UNIVERSITÀ DEGLI STUDI DI MILANO

FACOLTÀ DI SCIENZE E TECNOLOGIE
DIPARTIMENTO DI FISICA

Corso di Laurea triennale in Fisica (L-30)

Quantum walks with time-dependent Hamiltonians
and their application to the search problem on graphs

Relatore : **Prof. Matteo G.A. Paris**
Correlatore: **Prof. Stefano Olivares**
Correlatrice: **Dott.sa Claudia Benedetti**

Tesi di Laurea di:
Matteo Garbellini
Matricola 885615
Anno Accademico 2019/2020

Contents

Preliminaries

In this chapter we present the basic theoretical knowledge necessary to understand this thesis. We begin by introducing graph theory, random walks and quantum walks. We then discuss the quantum search problem firstly introduced by Grover and the quantum walks implementation for the unstructured search by Childs and Goldstone. Then we present the adiabatic theorem and its application to the adiabatic quantum search, studying the global and local adiabatic evolution. Lastly we look at the differences between the quantum walks approach and the local adiabatic evolution search by Roland and Cerf, which constitutes the basis from which our work is carried on.

1.1. Introduction to graph theory

A graph G is defined as a ordered pair (V, E), where V is a set of vertices and E is a set of edges,that represent the connection between any two pair of vertices. A vertex is usually indicated by the coursive letter v, and the corresponding edge connecting v to w is given by (v, w).

A graph can be characterized by many properties. Throughout our work we will only consider *simple graphs* characterized by being *undirected*, namely the edges E are symmetric, having no self loops such that $(v, v) \notin G$ and having no multiple equivalent edges. Additionally we require the graph to be *connected*, where each vertex can be reached by any other, following a path through the available edges.

It is then interesting to define the *vertex degree d_j*, that represents, given a vertex j, the number of edges that are incident to the vertex j (in the case of an

undirected graph the degree does not dependend on the edges incident from or to the selected vertex).

If indeed any two vertices (i, j) are connected by an edge we define them as *adjacent*, and from this we can construct an analytical representation of a graph, given by an N-dimensional square matrix called *adjacency matrix*, usually referred as A. The adjacency matrix is defined as following [1]:

$$A_{ij} = \begin{cases} 1 & \text{if } (i, j) \in G \\ 0 & \text{otherwise} \end{cases} \tag{1.1}$$

which represents the connectivity of the graph.

Furthermore, we can introduce a diagonal matrix D that encodes the informations of the vertex degrees for a particular graph G. Calling the N vertices of the graph as $j = 1, 2..N$ the matrix D is defined as:

$$D = diag(d_1, ..., d_N) \tag{1.2}$$

where d_j is the degree of the vertex j. In this particular context a natural operative basis arises, in which one can associate to each ordered vertex of the graph a vector of the standard basis of the N-dimensional vector space.

In order to study the dynamics of the system we introduce a the *Laplacian matrix L*, also known as the *discrete Laplacian operator*. It is defined as

$$L = D - A \tag{1.3}$$

where D is the diagonal degrees matrix and A is the adjacenty matrix. The discrete Laplacian operator is the analog of the continous Laplace operator on discrete domain, and for the finite, undirected, simple and connected graphs can charactarized by the following properties:

- L is symmetric given that both D and A are symmetric

---

[1]If we had to define the adjacency matrix for a general graph the value of A_{ij} is not necessarily equal to one, but a general a_{ij} since it takes into account the possibilities of self loops and multiple equivalent edges. For the simple graphs consider in this thesis the given definition suffices.

- the sum of all elements over a row/column equals to zero

- has a null eigenvalue which corresponds to the eigenvector $|\Psi_0\rangle = \frac{1}{\sqrt{N}}(1, 1, ..., 1)$

As previosuly mentioned we can associate every vertex to a basis vector of the N-dimensional vector space. Similarly we can expand this notation to an N-dimensional Hilbert space, where we can use the following Dirac notation to describe the vertex. This formalism allows us to write the Laplacian matrix in convenient quantum mechanical form using the projectors $|j\rangle\langle i|$

$$|1\rangle = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad |2\rangle = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \quad |N\rangle = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ N \end{bmatrix} \tag{1.4}$$

In the following paragraphs we give a brief description of the graphs that will be consider throughout the work.

