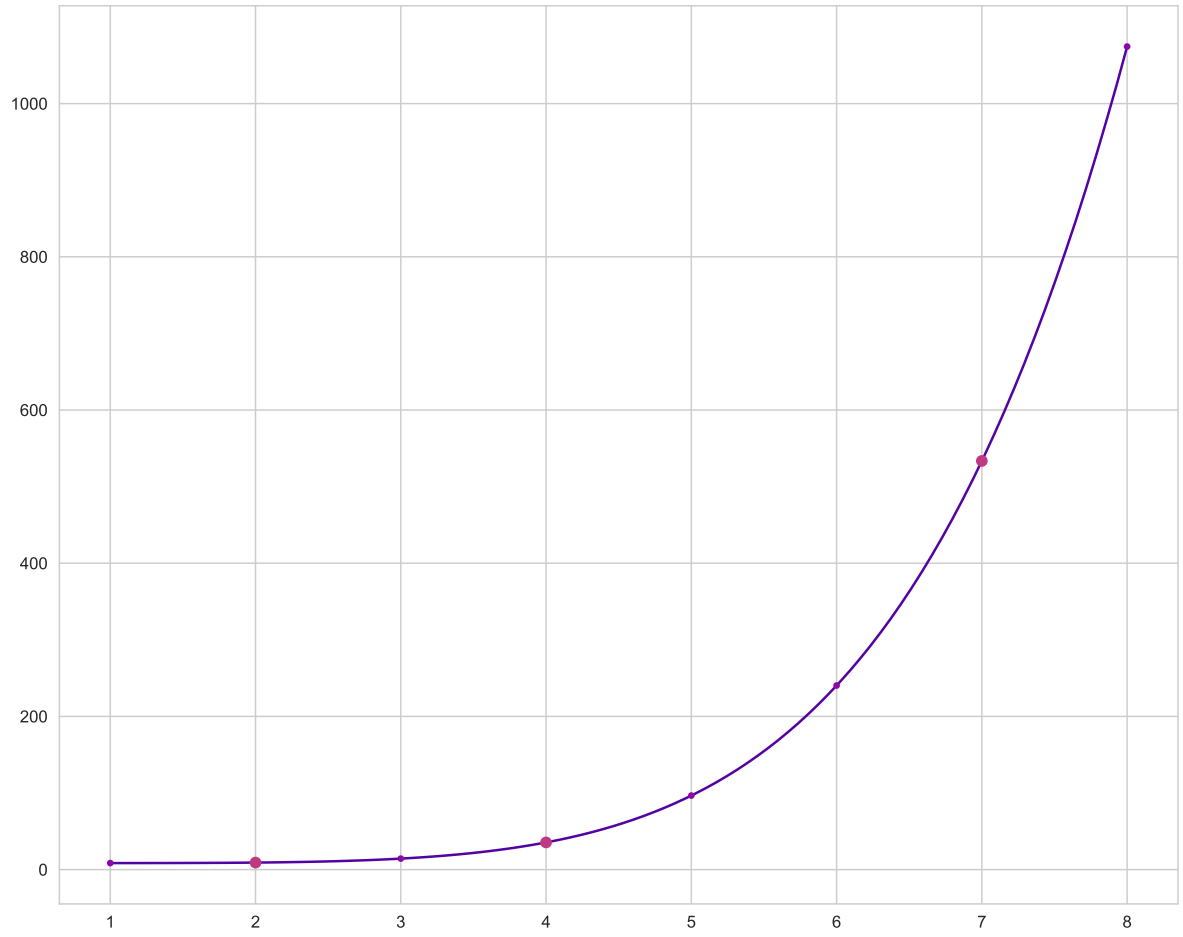

Figure 31: fitted exponential curve displaying the generations required based on oracle precision. Bigger dots represent the data points obtained by simulations. The parameters found are: $a \cong 0.017, b \cong 5.303, c \cong 8.486$.

3.5.3 Grover iterations scaling λ

Recall that the BBHT algorithm has the goal of finding a solution among a set of solutions of unknown size. Since the number of solutions is unknown, it is impossible to decide a priori the optimal number of Grover iterations to apply. To achieve its goal, the algorithm slightly increases the maximum number of Grover iterations m that can applied after each iteration, according to the equation:

$$m = min(\lambda m, \sqrt{N})$$

where n is the number of qubits in the register. The algorithm is assured to obtain a result in an expected number of iterations that is $O(\sqrt{N})$ [5]. The parameter $\lambda \in \left[1, \frac{4}{3}\right]$ defines the scaling of the number of iterations required. Increasing the number of Grover iterations should result in a stronger amplification of the coefficients related to solutions in the quantum superposition. However, in the case of partial oracles, the value λ should somehow depend on the precision of the oracle. There is no effective use in enforcing the results of the oracle

if the oracle has a low precision. On the other hand, it may result beneficial to increase the value of the hyperparameter λ in case the oracle is close to the optimal one. In the case of an oracle with high precision, higher values of the parameter λ enable the EQDR algorithm to find the optimum in a lower expected number of generations.
The experiment consists in using two different oracles with different precisions and analyzing the results as a function of the hyperparameter λ. The problem optimized is the Rastrigin fitness function with dimensionality $\kappa = 8$. Two different oracles are used, $\tilde{s}_1$ and $\tilde{s}_2$:

$$\tilde{s}_1 = (0 * 0010 * 0), \qquad \varepsilon = \frac{1}{2^2} = \frac{1}{4}$$

$$\tilde{s}_2 = (0 ****** 0), \qquad \varepsilon = \frac{1}{2^6} = \frac{1}{64}.$$

Four instances of the EQDR are used. The instances are constructed with one of the two oracles presented above and one value of the parameter λ. Each instance has been executed for 100 times. Each oracle is run with both values of the hyperparameter λ.

Oracle	λ
$\tilde{s}_1 = (0 * 0010 * 0)$	$\lambda \in \{1, 4/3\}$
$\tilde{s}_2 = (0 ****** 0)$	$\lambda \in \{1, 4/3\}$

In the case of the first oracle with high precision, the results are presented in Figure 32. The trend is extremely marked and it is clear that increasing the value of λ whenever the oracle has a high precision does bring impressive results. This is due to the fact that higher values of the scaling parameter result in a faster amplification of the coefficients corresponding to the solutions.