1.1.1   Cycle Graph

A N-dimensional cycle graph $Cy(N)$ is a monodimensional structure with periodic boundary conditions $|N + 1\rangle = |N\rangle$. The Laplacian is given by

$$L = 2\sum_{k=1}^{N} |k\rangle\langle k| - \sum_{k=1}^{N-1} |k\rangle\langle k + 1| - \sum_{k=2}^{N} |k\rangle\langle k - 1| - |N\rangle\langle 1| - |1\rangle\langle N| \tag{1.5}$$

A pictorial representation of a cycle graph is given in Fig with the corrisponding laplacian matrix.

1.1.2   Complete Graph

A graph with N vertices is said to be complete if every node is adjacent to all the other N-1 nodes, thus representing a finite bidimensional structure. Its Laplacian matrix is given by:

$$L = (N - 1)\sum_{j=1}^{N} |j\rangle\langle j| - \sum_{k \neq j} |j\rangle\langle k| \tag{1.6}$$

A pictorial representation of a complete graph is given in figure. We will refere to this kind of graph with $C(N)$.

./figures/chapter1/cycle.png

./figures/chapter1/Laplacian.pdf

(a)   (b)
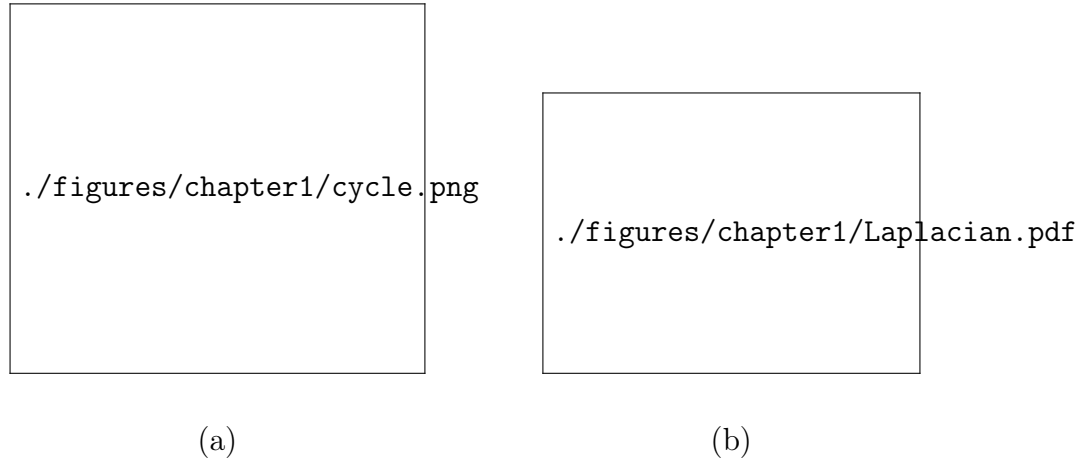
Figure 1.1: Pictorial representation of a cycle graph with 5 nodes (a), and the matricial representation for the Laplacian of Cy(5)

1.2. Quantum Walks

The continous time quantum walk (CTQW) is the direct analougue of the classical continous time random walk (CTRW). We begin by considering the classical one, expanding later on the quantum mechanical counterpart.

A continous-time random walk is a stochastic process by which a walker randomly moves on a particular mathematical structure, which in our scenario is a graph G. Let j be a note of a graph G and the initial node, such date the initial state of the system is $|j\rangle$. Then, we denote the transition probability of the walker to go from node j to a node k in a time t with $p_{k,j}(t)$. The state after time t is given by $|j;t\rangle$, such that the overlap with node k is $\langle k|j;t\rangle = p_{j,k}(t)$. The dynamics resulting in the state $|j;t\rangle$ follows from transition rates per unit time between two nodes. In particular these transition rates are the compo-
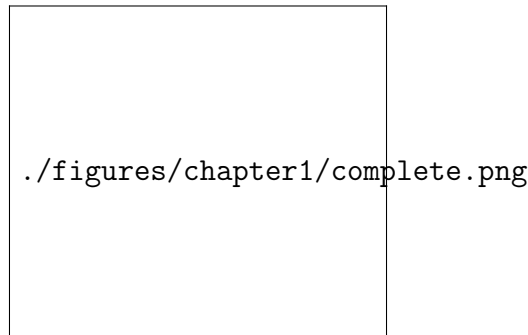
./figures/chapter1/complete.png