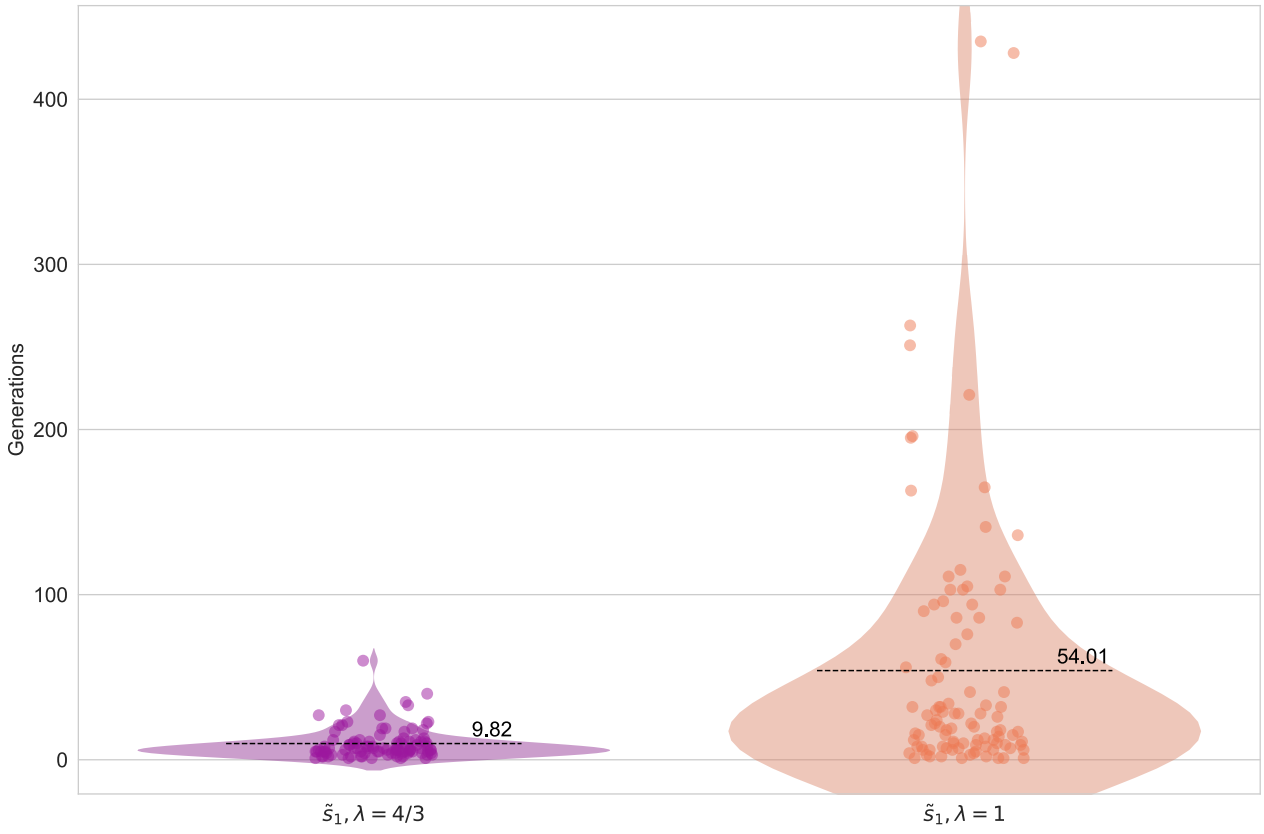

Figure 32: results of the oracle with high precision for two different values of the scaling parameter λ.

In the case of the oracle $\tilde{s}_2$ with low precision, the version of the EQDR with the parameter λ set to $4/3$ was not able to conclude all the 100 runs even after multiple restarts and remained stuck, preventing the test to be completed. On the other hand, decreasing the parameter λ to 1 enabled the algorithm to complete the test with the results showed in Figure 33.

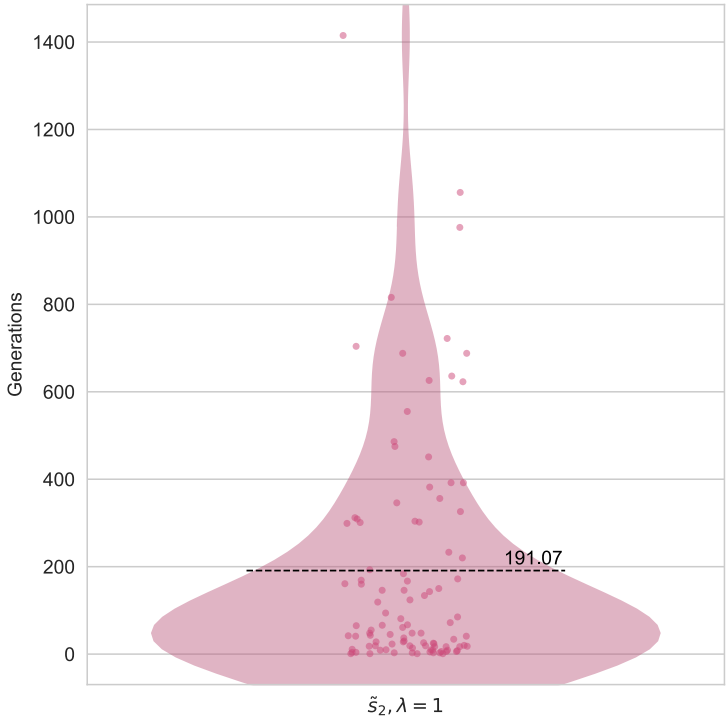

Figure 33: results of the oracle with low precision for the instance with $\lambda = 1$. The other instance with $\lambda = \frac{4}{3}$ was not able to complete the 100 runs even after multiple restarts.

The results of the experiment show that the optimal value for the hyperparameter λ depends on the quantum oracle selected. The finding is that it is better to use lower values of the λ parameter whenever the precision ε of the oracle is low. On the other hand, if the oracle has a high precision, using larger values for λ results in a remarkable performance improvement.
A possible solution to choose a suitable value for the Grover iteration scaling is to make it depend linearly on the precision of the oracle. Let us consider a system of dimension n. A generic quantum oracle has a precision $\varepsilon \in [2^{-n}, 1]$, while the parameter λ must be in the range $\left[1, \frac{4}{3}\right]$. Recalling from Section 1.2.2, the symbol $|K| \in [0, n]$ can be used to identify the number of indexes that the oracle recognizes. The parameter λ can therefore be set as:

$$\lambda = 1 + \frac{|K|}{3n}.$$

The equation presented above simply linearly remaps the number of recognized indexes of the oracle into the domain $\left[1, \frac{4}{3}\right]$. As an example, let us consider two oracles O_1 and O_2 for a system with dimension $n = 10$. The two oracles have precisions respectively $\varepsilon_1 = \frac{1}{2^2} \Rightarrow |K_1| = 8$ and $\varepsilon_2 = \frac{1}{2^7} \Rightarrow |K_2| = 3$. The two values λ_1 and λ_2 can be computed as:

$$\lambda_1 = 1 + \frac{|K_1|}{3n} = 1 + \frac{8}{30} = \frac{19}{15}$$

$$\lambda_2 = 1 + \frac{|K_2|}{3n} = 1 + \frac{3}{30} = \frac{11}{10}.$$

It is worth noting that the experiments carried out in this section have been intentionally designed to show the importance of the hyperparameter λ in limit cases. However, as shown in the previous sections, usually setting λ equal to the value suggested by Dürr and Høyer (1996) in the original paper of the BBHT algorithm enables the EQDR to achieve good results.

4 Conclusions

The EQDR algorithm presented in this work has the goal of enabling the application of quantum algorithms for fitness-based optimization problems through the exploitation of both quantum search algorithms and evolutionary computation notions. The main characteristic of the EQDR algorithm is the possibility to use partial oracles. This allows the optimization process to take place even in situations in which the optimal oracle is either very difficult to implement or simply extremely noisy due to the NISQ era devices available nowadays. In order to effectively use partial oracles, the EQDR algorithm borrows notions from the evolutionary computation framework. In particular, the concept of elitism is used to gather inter-generational knowledge and guide the optimization.

Grover's algorithm is the main building block of the EQDR algorithm. The quantum search is run with a partial oracle that returns a superposition of possible solutions. At this point, the algorithm uses the genetic knowledge gathered to apply two quantum operators that, acting on the amplitudes of the superposition, are able to alter the probabilities of the various results. These two operators are the diffusion recombination operator and the superposition mutation. The diffusion recombination step enforces a certain classical state into the quantum register. This classical state is computed by mixing the chromosomes of the elites. This procedure has proven to be able to deterministically enforce a certain classical state into a quantum register. However, determinism is not the best option when dealing with heuristics. For this reason, a certain degree of stochasticity has been introduced by applying the recombination step to each qubit with a certain probability. In this way, it is possible to prevent measurements that would cause the wave function to collapse while stochastically driving the evolution toward a general state that effectively represents the ensemble of chromosomes of the elites. At this point, the mutation step applies variational rotational quantum gates to reintroduce the genetic diversity that may have been lost during the recombination step.

An important remark is the fact the mutation step is not a fundamental operator. It is known from the evolutionary computation theory that any genetic algorithm cannot run without mutation. The reason is the fact that mutation is the only non-deterministic procedure that is able to introduce new genetic material into the population. In the quantum world, the measurements are inherently non-deterministic. This means that a certain degree of stochasticity is always present in the quantum register. The mutation operator in the quantum world has the job of increasing this stochasticity degree in order to enhance the exploration capabilities of the algorithm. Finally, the quantum state is measured and the actual fitness of the result is computed classically. Since the classical fitness is computed exclusively on measured individuals, the number of fitness function calls is exactly one per generation. This is a remarkable result since many optimization problems entail the computation of extremely complicated and computationally intensive fitness functions.

In the experimental section, the EQDR algorithm has been validated in various situations. The comparison against the pure BBHT algorithm has shown promising results. In particular, the performance improvement, in terms of the number of generations required to find the optimum, increases with the dimensionality of the quantum register, hence the number of qubits. This result suggests that the ability to coherently scale quantum hardware could open the road to very important performance improvements in the world of quantum optimization. However, it has been proven that, in agreement with the no free lunch theorem, the EQDR algorithm does not perform better in any situation. Whenever the precision ε of the partial oracle increases to a value close to 1, the EQDR algorithm loses its advantage. In this case the BBHT algorithm is able to random sample from a superset of solutions that almost coincides with the set of true solutions. The EQDR algorithm requires some time in order to gather inter-generational knowledge. During the initial generations, the algorithm effectively operates as the pure BBHT, but with a small computational overhead. In these situations, it has been shown that the pure BBHT algorithm can in fact find the optimum in an expected number of generations that is equal or even lower than the EQDR algorithm.

The influence of the precision of the partial quantum oracle has been quantified. Oracles with higher precision correspond to more efficient algorithms. As the number of unrecognized indexes linearly decreases, the performance loss, in terms of the number of generations required to find the optimum, is exponential. This entails that, although the EQDR is able to adapt its search by exploiting genetic knowledge, an efficient and precise partial quantum oracle is a mandatory requirement for the effectiveness of the proposed algorithm.

Each component of the EQDR algorithm has been analyzed and its function determined in the experimental tests. The adaptability of the EQDR algorithm is another point of strength. A particular focus on exploitation and exploration has been given. By tweaking the correct hyperparameters, it is possible to adapt the EQDR algorithm to different situations. Thus, having prior knowledge of the problem that is being solved allows adapting the algorithm in order to obtain more consistent result.
The experiment on the binary knapsack problem has shown how the different diffusion and accuracy vectors computation methods can influence the performance of the algorithm in optimizing highly multimodal fitness functions. Another point of strength of the proposed method is its hybrid nature. In particular, the computation of the diffusion and accuracy vectors is a classical procedure. In this sense, it is possible to exploit any kind of black box operation that, provided the input, is able to output the two vectors in the correct form. The modularity of the EQDR algorithm allows devising methods for computing these vectors that better adapt to the problem at hand. Three different methods, each with its strengths and weaknesses, have been proposed and validated through experimental results.
In conclusion, the EQDR algorithm does bring a noticeable improvement in certain situations and on certain classes of problems with respect to the BBHT algorithm. The main reason is of course the possibility to mitigate the loss of precision due to the use of quantum partial oracle by exploiting evolutionary computation notions. Depending on the problem that is being optimized and the nature of the quantum oracle, the proposed algorithm does indeed represent a valid alternative. In situations in which the quantum oracle is only partially known or can only partially be developed, and the fitness function is extremely complicated, the EQDR can bring a remarkable advantage in the world of quantum optimization. Another important consideration is the fact that the proposed algorithm does not only bring performance improvements, but also extends quantum search algorithms such as Grover's algorithm to be used for fitness-based optimization problems, which are nowadays very common in the topic of performance engineering.
A final note is about the implementation of the EQDR algorithm. The code has been written using the Qiskit [9] framework which enables seamless interoperability among simulations and quantum hardware. However, the algorithm comprehends classical and quantum sections; in such a way the EQDR is a hybrid algorithm. Such algorithms require the capability of switching between quantum subroutines and classical ones with ease and with limited overhead in order to be executed efficiently. In the current era, NISQ devices have an extremely high initialization overhead and such an algorithm would be impossible to execute with success.
The code is practically ready to be run on a real quantum device. However, the algorithm entails the use of multiple quantum subroutines in a single iteration. At the moment of writing, the average waiting time for a single subroutine on a real quantum IBM hardware is in the order of hours, while, on the simulator, a single iteration does only take a magnitude of time in the order of minutes. For this reason, all the experiments have been run on the Qiskit's Aer Simulator [10] backend.

4.1.1 Related work

The idea of fusing the worlds of quantum computing and evolutionary computation has been around for quite some time. As summarized by Lahoz-Beltra (2016), there are a various number of methodologies that entail the use of quantum features in an evolutionary computation framework. From circuit-based quantum computing [11] [12] to adiabatic quantum computing [13] [14], the idea is to exploit the non-determinism of quantum mechanics to devise novel hybrid evolutionary computation algorithms. The most common solutions are based on Grover's algorithm. In particular, since the number of solutions is unknown, the BBHT algorithm, a variant of Grover's algorithm, is used. An interesting possible solution is to apply the minimum finding quantum algorithm presented by Dürr and Høyer (1996) to the balanced superposition of all possible solutions. This algorithm is a special version of the BBHT algorithm and requires an oracle that marks all the solutions that are associated with a value lower than a predefined one. In this way, running multiple iterations and updating each time the predefined minimum value, it is possible to obtain one of the elements associated with the minimum value with high probability. Although very simple and efficient, this solution requires an oracle that implements a fitness function and is able to compare two individuals to mark the fitter one.
Another possibility is to use the quantum search as a selection operator, as shown by Malossini, Blanzieri et al. (2008). The quadratic speedup that can be achieved by quantum search is not the only advantage in using the BBHT algorithm as a selection operator. Indeed, it is possible to modify the number of times the Grover

iteration is applied in order to modulate the amount of stochasticity introduced in the solutions. Reducing the number of times the Grover iteration is applied results in a remarkable speedup in terms of running time. The drawback of using quantum features solely to devise the selection operator is that the crossover and mutation operators must be applied classically. Moreover, each methodology based on the BBHT algorithm requires a well-defined quantum oracle. As presented, quantum oracles are very difficult to implement. To devise an optimal oracle for a fitness-based optimization problem means translating the fitness function into the quantum realm. This can be very difficult to achieve, if not impossible in some cases. For these reasons, in the last few years, the number of works that attempt to define iterative algorithmic ways to design quantum oracles has been steadily increasing [6] [8] [15] [7].