Figure 1.2: Pictorial representation of a complete graph with N=7

nents of the so-called *transfer matrix* T, namely $T_{ij} = \langle k|T|j \rangle$. If we assume a Markovian process, the following master equation defines the CTRW evolution:

$$\frac{d}{dt}p_{k,j}(t) = \sum_l T_{kl}p_{l,j}(t) \tag{1.7}$$

In the simplest case, where the transitions rates for all edges are equal, the transfer matrix is closely related to the Laplacian matrix through :

$$T = -\gamma L \tag{1.8}$$

where γ is the transition rate. The solution of eq. (1.7), along with the normalization constrains $\sum_{k=1}^{N} p_{k,j}(t) = 1 \ \forall t$, is given by

$$p_{ij}(t) = \langle k|e^{Tt}|j \rangle = \langle k|e^{-\gamma At}|j \rangle \tag{1.9}$$

Turning to quantum mechanics the evolution of any physical system obeys the Schroedinger equation, and QWs represent no exception. The dynamics of the CTQW is governed by a specific Hamiltonian H, such that the Schroedinger equation for the transition *amplitudes* $\alpha_{i,j}(t)$ is given by

$$\frac{d}{dt}\alpha_{i,j}(t) = -i \sum_l H_{k,l}\alpha_{l,j}(t) \tag{1.10}$$

where H is the Hamiltonian of the systemm, and for semplicity we assume $\hbar = 1$. The formal solution of such differential equation is similarly to the CTRW given by

$$\alpha_{l,j}(t) = \langle k|e^{-iHt}|j \rangle \tag{1.11}$$

where e^{-iHt} is the quantum mechanical time-evolution operator. We immediately notice the similar structure of equations (1.7) and (1.11), with the only difference apart from the imaginary unit that the first is a differential equation for transition probabilities while the latter allows to compute transition amplitudes. The similarity is further pushed bu Farhi and Gutmann in 1998 [1], when they proposed to identify the Hamiltonian H of the system with the negative of the classical transfer matrix T, which as we've seen previously is the Laplacian of the graph [2]:

$$H = -T = L \tag{1.12}$$

---

[2]Please note that in this particular scenario the transition amplitude γ is set to one.

Therefore the Laplacian matrix completely determines the evolution of the quantum system as well as the classical scenario.

1.3.   Grover's Quantum Search

In 1997 Lov K. Grover addressed the search problem through a quantum mechanical algorithm [4]. The search problem itself can be formulated in the following way: given an unsorted database containing N items, with only one satisfing a given condition, that one item has to be retrieved. Once an item is examined, it's possible to determine whether it represents the solution or not in just one step. Classically, the most efficient algorithm has to check each and every item in the database individually. If the item checked satisfies the condition the process stops, otherwise it will continue to examine the remaining items until the solution is find. On average, the algorithm will have to check $N/2$ items before finding the desired one.

On the other hand, Grover's quantum mechanical approach takes advantage of the *superposition* of states in a quantum system, and by having the input and output in such superposition can find the desired objects in $O(\sqrt{N})$ *quantum mechanical steps*, instead of $O(N)$ classical steps. As we shall see later, this quantum mechanicals steps consists of an elementary unitary operation.

Let's look at the algorithm to the quantum search problem more in detail, in particular setting the stage for the search algorithm in terms of an *oracle*, which plays a central role in Grover's search and throughout our work. Additionally, this allows us to present a very general description of the search procedure, and a geometric way to visualize its action [7].

Suppose we want to search through a search space of N elements, and let's consider the indices of such elements - instead of the actual elements - which clearly are numbers from 0 to $N-1$. In order to easily store the index in n bits we consider $N = 2^n$, and assume that the particular search problem has only one solution. We can now easily represent one instance of the search, i.e. checking if the item satisfies a particular condition, with a function $f(x)$, where x is the index integer ranging from 0 to $N-1$. The function is such that if x represents the solution to the search problem $f(x) = 1$, and $f(x) = 0$ if not a solution. In particular we suppose that we are supplied by a quantum *oracle O* with the

ability to *recognize* the solution of the search problem. The action of the oracle O is to *mark* the solution by applying a phase, namely $|x\rangle \xrightarrow{O} (-1)^{f(x)}|x\rangle$.