Another possible way to offload the burden placed on quantum oracles is to use partial quantum oracles, often called smaller oracles. These operators are simpler quantum oracles that are able to recognize only certain features of the solutions [16]. However, to use partial oracles in quantum search algorithms, there is the need to compensate for the loss of precision. The concept of partial oracles perfectly fits the heuristic nature of evolutionary computation. In a heuristic, determinism is not the best option as a certain degree of stochasticity in the solutions is required in order to prevent premature convergence. Quantum search algorithms that entail the use of partial oracles usually edit the scaling factor λ of the BBHT algorithm in order to reduce the number of Grover iterations applied at each generation. This causes a larger number of possible solutions to be returned by the quantum search routine, effectively increasing the exploration capabilities of the algorithm. Apart from exploiting quantum features to enhance the selection procedure of an evolutionary computation algorithm, it is possible to use quantum computing to devise operators that effectively act as the quantum counterpart of crossover and mutation. The operator presented by Acampora et al. (2022) exploits quantum non-determinism to perform a truly random recombination of classical states. The operator classically counts the frequencies of bits of a set of classical states and applies quantum rotational gates to generate a quantum superposition with amplitudes proportional to the frequencies computed. The rotational gates are applied to a well-defined initial state and generate a superposition that, upon measurement, will output a single classical state. Being generated from a set of classical states, the offspring generated is effectively an ensemble of the parents. Hence, the operator acts in the same way as a classical recombination operator. However, due to the problem related to rotational gates applied to unknown states for state biasing presented in section 2.3, the operator cannot be applied with the same results to an aleatory quantum state returned by Grover's algorithm. The idea is to delay the measurement of the state as much as possible. This allows maintaining the information contained in the superposition of states returned by Grover's algorithm in order to perform crossover and mutation in a quantum environment. Hence, the recombination is required to be executed with purely quantum operators.

Future work

The purpose of this work was to devise, develop and validate a new hybrid quantum algorithm to be applied to fitness-based optimization problems. Numerous steps are required in order to better understand the improvements of the algorithm and the class of problems that may be optimized by the EQDR more efficiently. An extensive performance analysis of the EQDR algorithm, compared with state-of-the-art quantum optimization techniques, may allow understanding the potential of the proposed algorithm.

The modularity of the algorithm allows considering each procedure as a standalone module. This characteristic enables to test how different modules implemented in different ways contribute to the overall performance of the algorithm. In particular, the diffusion recombination step can be modified with ease. A novel quantum recombination operator that can exploit the gathered classical information and use it to act on the quantum superposition could be studied, allowing the EQDR to be applied to different type of problems. Moreover, the classical computation of the diffusion and accuracy vectors can be easily adapted to different problems. A study regarding the effectiveness of the proposed methods and the design of new computation methodologies may allow the EQDR algorithm to solve classes of problems not treated in this work.

Finally, the EQDR algorithm is susceptible to the numerous hyperparameters. The role and impact of each hyperparameter need to be clarified. The proposed cues regarding an extensive study of the EQDR algorithm, alongside a further development of quantum hardware in the next years, with the consequent resolution of the challenges of quantum computing, could enable the proposed Elite Quantum Diffusion Recombination algorithm to bring remarkable advantages to the novel topic of fitness-based quantum optimization.

5 Bibliography

[1] "Glosser.ca," [Online]. Available: https://commons.wikimedia.org/w/index.php?curid=23263326.

[2] M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2010.

[3] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Journal on Computing,* vol. 26, p. 1484–1509, 25 Jan 1997.

[4] L. K. Grover, "A fast quantum mechanical algorithm for database search," *STOC '96: Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing,* p. 212–219, 19 November 1996.

[5] M. Boyer, G. Brassard, P. Høyer and A. Tapp, "Tight bounds on quantum searching," *Fortschritte der Physik,* vol. 46, pp. 187-199, 10 May 1998.

[6] J. M. Henderson, E. R. Henderson, A. Sinha, M. A. Thornton and M. D. Miller, "Automated Quantum Oracle Synthesis with a Minimal Number of Qubits," in *Proc. SPIE 12517, Quantum Information Science, Sensing, and Computation XV*, Orlando, 2023.

[7] K. Murakami and J. Zhao, "AutoQC: Automated Synthesis of Quantum Circuits Using Neural Networks," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, Fukuoka, 2022.

[8] B. Schmitt and G. De Micheli, "tweedledum: A Compiler Companion for Quantum Computing," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2022.

[9] "Qiskit," [Online]. Available: https://qiskit.org/.

[10] "Qiskit Aer documentation," [Online]. Available: https://qiskit.org/ecosystem/aer/.

[11] G. Jian, Y. Zhong-gen, W. Ru-chuan and S. Li-juan, "An improved quantum genetic algorithm," in *International Conference on Genetic and Evolutionary Computing (WGEC)*, Nanjing, 2009.

[12] G. Zhang, L. Na, W. Jin and L. Hu, "Novel Quantum Genetic Algorithm and Its Applications," *Frontiers of Electrical and Electronic Engineering in China,* pp. 31-36, 2006.

[13] S. A. Abel, N. A. Luca and M. Spannowsky, *A genetic quantum annealing algorithm,* Durham: Institute for Particle Physics Phenomenology, 2022.

[14] H. Bingjiao and S. Wanneng, "A Quantum Genetic Simulated Annealing Algorithm for Task Scheduling," *ISICA 2007: Advances in Computation and Intelligence,* pp. 169-176, 2007.

[15] R. Seidel, C. K.-U. Becker, S. Bock, N. Tcholtchev, I.-D. Gheorge-Pop and M. Hauswirth, "Automatic generation of Grover quantum oracles for arbitrary data structures," *Quantum Science and Technology,* vol. 8, no. 2, 14 October 2023.

[16] D. Li, Q. Ling, Z. Yu-Qian and Y. Yu-Guang, "Quantum partial search algorithm with smaller oracles for multiple target items," *Quantum Information Processing,* no. 160, 2022.

[17] G. Acampora, R. Schiattarella and A. Vitiello, "Quantum mating operator: a new approach to evolve chromosomes in genetic algorithms," *IEEE Congress on Evolutionary Computation (CEC),* 2022.

[18] G. Debabrata and K. Niraj, "Quantum algorithm to solve a maze: converting the maze problem into a search problem," in *Asian Quantum Information Science(AQIS'13)*, Kanpur, 2013.

[19] C. Dürr and P. Høyer, *A quantum algorithm for finding the minimum,* Cornell University, 1996.

[20] R. Lahoz-Beltra, "Quantum genetic algorithms for Computer Scientists," *Computers,* vol. 5, no. 4, 15 October 2016.

[21] A. Malossini, E. Blanzieri and T. Calarco, "Quantum genetic optimization," *IEEE Transactions on Evolutionary Computation,* vol. 12, no. 2, April 2008.

[22] F. M. Creevey, C. D. Hill and L. C. L. Hollenberg, "GASP: a genetic algorithm for state preparation on quantum computers," *Scientific Reports,* vol. 13, 2023.

[23] D. A. Sofge, "Prospective Algorithms for Quantum Evolutionary Computation," *Proceedings of the Second Quantum Interaction Symposium (QI-2008),* 2008.

[24] J. Supasil, P. Pathumsoot and S. Suwanna, "Simulation of implementable quantum-assisted genetic algorithm," *Journal of Physics: Conference Series*, vol. 1719, 2020.

[25] K. Zhang and V. Korepin, "Quantum partial search for uneven distribution of multiple target items," *Quantum Information Processing,* vol. 17, no. 143, 2018.

6 Appendix

6.1 Grover's algorithm, geometrical interpretation

To better understand the action of the Grover iteration $G = (2|\psi\rangle\langle\psi| - I)U_f$, this appendix shows how the operator can be regarded as a rotation in a two-dimensional space. This space is spanned by the initial state and the superposition of all the solutions of the search problem. The initial state $|\psi\rangle$ is the balanced superposition of all states. Let us consider the two normalized states:

$$|\alpha\rangle = \frac{1}{\sqrt{N-M}} \sum_x{}'' |x\rangle$$

$$|\beta\rangle = \frac{1}{\sqrt{M}} \sum_x{}' |x\rangle$$

where the notation $\sum_x' |x\rangle$ indicates the sum over all states that are solutions of the problem while $\sum_x'' |x\rangle$ is the sum over all states that are not. The initial state $|\psi\rangle$ can be expressed as:

$$|\psi\rangle = \sqrt{\frac{N-M}{N}}|\alpha\rangle + \sqrt{\frac{M}{N}}|\beta\rangle. \qquad 6.1$$

Since the oracle U_f applies a phase factor of -1 to all states that are a solution, the application of U_f effectively generates a reflection with respect to the vector $|\alpha\rangle$ in the plane spanned by $|\alpha\rangle$ and $|\beta\rangle$:

$$U_f(a|\alpha\rangle + b|\beta\rangle) = a|\alpha\rangle - b|\beta\rangle.$$

In a similar way, the operator $2|\psi\rangle\langle\psi| - I$ causes a reflection on the same plane about the vector $|\psi\rangle$. The core concept behind this geometrical interpretation is that the two reflections combined, create a rotation. Notice that the state $G^k|\psi\rangle$ remains in the same plane during the whole computation for each k.
Let $\theta/2 = arcsin\sqrt{M/N}$, then the state $|\psi\rangle$ can be expressed as $|\psi\rangle = cos\frac{\theta}{2}|\alpha\rangle + sin\frac{\theta}{2}|\beta\rangle$.

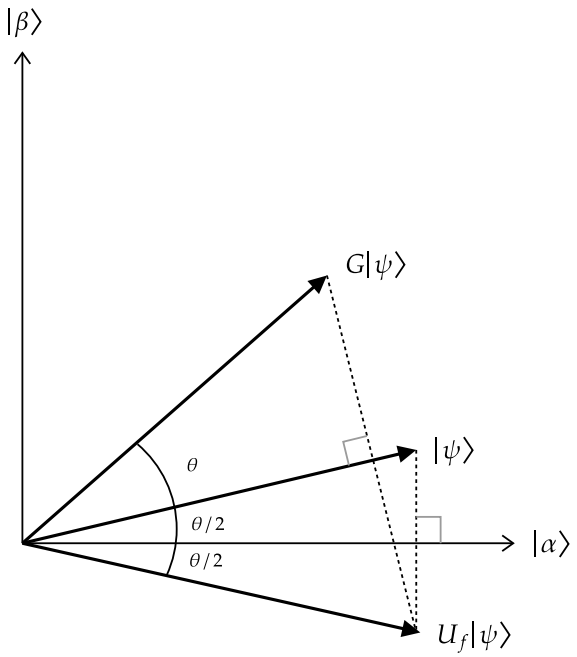

Figure 34: geometrical interpretation of the Grover iteration. The application of the Grover iteration G effectively applies a rotation. The oracle U_f reflects the state with respect to the state $|\alpha\rangle$ while the operation $2|\psi\rangle\langle\psi| - I$ is a reflection with respect to the initial state $|\psi\rangle$. After multiple applications of the Grover iteration, the state gets close to $|\beta\rangle$ at which point a measurement in the computational basis outputs a solution with high probability [2].

Figure 34 depicts the action of the Grover iteration. As can be seen, the two reflections generate a rotation on the space spanned by $|\alpha\rangle$ and $|\beta\rangle$. Since at each Grover iteration the state is rotated by θ, repeated applications of the operator G have the following effect on the state:

$$G^k|\psi\rangle = cos\left(\frac{2k+1}{2}\theta\right)|\alpha\rangle + sin\left(\frac{2k+1}{2}\theta\right)|\beta\rangle,$$

bringing the state $|\psi\rangle$ close to $|\beta\rangle$. To understand the computational complexity of the algorithm, the number of times the Grover iteration must be applied needs to be calculated. The probability of measuring a solution after k Grover iterations is $sin^2\left(\frac{2k+1}{2}\theta\right)$. To maximize the probability of measuring a solution, this value must be equal to $\frac{\pi}{2}$:

$$sin^2\left(\frac{2k+1}{2}\theta\right) = 1 \Longrightarrow \frac{2k+1}{2}\theta = \frac{\pi}{2} \Longrightarrow (2k+1)\theta = \pi.$$

Considering that the variable k is an integer, the computation proceeds as:

$$k = \left\lceil \frac{\pi}{2\theta} - \frac{1}{2} \right\rceil = \left\lceil \frac{\pi}{4\,arcsin\left(\sqrt{M/N}\right)} - \frac{1}{2} \right\rceil.$$

When $M \ll N$, $\theta \approx sin\,\theta \approx 2\sqrt{M/N}$:

$$k = \left\lceil \frac{\pi}{4\sqrt{M/N}} - \frac{1}{2} \right\rceil = \left\lceil \frac{\pi}{4}\sqrt{\frac{N}{M}} - \frac{1}{2} \right\rceil = O\left(\sqrt{\frac{N}{M}}\right),$$

hence the quadratic speedup of Grover's algorithm. In the case in which there is a single solution in the database, meaning $M = 1$, the complexity reduces to $R = O(\sqrt{N})$.

However, how to deal with the situation in which the number of solutions M is not known a priori?

If $M \geq N/2$ then the most efficient algorithm reduces to simply randomly picking an element and verifying whether it is a solution or not. If $M < N/2$ but it is not known, an efficient algorithm called BBHT [5] allows finding a solution by executing multiple times the Grover iteration and the measurement, each time probabilistically increasing the number of Grover iterations by a factor, keeping the complexity of the algorithm as $O\left(\sqrt{N/M}\right)$. Moreover, it is shown that Grover's algorithm cannot be improved to require much less than half the number of table lookups that it currently makes when a $1/2$ probability of success is desired [5]. This proves, in a sense, the optimality of the algorithm.

6.2 EQDR diffusion, state evolution

As explained in Section 2.3, the EQDR procedure is able to push the state of each qubit in the register towards a target state. This is done in order to exploit the genetic knowledge acquired during the runs of the algorithm. Let us consider a quantum register, composed of n qubits, in an arbitrary state. To push the state towards a predefined basis state, it is possible to apply the EQDR diffusion operator to the n qubits independently, each paired with an ancilla. The oracle used depends on whether the target value is 1 or 0. Since each qubit is paired with an ancilla, the total number of qubits becomes $2n$. Notice that, as explained in Section 2.4, the diffusion operator described below is not applied to all qubits deterministically. Instead, the diffusion operator is applied to each qubit based on a probability identified by the i-th element of the accuracy vector $\hat{a}$.

Adaptive diffusion recombination

The operator acts on a single qubit of the original state of the quantum register and on an ancilla qubit that is used to make the space two-dimensional. To enforce the target qubit to one of the two basis states $\{|0\rangle, |1\rangle\}$, it is possible to apply a single Grover iteration to the subspace spanned by the target qubit and the ancilla qubit. The effect of the quantum oracle is the same as always: it marks with a phase factor of -1 states that are solutions. In the case in which there is a single solution $|s\rangle$, the oracle can be defined as:

$$O = \mathbb{I} - 2|s\rangle\langle s|,$$

where $|s\rangle \in \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. The effect of the application of the oracle is the following:

$$O|\psi\rangle = \begin{cases} -|\psi\rangle, & \langle s|\psi\rangle = 1 \Rightarrow |s\rangle = |\psi\rangle \\ |\psi\rangle, & \langle s|\psi\rangle = 0 \Rightarrow |s\rangle \neq |\psi\rangle \end{cases}.$$

Each qubit of the register starts in an arbitrary state, while the ancilla is always initialized in the state $H|0\rangle = |+\rangle$:

$$|\psi_0\rangle = (\alpha|0\rangle + \beta|1\rangle) \otimes |+\rangle$$

$$|\psi_0\rangle = \frac{\alpha}{\sqrt{2}}|00\rangle + \frac{\alpha}{\sqrt{2}}|01\rangle + \frac{\beta}{\sqrt{2}}|10\rangle + \frac{\beta}{\sqrt{2}}|11\rangle.$$