In a simplified manner, the quantum search algorithm proposed by Grover consists of reapeted application of a quantum subroutine known as the *Grover iteration* or *Grover operator G* defined as:

$$G = (2|\psi\rangle\langle\psi| - I)O \tag{1.13}$$

The questions that arises is what does this Grover operator do. We can show that it can be regarded as a *rotation* in the 2-dimensional space spanned by the beginning vector $|\psi\rangle$ and the vector $|w\rangle$ representing the solution to the search problem. We can then define the normalized state $|\alpha\rangle$ and $|\beta\rangle$:

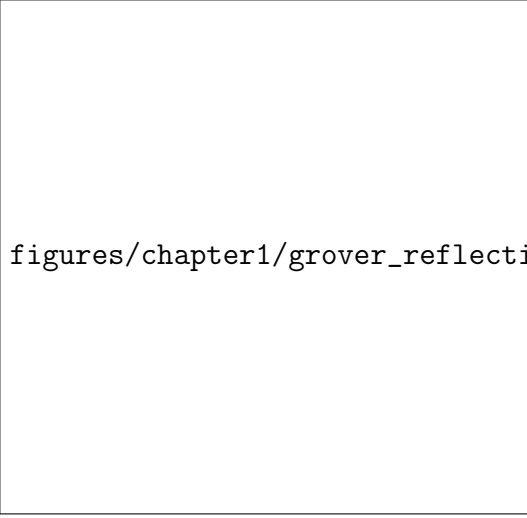$$|\alpha\rangle = \frac{1}{\sqrt{N-1}} \sum_{x \neq w} |x\rangle \tag{1.14}$$

$$|\beta\rangle = |w\rangle \tag{1.15}$$

representing respectively, the superposition of all states that are not a solution and the state of the solution. Therefore the initial state $|\psi\rangle$ can be re-expressed by:

$$|\psi\rangle = \sqrt{\frac{N-1}{N}}|\alpha\rangle + \sqrt{\frac{1}{N}}|\beta\rangle \tag{1.16}$$

The effect of G can be understood by realizing that the oracle O performs a *reflection* of $|\psi\rangle$ about the vector $|\alpha\rangle$ in the plane defined by $|\alpha\rangle$ and $|\beta\rangle$. Then, $2|\psi\rangle\langle\psi| - I$ performs another reflection about the vector $|\psi\rangle$. As can be seen in figure?, the product of these two reflections i a rotation. By applying the grover operator k times we can rotate the initial state vector $|\psi\rangle$ close to $|\beta\rangle$, that is, the solution to the search problem. The number of iterations necessary - for which the derivation is not of our interest in the scope of this thesis - is given by $\frac{\pi}{4}\sqrt{N}$.

To summerize, G is the rotation in the 2-dimensional space spanned by $|\alpha\rangle$ and $|\beta\rangle$. Repeated application of the Grover operator rotate the initial state vector $|\psi\rangle$ close to $|\beta\rangle$. For applications in the order of $O(\sqrt{N})$, a measurement of the state in the computational basis produces with high probability - very close to 1 - $|\beta\rangle$ as the outcome, that is, the solution to the search problem.

Figure 1.3: **Action of the Grover operator G**: The figure shows the action of the Grover operator $G = (2|\psi\rangle\langle\psi| - I)O$. Starting from the initial state $|\psi\rangle$ the action of the Oracle O reflects it about the vector $|\alpha\rangle$. Then, the action of $(2|\psi\rangle\langle\psi| - I)$ is an additional reflection about $|\psi\rangle$. The application of the Grover operator for $\pi/4\sqrt{N}$ times rotates the inital state bringing it close to $|\beta\rangle$, that is, the solution of the search problem.

1.4.   Search by Quantum Walks

The search problem first introduced by Grover can be reformulated in terms of search based on a continous-time quantum walk on a graph [2].

In order to do so we need to modify the quantum walk hamiltonian of eq. (1.12) so that the vertex $|w\rangle$ is somewhat special. Therefore, an *oracle hamiltonian* is introduced:

$$H_w = -|w\rangle\langle w| \tag{1.17}$$

that has energy zero for all but the vertex $|w\rangle$ for which it has enenergy -1. Therefore the Grover problem becomes finding the ground state of this hamiltonian.