For convenience, it is possible to define the following delta:

$$\delta_s^x = \begin{cases} 1, & x \neq s \\ -1, & x = s \end{cases}.$$

Therefore, the application of the oracle is carried out as:

$$O|\psi_0\rangle = \frac{1}{\sqrt{2}}(\delta_s^{00}\alpha|00\rangle + \delta_s^{01}\alpha|01\rangle + \delta_s^{10}\beta|10\rangle + \delta_s^{11}\beta|11\rangle) = |\psi_1\rangle.$$

Defining $|\phi\rangle$ as $|\phi\rangle = H|0\rangle \otimes H|0\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$, the application of the Grover diffusion operator proceeds as follows:

$$(2|\phi\rangle\langle\phi| - \mathbb{I})|\psi_1\rangle = 2|\phi\rangle\langle\phi|\psi_1\rangle - |\psi_1\rangle$$

$$= \frac{1}{\sqrt{2}}|\phi\rangle(\langle 00| + \langle 01| + \langle 10| + \langle 11|)(\delta_s^{00}\alpha|00\rangle + \delta_s^{01}\alpha|01\rangle + \delta_s^{10}\beta|10\rangle + \delta_s^{11}\beta|11\rangle) - |\psi_1\rangle$$

$$= \frac{1}{\sqrt{2}}(\delta_s^{00}\alpha + \delta_s^{01}\alpha + \delta_s^{10}\beta + \delta_s^{11}\beta)|\phi\rangle - |\psi_1\rangle$$

$$= \frac{1}{2\sqrt{2}}(\delta_s^{00}\alpha + \delta_s^{01}\alpha + \delta_s^{10}\beta + \delta_s^{11}\beta)(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$

$$-\frac{1}{\sqrt{2}}(\delta_s^{00}\alpha|00\rangle + \delta_s^{01}\alpha|01\rangle + \delta_s^{10}\beta|10\rangle + \delta_s^{11}\beta|11\rangle)$$

$$= \frac{1}{2\sqrt{2}}(-\delta_s^{00}\alpha + \delta_s^{01}\alpha + \delta_s^{10}\beta + \delta_s^{11}\beta)|00\rangle +$$

$$\frac{1}{2\sqrt{2}}(\delta_s^{00}\alpha - \delta_s^{01}\alpha + \delta_s^{10}\beta + \delta_s^{11}\beta)|01\rangle +$$

$$\frac{1}{2\sqrt{2}}(\delta_s^{00}\alpha + \delta_s^{01}\alpha - \delta_s^{10}\beta + \delta_s^{11}\beta)|10\rangle +$$

$$\frac{1}{2\sqrt{2}}(\delta_s^{00}\alpha + \delta_s^{01}\alpha + \delta_s^{10}\beta - \delta_s^{11}\beta)|11\rangle = |\psi'\rangle.$$

The state $|\psi'\rangle$ is the state after the application of a single Grover iteration. The register qubit can be forced towards $|1\rangle$ or $|0\rangle$. By convention, the oracle used to enforce the state $|0\rangle$ is $O = \mathbb{I} - 2|00\rangle\langle 00|$, while to enforce the state $|1\rangle$ the oracle used is $O = \mathbb{I} - 2|11\rangle\langle 11|$.
Due to this convention, $\delta_s^{10} = \delta_s^{01} = 1\ \forall s$ and the state $|\psi'\rangle$ can therefore be rewritten as:

$$|\psi'\rangle = \frac{1}{2\sqrt{2}}\big(\alpha(1 - \delta_s^{00}) + \beta(1 + \delta_s^{11})\big)|00\rangle + \frac{1}{2\sqrt{2}}\big(\alpha(\delta_s^{00} - 1) + \beta(1 + \delta_s^{11})\big)|01\rangle +$$

$$\frac{1}{2\sqrt{2}}\big(\alpha(\delta_s^{00} + 1) + \beta(\delta_s^{11} - 1)\big)|10\rangle + \frac{1}{2\sqrt{2}}\big(\alpha(\delta_s^{00} + 1) + \beta(1 - \delta_s^{11}\beta)\big)|11\rangle$$

$$|\psi'\rangle = \frac{1}{2\sqrt{2}}(\xi_{00}|00\rangle + \xi_{01}|01\rangle + \xi_{10}|10\rangle + \xi_{11}|11\rangle) \qquad 6.2$$

where the four ξ_{ab} are defined as follows:

$$\xi_{00} = \alpha(1 - \delta_s^{00}) + \beta(1 + \delta_s^{11}), \qquad \xi_{01} = \alpha(\delta_s^{00} - 1) + \beta(1 + \delta_s^{11}),$$

$$\xi_{10} = \alpha(\delta_s^{00} + 1) + \beta(\delta_s^{11} - 1), \qquad \xi_{11} = \alpha(\delta_s^{00} + 1) + \beta(1 - \delta_s^{11}\beta).$$

A single Grover iteration is enough to obtain the target state on the register qubit with probability $p = 1$, independently of α and β. Even if there are two states in the subspace that have an outcome probability different from zero, they both have the same state on the register qubit:

- If $O = \mathbb{I} - 2|00\rangle\langle 00| \Longrightarrow \delta_s^{00} = -1, \delta_s^{11} = 1$

$$\xi_{00} = 2\alpha + 2\beta, \qquad \xi_{01} = -2\alpha + 2\beta, \qquad \xi_{10} = \xi_{11} = 0$$

$$P(00) = \left|\frac{2\alpha + 2\beta}{2\sqrt{2}}\right|^2, \qquad P(01) = \left|\frac{-2\alpha + 2\beta}{2\sqrt{2}}\right|^2, \qquad P(10) = P(11) = 0$$

- If $O = \mathbb{I} - 2|11\rangle\langle 11| \Longrightarrow \delta_s^{00} = 1, \delta_s^{11} = -1$

$$\xi_{00} = \xi_{01} = 0, \qquad \xi_{10} = 2\alpha - 2\beta, \qquad \xi_{11} = 2\alpha + 2\beta$$

$$P(11) = \left|\frac{2\alpha + 2\beta}{2\sqrt{2}}\right|^2, \qquad P(10) = \left|\frac{2\alpha - 2\beta}{2\sqrt{2}}\right|^2, \qquad P(01) = P(00) = 0.$$

In the first case, the probability of measuring the state $|1\rangle$ in the first qubit, is equal to 0. Oppositely, in the second case the probability of obtaining the state $|0\rangle$ in the first qubit as an outcome of a measurement is equal to 0. To prove the correctness of the computation, notice that the two probabilities sum to one independently of the values of α and β:

$$\left|\frac{2\alpha + 2\beta}{2\sqrt{2}}\right|^2 + \left|\frac{\mp 2\alpha \pm 2\beta}{2\sqrt{2}}\right|^2$$

$$= \frac{1}{8}(4|\alpha + \beta|^2 + 4|\mp\alpha \pm \beta|^2)$$

$$= \frac{1}{8}(8|\alpha|^2 + 8|\beta|^2 + 4\alpha^*\beta + 4\beta^*\alpha - 4\alpha^*\beta - 4\beta^*\alpha) = |\alpha|^2 + |\beta|^2 = 1$$

The last equation comes from the fact that α and β are normalized by definition.
This property causes the first qubit measurement to be perfectly deterministic. To allow the exploration of the space, this is not optimal, as the population will converge toward the genome encoding the target bits. As explained in Section 2.4, the diffusion operator presented in this appendix is not deterministically applied to all qubits. The diffusion vector $\hat{b}$ is associated with an accuracy vector $\hat{a}$. The i-th element of the diffusion vector is enforced into the i-th qubit with probability $\hat{a}_i$. Naturally, the values $\hat{a}_i$ of the accuracy vector must reflect in some way the computation that has been used to obtain the diffusion vector from the genetic knowledge. Details on the computation of these two vectors are presented in Section 2.5.