To implement the search we consider the Hamiltonian

$$H = \gamma L + H_w = \gamma L - |w\rangle\langle w| \tag{1.18}$$

where L is the laplacian of the graph G, that as we've seen in section? governs the evolution of the quantum walk.

The quantum search routine works as following:

- we consider the balanced superposition of all possible states, namely

$$|s\rangle = \frac{1}{\sqrt{N}} \sum_j |j\rangle \qquad (1.19)$$

- we run the quantum walk for a time t_f and find the corrisponding evolved state using the hamiltonian H

$$|\psi(t_f)\rangle = U(t_f)|s\rangle = \exp\left\{-\frac{i}{\hbar}Ht_f\right\}|s\rangle \qquad (1.20)$$

(Note that this evolution is valid only for time-independent hamiltonians.)

- we then measure the state onto the target $|w\rangle$ and find the corrisponding probability

$$p = |\langle w|\psi(t_f)\rangle|^2 \qquad (1.21)$$

The objective is to find the optimal value of γ so that the success probability $|\langle w|\psi(t_f)\rangle|^2$ is as close as possible to 1 for the smallest t_f.

We can turn our attention on trying to understand why we should expect this algorithm to give a success probability for some values of γ, t_f. To motivate this we need to frame the problem in terms of Hamiltonian spectrum. In particular we look at the two extremes, namely $\gamma \to \infty$ and $\gamma \to 0$.

- as $\gamma \to \infty$ the contribution of H_w is somewhat negligible, to the point that the ground state of H is $|s\rangle$[3]

- on the other hand, if $\gamma \to 0$ the contribution of the Laplacian to the overall Hamiltonian H disappears and thus the ground state of H is close to $|w\rangle$

We expect that for some intermediate γ the ground state will transition from $|w\rangle$ to $|s\rangle$ and thus could have substantial overlap on both for a certain value of γ. Additionally, if the first excited states have substantial overlap at such values of γ, then the Hamiltonian will drive transitions between two states, thus rotating the state from $|s\rangle$ to one with substantial overlap with $|w\rangle$, giving the success probability that we previously talked about. In particular this transition will happen in a time that dependes on the Hamiltonian eigenvalues separation between the first and ground state, namely $1/(E_1 - E_0)$.

---

[3]Note that regardless of the graph topology considered, $|s\rangle$ is the ground state of the Laplacian, with $L|s\rangle = 0$.

This is a good description of the algorithm if the dimension of the graph is sufficiently high. However, if we consider a d-dimensional lattice with d independent of N we still see that for a critical value of γ the state switches from the ground state to the first excited state, but for $d < 4$ the $|w\rangle$ state does not have substantial overlap on the ground and fist excited state, thus the algorithm does not work.

Comments on the oracle H_w: The oracle introduce in this fashion is exactly the continous-time version of the reflection R_w in Grover's algorithm [10] because by itself it only evolves the marked vertex $|w\rangle$ by a phase

$$e^{-i|w\rangle\langle w|t}|w\rangle = e^{-it}|w\rangle \tag{1.22}$$

while leaving the other vertices unchanced, thus making this the continous-time version of a yes/no oracle.

1.4.1   Search on Complete Graph

We now look at the complete graph which can be thought of having dimension proportional to N and represents the simples example of the quantum walk search application [2].

We begin by noticing that adding a multiple of the identity matrix to the Laplacian only contributes a global unobservable phase. We add $-NI$ so that:

$$L - NI = N|s\rangle\langle s| \tag{1.23}$$

This gives us the following Hamiltonian:

$$H = -\gamma N|s\rangle\langle s| - |w\rangle\langle w| \tag{1.24}$$

In particular we know that this hamiltonian acts non-trivially in a two dimensional subspace spanned by $|s\rangle$ and $|w\rangle$, thus making it stright-forward to compute its spectrum. If we use the $\{|w\rangle, |r\rangle\}$ basis, the hamiltonian can be expressed in the following way:

$$H = \frac{-1}{N}\begin{pmatrix} N+1 & \sqrt{N-1} \\ \sqrt{N-1} & N-1 \end{pmatrix} \tag{1.25}$$

Applying the time-evolution operator to the initial state $|s\rangle$, the system at time t will be

$$|\psi(t)\rangle = e^{it} \begin{pmatrix} \frac{1}{\sqrt{N}} \cos(\frac{t}{\sqrt{N}}) + i \sin(\frac{t}{\sqrt{N}}) \\ \sqrt{\frac{N-1}{N}} \cos(\frac{t}{\sqrt{N}}) \end{pmatrix} \tag{1.26}$$

and for $t = \pi\sqrt{N}/2$ the system reaches a success probability of 1, namely

$$p = \left| \langle w|\psi(t)\rangle \right|^2 \Big|_{t=\pi\sqrt{N}/2} = \left| \langle w|ie^{it}w\rangle \right|^2 = 1 \tag{1.27}$$

recalling that in the 2-dimensional subspace $w = (0,1)$. Thus the walk rotates the state from $|s\rangle$ to $|w\rangle$ in $O(\sqrt{N})$.

1.5. Search by Adiabatic Evolution

We now address the computation by adiabatic evolution firstly introduced by Farhi et al., that takes advantage of the adiabatic theorem to find the solution of a computational problem [3]. We begin by looking at the *adiabatic theorem* and its implication. Then, an overview of the adiabatic implementation of the algorithm for solving the unstructured search problem is given which has a time scaling of $O(N)$. Lastly, we see how applying the adiabatic theorem *locally* can lead to a time of order $O(\sqrt{N})$ which is optimal.

1.5.1 Adiabatic Theorem

A quantum system evolves according to the Schroedinger equation

$$i\frac{d}{dt}|\psi(t)\rangle = H(t)|\psi(t)\rangle \tag{1.28}$$

and defining the instantaneous eigenstates and eigenvalues of H(t) by

$$H|\psi_l(t)\rangle = E_l|\psi_l(t)\rangle \tag{1.29}$$

such that $E_0(t) \geq E_1(t) \geq ... \geq E_{N-1}(t)$.

The adiabatic theorem states that if the gap between the two lowest energy levels, $E_1(t) - E_0(t) > 0$, is stritcly greater than zero then for $T \to \infty$ the probability of being in the ground state is equal to one, namely

$$\lim_{T\to\infty} \langle\psi_0(T)|\psi(T)\rangle = 1 \tag{1.30}$$

This means that if the system is chosen to evolve at a slow enough rate, the instantaneous hamiltonian will remain in the ground state throughout the evolution. Furthermore it's convenient to parametrize the Hamiltonian as $H(s)$, where $s = t/T$, with $t \in [0, T]$ so that $s \in [0, 1]$. Let's now define the energy minimum gap between the lowest eigenvalues of H by

$$g_{min} = \min_{0 \leq s \leq 1} (E_1(s) - E_0(s)) \tag{1.31}$$

This allows to find a minimum time T^* such that, for $T \gg T^*$ the probability of being in the ground state is arbitrarily close to 1. In details, T^* is given by:

$$T^* = \frac{\varepsilon}{g_{min}^2} \tag{1.32}$$

where

$$\varepsilon = \max_{0 \leq s \leq 1} \left| \left\langle l = 1; s \left| \frac{dH(s)}{dt} \right| l = 0; s \right\rangle \right| \tag{1.33}$$

Let's now discuss how to take advantage of the adiabatic theorem. It is often presented a problem hamiltonian H_P whose ground state is not so straight forward to find; on the other hand we can prepare the system in a beginning hamiltonian H_B whose ground state is known. The problem hamiltonian encodes the solution of the problem, while the beginning hamiltonian is a tool for easily preparing the state to be evolved. The adiabatic evolution then consists, assuming that the ground state of H_P is unique, in having a time dependent hamiltonian $H(s)$ such that

$$H(s) = (1 - s)H_B + sH_P \tag{1.34}$$

In this way we can prepare for $s = 0$ the system in H_B and let it evolve so that for $s = 1$ it reaches H_P. Therefore, if the system is made to evolve sufficiently slowly we will find ourself in the ground state of the problem hamiltonian, that is exactly the solution.