Superposition mutation

The superposition mutation procedure stochastically applies rotational variational gates to each of the n qubits, based on the mutation rate η and the mutation amplitude m_a. The single qubit rotational gate is defined as:

$$R_y(\theta) = \begin{bmatrix} cos\left(\frac{\theta}{2}\right) & -sin\left(\frac{\theta}{2}\right) \\ sin\left(\frac{\theta}{2}\right) & cos\left(\frac{\theta}{2}\right) \end{bmatrix}.$$

After the adaptive diffusion recombination step, if the i-th element of the diffusion vector has been enforced into the i-th qubit, the state of the qubit will coincide with one of the two basis states $|0\rangle$ and $|1\rangle$. On the other

hand, the qubit state may has been left untouched. The computations below represent the effect of the application of the rotational gate in both situations.

Known qubit state

The following computation explains the effect of the application of the rotational variational gate in the case in which the qubit state coincides with one of the two basis states. The initial state of the qubit is presented in Equation (6.2). The application of the $R_y(\theta)$ gate to the single qubit state $|\psi'\rangle$ is computed as:

$$R_y(\theta)|\psi'\rangle =$$

$$\frac{\xi_{00}}{2\sqrt{2}}\left[cos\left(\frac{\theta}{2}\right)|0\rangle + sin\left(\frac{\theta}{2}\right)|1\rangle\right]\otimes|0\rangle + \frac{\xi_{01}}{2\sqrt{2}}\left[cos\left(\frac{\theta}{2}\right)|0\rangle + sin\left(\frac{\theta}{2}\right)|1\rangle\right]\otimes|1\rangle +$$

$$\frac{\xi_{10}}{2\sqrt{2}}\left[-sin\left(\frac{\theta}{2}\right)|0\rangle + cos\left(\frac{\theta}{2}\right)|1\rangle\right]\otimes|0\rangle + \frac{\xi_{11}}{2\sqrt{2}}\left[-sin\left(\frac{\theta}{2}\right)|0\rangle + cos\left(\frac{\theta}{2}\right)|1\rangle\right]\otimes|1\rangle$$

$$= cos\left(\frac{\theta}{2}\right)\frac{\xi_{00}}{2\sqrt{2}}|00\rangle + sin\left(\frac{\theta}{2}\right)\frac{\xi_{00}}{2\sqrt{2}}|10\rangle + cos\left(\frac{\theta}{2}\right)\frac{\xi_{01}}{2\sqrt{2}}|01\rangle + sin\left(\frac{\theta}{2}\right)\frac{\xi_{01}}{2\sqrt{2}}|11\rangle$$

$$-sin\left(\frac{\theta}{2}\right)\frac{\xi_{10}}{2\sqrt{2}}|00\rangle + cos\left(\frac{\theta}{2}\right)\frac{\xi_{10}}{2\sqrt{2}}|10\rangle - sin\left(\frac{\theta}{2}\right)\frac{\xi_{11}}{2\sqrt{2}}|01\rangle + cos\left(\frac{\theta}{2}\right)\frac{\xi_{11}}{2\sqrt{2}}|11\rangle$$

$$= \frac{1}{2\sqrt{2}}\left[\xi_{00}\,cos\left(\frac{\theta}{2}\right) - \xi_{10}\,sin\left(\frac{\theta}{2}\right)\right]|00\rangle + \frac{1}{2\sqrt{2}}\left[\xi_{01}\,cos\left(\frac{\theta}{2}\right) - \xi_{11}\,sin\left(\frac{\theta}{2}\right)\right]|01\rangle$$

$$+\frac{1}{2\sqrt{2}}\left[\xi_{00}\,sin\left(\frac{\theta}{2}\right) + \xi_{10}\,cos\left(\frac{\theta}{2}\right)\right]|10\rangle + \frac{1}{2\sqrt{2}}\left[\xi_{01}\,sin\left(\frac{\theta}{2}\right) + \xi_{11}\,cos\left(\frac{\theta}{2}\right)\right]|11\rangle$$

$$|\psi_f\rangle = \frac{1}{2\sqrt{2}}\left[\xi_{00}\,cos\left(\frac{\theta}{2}\right) - \xi_{10}\,sin\left(\frac{\theta}{2}\right)\right]|00\rangle + \frac{1}{2\sqrt{2}}\left[\xi_{01}\,cos\left(\frac{\theta}{2}\right) - \xi_{11}\,sin\left(\frac{\theta}{2}\right)\right]|01\rangle$$

$$+\frac{1}{2\sqrt{2}}\left[\xi_{00}\,sin\left(\frac{\theta}{2}\right) + \xi_{10}\,cos\left(\frac{\theta}{2}\right)\right]|10\rangle + \frac{1}{2\sqrt{2}}\left[\xi_{01}\,sin\left(\frac{\theta}{2}\right) + \xi_{11}\,cos\left(\frac{\theta}{2}\right)\right]|11\rangle. \quad 6.3$$

The state $|\psi_f\rangle$ identifies the final sate obtained after applying the rotational gate to a single qubit. The probabilities of the different outcomes are dependent on the quantum oracle used. This means that depending on the oracle used, the results are different:

- If $O = \mathbb{I} - 2|00\rangle\langle 00|$:

$$|\psi_f\rangle = \frac{1}{2\sqrt{2}}\left[\xi_{00}\,cos\left(\frac{\theta}{2}\right)|00\rangle + \xi_{01}\,cos\left(\frac{\theta}{2}\right)|01\rangle + \xi_{00}\,sin\left(\frac{\theta}{2}\right)|10\rangle + \xi_{01}\,sin\left(\frac{\theta}{2}\right)|11\rangle\right]$$

$$\xi_{00} = 2\alpha + 2\beta, \qquad \xi_{01} = -2\alpha + 2\beta, \qquad \xi_{10} = \xi_{11} = 0$$

$$P(00) = \left|cos\left(\frac{\theta}{2}\right)\frac{(2\alpha + 2\beta)}{2\sqrt{2}}\right|^2 = \frac{cos^2\left(\frac{\theta}{2}\right)}{2}|\alpha + \beta|^2$$

$$P(01) = \left|cos\left(\frac{\theta}{2}\right)\frac{(-2\alpha + 2\beta)}{2\sqrt{2}}\right|^2 = \frac{cos^2\left(\frac{\theta}{2}\right)}{2}|-\alpha + \beta|^2$$

$$P(10) = \left|sin\left(\frac{\theta}{2}\right)\frac{(2\alpha + 2\beta)}{2\sqrt{2}}\right|^2 = \frac{sin^2\left(\frac{\theta}{2}\right)}{2}|\alpha + \beta|^2$$

$$P(11) = \left|sin\left(\frac{\theta}{2}\right)\frac{(-2\alpha + 2\beta)}{2\sqrt{2}}\right|^2 = \frac{sin^2\left(\frac{\theta}{2}\right)}{2}|-\alpha + \beta|^2.$$