1.5.2   Global adiabatic evolution

Let's now apply what just seen to the unsorted search problem [8]. As done in Section 1.3 we consider an unsorted database of N elements such that $N = 2^n$, so that the elements in this particular basis can be written as $|i\rangle$, with $i = 0, ..., N - 1$. The marked state can be then denoted as $|w\rangle$, and we begin by

considering the initial state as the superposition of all the elements $|i\rangle$:

$$|\psi_0\rangle = \frac{1}{\sqrt{N}} \sum_{i=1}^{N-1} |i\rangle \tag{1.35}$$

With this in mind we design two particular Hamiltonians H_0 and H_w such that $|\psi_0\rangle$ is the ground state of the first while $|w\rangle$ is the ground state of the latter:

$$H_0 = I - |\psi_0\rangle\langle\psi_0| \tag{1.36}$$

$$H_w = I - |w\rangle\langle w| \tag{1.37}$$

The time-dependent Hamiltonian follows from the adiabatic implementation discussed in Section 1.5, where H_0 is the beginning hamiltonian while H_w is the problem hamiltonian. It thus consists in a linear interpolation between H_0 and H_w:

$$H(t) = (1 - s)H_0 + sH_w \tag{1.38}$$

where $s = t/T$ is the linear interpolating schedule.

The search routine runs as usual, beginning with preparing the system in the state $|\psi(0)\rangle = |\psi_0\rangle$ and then applying the Hamiltonian $H(t)$ for a time T. We're interested in finding the time-dependent condition such that the system evolves sufficiently slowly, allowing us to find the solution $|w\rangle$ with high probability. First, we determine the quantity $\langle\frac{dH}{dt}\rangle$:

$$\left\langle\frac{dH}{dt}\right\rangle_{0,1} = \frac{ds}{dt}\left\langle\frac{dH}{ds}\right\rangle_{0,1} = \frac{1}{T}\left\langle\frac{dH}{ds}\right\rangle_{0,1} \tag{1.39}$$

We then find the eigenvalues of the Hamiltonian and determine the separation g between the ground state and first excited one, namely E_1 and E_0, as a function of the interpolating schedule s:

$$g = \sqrt{1 - 4\frac{N-1}{N}s(1-s)} \tag{1.40}$$

We're interested in the minimum gap $g_{\min}$ which is found for $s = 1/2$. Additionally we find that $\left|\langle\frac{dH}{ds}\rangle_{0,1}\right| \leq 1$, therefore Equation (1.32) becomes:

$$\frac{\left|\langle\frac{dH}{ds}\rangle_{0,1}\right|}{g_{\min}^2} \leq \varepsilon \tag{1.41}$$

so that the adiabatic condition is verified, and thus the hamiltonian stays in the ground state at all times, provided that :

$$T \geq N/\varepsilon \tag{1.42}$$

Therefore the computation time is of order N, showing no speed up compared to the classical search.

1.5.3   Local adiabatic evolution

Roland and Cerf showed that the adiabatic evolution can be improved by applying the adiabatic condition of eq. (1.41) locally instead of globally [8]. If we look at the plot of the separation g we see that the adiabatic condition is critical only for $s = 1/2$ where it reaches its minimum. At the beginning and at the end of
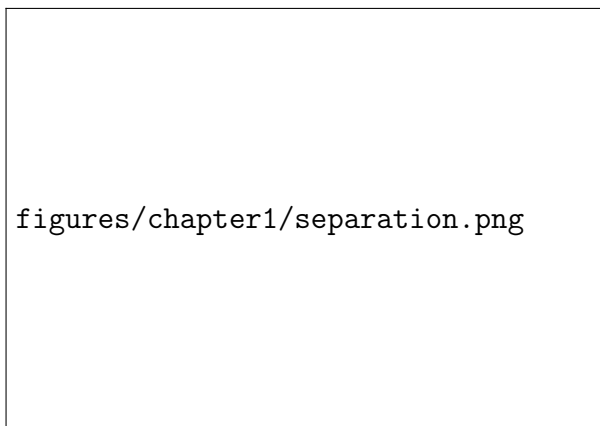

Figure 1.4: Eigenvalue separation of the time-dependent Hamiltonian $H(s)$ as a function of the reduced time s, for N=64. *Figure from Roland and Cerf* [8]

the evolution the separation is large enough so that the adiabatic condition and thus the time necessary for the system to evolve adiabatically is small, allowing for better time scaling. The improvement comes in the form of the interpolating schedule, following the idea that it should be steeper for large separation g and flatter (more than linear) for small separation around $s = 1/2$. Let's see how can this be done.