- If $O = \mathbb{I} - 2|11\rangle\langle 11|$:

$$|\psi_f\rangle = \frac{1}{2\sqrt{2}}\left[-\xi_{10}\, sin\left(\frac{\theta}{2}\right)|00\rangle - \xi_{11}\, sin\left(\frac{\theta}{2}\right)|01\rangle + \xi_{10}\, cos\left(\frac{\theta}{2}\right)|10\rangle + \xi_{11}\, cos\left(\frac{\theta}{2}\right)|11\rangle\right]$$

$$\xi_{11} = 2\alpha + 2\beta, \qquad \xi_{10} = 2\alpha - 2\beta, \qquad \xi_{00} = \xi_{01} = 0$$

$$P(00) = \left|sin\left(\frac{\theta}{2}\right)\frac{(-2\alpha + 2\beta)}{2\sqrt{2}}\right|^2 = \frac{sin^2\left(\frac{\theta}{2}\right)}{2}|-\alpha + \beta|^2$$

$$P(01) = \left|sin\left(\frac{\theta}{2}\right)\frac{(-2\alpha - 2\beta)}{2\sqrt{2}}\right|^2 = \frac{sin^2\left(\frac{\theta}{2}\right)}{2}|-\alpha - \beta|^2$$

$$P(10) = \left|cos\left(\frac{\theta}{2}\right)\frac{(2\alpha - 2\beta)}{2\sqrt{2}}\right|^2 = \frac{cos^2\left(\frac{\theta}{2}\right)}{2}|\alpha - \beta|^2$$

$$P(11) = \left|cos\left(\frac{\theta}{2}\right)\frac{(2\alpha + 2\beta)}{2\sqrt{2}}\right|^2 = \frac{cos^2\left(\frac{\theta}{2}\right)}{2}|\alpha + \beta|^2.$$

Notice how the probabilities of measuring $|0\rangle$ in the first case and $|1\rangle$ in the second case are not 0 anymore. The mutation step has introduced a certain degree of non-determinism that can be adjusted by simply tweaking the parameter θ. It is also possible to define different angles for different qubits, using different amplitudes of mutation depending on the qubit that is being mutated. This may result useful in situations in which different bits of the solutions, intended as alleles of the chromosome, may require diverse mutations amplitudes. Taking into consideration only the first qubit and leaving the ancilla aside, the probabilities of the various outcomes become:

- If $O = \mathbb{I} - 2|00\rangle\langle 00|$:

$$P(0) = \frac{cos^2\left(\frac{\theta}{2}\right)}{2}(|\alpha + \beta|^2 + |-\alpha + \beta|^2) = \frac{cos^2\left(\frac{\theta}{2}\right)}{2}(2|\alpha|^2 + 2|\beta|^2) = cos^2\left(\frac{\theta}{2}\right)(|\alpha|^2 + |\beta|^2)$$

$$P(1) = \frac{sin^2\left(\frac{\theta}{2}\right)}{2}(|\alpha+\beta|^2 + |-\alpha+\beta|^2) = \frac{sin^2\left(\frac{\theta}{2}\right)}{2}(2|\alpha|^2 + 2|\beta|^2) = sin^2\left(\frac{\theta}{2}\right)(|\alpha|^2 + |\beta|^2)$$

$$|\alpha|^2 + |\beta|^2 = 1$$

$$P(0) + P(1) = cos^2\left(\frac{\theta}{2}\right) + sin^2\left(\frac{\theta}{2}\right) = 1.$$

- If $O = \mathbb{I} - 2|11\rangle\langle 11|$:

$$P(0) = \frac{sin^2\left(\frac{\theta}{2}\right)}{2}(|-\alpha+\beta|^2 + |-\alpha-\beta|^2) = \frac{sin^2\left(\frac{\theta}{2}\right)}{2}(2|\alpha|^2 + 2|\beta|^2) = sin^2\left(\frac{\theta}{2}\right)(|\alpha|^2 + |\beta|^2)$$

$$P(1) = \frac{cos^2\left(\frac{\theta}{2}\right)}{2}(|\alpha-\beta|^2 + |\alpha+\beta|^2) = \frac{cos^2\left(\frac{\theta}{2}\right)}{2}(2|\alpha|^2 + 2|\beta|^2) = cos^2\left(\frac{\theta}{2}\right)(|\alpha|^2 + |\beta|^2)$$

$$|\alpha|^2 + |\beta|^2 = 1$$

$$P(0) + P(1) = sin^2\left(\frac{\theta}{2}\right) + cos^2\left(\frac{\theta}{2}\right) = 1.$$

The probabilities are the same as the ones obtained by applying a single rotation to one of the basis states. This is indeed what is happening: the diffusion recombination internally changes the state of the target qubit, acting on its amplitudes in order to deterministically make it coincide with one of the basis states. After this step, the mutation operator is a simple rotation that reintroduces a small stochasticity in the measurement process, in order to prevent measuring always the state represented by the diffusion vector $\hat{b}$.

Unknown qubit state

In the case in which the qubit's state has not been altered by the adaptive diffusion recombination, the initial state can be defined in its most generic form as:

$$|\psi_0\rangle = (\alpha|0\rangle + \beta|1\rangle)\otimes|a\rangle.$$

The rotational gate is applied only to the first qubit:

$$\left(R_y(\theta)\otimes\mathbb{I}\right)[(\alpha|0\rangle + \beta|1\rangle)\otimes|a\rangle] = R_y(\theta)(\alpha|0\rangle + \beta|1\rangle)\otimes\mathbb{I}|a\rangle$$

Again, the ancilla qubit is not important for the computation and can be safely ignored:

$$R_y(\theta)(\alpha|0\rangle + \beta|1\rangle) = \begin{bmatrix} cos\left(\frac{\theta}{2}\right) & -sin\left(\frac{\theta}{2}\right) \\ sin\left(\frac{\theta}{2}\right) & cos\left(\frac{\theta}{2}\right) \end{bmatrix}\begin{bmatrix}\alpha \\ \beta\end{bmatrix} = \begin{bmatrix} \alpha\, cos\left(\frac{\theta}{2}\right) - sin\left(\frac{\theta}{2}\right)\beta \\ \alpha\, sin\left(\frac{\theta}{2}\right) + cos\left(\frac{\theta}{2}\right)\beta \end{bmatrix}$$

$$R_y(\theta)(\alpha|0\rangle + \beta|1\rangle) = \left(\alpha\, cos\left(\frac{\theta}{2}\right) - sin\left(\frac{\theta}{2}\right)\beta\right)|0\rangle + \left(\alpha\, sin\left(\frac{\theta}{2}\right) + cos\left(\frac{\theta}{2}\right)\beta\right)|1\rangle$$

$$P(0) = \left|\alpha\, cos\left(\frac{\theta}{2}\right) - sin\left(\frac{\theta}{2}\right)\beta\right|^2$$

$$P(1) = \left|\alpha\, sin\left(\frac{\theta}{2}\right) + cos\left(\frac{\theta}{2}\right)\beta\right|^2.$$

The final probabilities depend on the values of α and β. This is an expected result as the superposition mutation, in this case, has a completely random effect due to the stochasticity of the value θ and the initial coefficients of the state. In the adaptive diffusion recombination, having an unknown qubit's state was a problem since the goal was to bias the qubit's state toward a certain basis state. In this situation, instead, the requirement is exactly to perturbate the state in a random way. For this reason, the uncertainty of the effect is fundamental for the evolution of the state.

An important remark is the fact that there is no measurement during the EQDR procedure. Both the diffusion recombination and the mutation step are applied to the superposition that is returned by Grover's algorithm. The measurement process takes place after both the adaptive diffusion recombination and the superposition mutation steps. The measured outcome is then evaluated with the classical fitness function and the genetic knowledge is updated accordingly. In the next iterations, the newly added genetic knowledge will contribute to the computation of the parameters that will drive the application of the next diffusion recombination.