We divide the interval $[0, t_f]$ into infinitesimal intervals dt and adapt the evolution rate $\frac{ds}{dt}$ to the local adabatic condition. In this way we can find the optimal $s(t)$, with boundary conditions $s(0) = 0$ and $s(t_f) = 1$. We find the new adiabatic

condition for all time t:

$$\left|\frac{ds}{dt}\right| \leq \varepsilon \frac{g^2(t)}{\left|\langle\frac{dH}{ds}\rangle_{0,1}\right|} \tag{1.43}$$

Using eq. (1.40) and $\left|\langle\frac{dH}{ds}\rangle_{0,1}\right| \leq 1$ and Hamiltonian is chosen to evolve with the interpolating schedule that is solution of the following differential equation (with $\varepsilon << 1$):

$$\frac{ds}{dt} = \varepsilon g^2(t) = \varepsilon\left[1 - 4\frac{N-1}{N}s(1-s)\right] \tag{1.44}$$

After integration we get the following

$$t = \frac{1}{2\varepsilon}\frac{N}{\sqrt{N-1}}\left[\arctan\left(\sqrt{N-1}(2s-1)\right) + \arctan\sqrt{N-1}\right] \tag{1.45}$$

By inverting we find the interpolating schedule $s(t)$ as can be seen in fig. 1.5. In order to determine the computation time of this algorithm. To do so we evaluate $s = 1$ and in the approximation that $N >> 1$ we get:

$$T = \frac{\pi}{2\varepsilon}\sqrt{N} \tag{1.46}$$

which represents a great improvement on Equation (1.42) Indeed we have a quadratic speed up compared to the global adiabatic evolution, and thus this algorithm can be seen as the adiabatic implementation of the Grover's search algorithm.
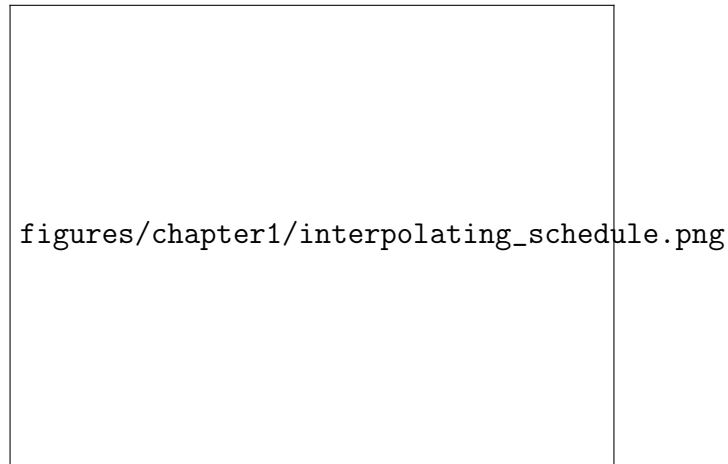

figures/chapter1/interpolating_schedule.png

Figure 1.5: Roland and Cerf interpolating schedule for the unstructured search with N=64. *Figure from Roland and Cerf* [8]

1.6. Impossibility of a search algoritm based on adiabatic quantum walks

In this section we show that an *adiabatic quantum walks* (AQW) based search algorithm cannot be implemented with the usual Grover's oracle and requires a more sophisticated structure. This section, based on a paper by Wong et al. [10], sets the basis on which our work builds upon.

In the previous sections we discussed Grover's original discrete-time search algorithm (GR), Farhi and Gutmann's quantum walk analogue (FG) and Roland and Cerf's adiabatic analogue (RC). Throught the discussion we analysized the systems in a 2-dimensional subspace spanned by $\{|w\rangle, |r\rangle\}$. This allows to visualize the evolution of the quantum algorithm on the two dimensional Bloch sphere and compare the path taken by each different approach. In particular we notice that the original Grover's algorithm and the Roland and Cerf's local adiabatic evolution follow the same path on the xz-plane since they both always have real coefficients, as can be seen in Figures 1.6a and 1.6b respectively. On the other hand the quantum walk formulation of Farhi and Gutmann has an unmeasurable complex phase thus it evolves on a different path on the yz-plane, see Figure 1.6c.

figures/chapter1/blochsphere_Grover.pdf
figures/chapter1/blochsphere_RC.pdf
figures/chapter1/blochsphere_FG.pdf

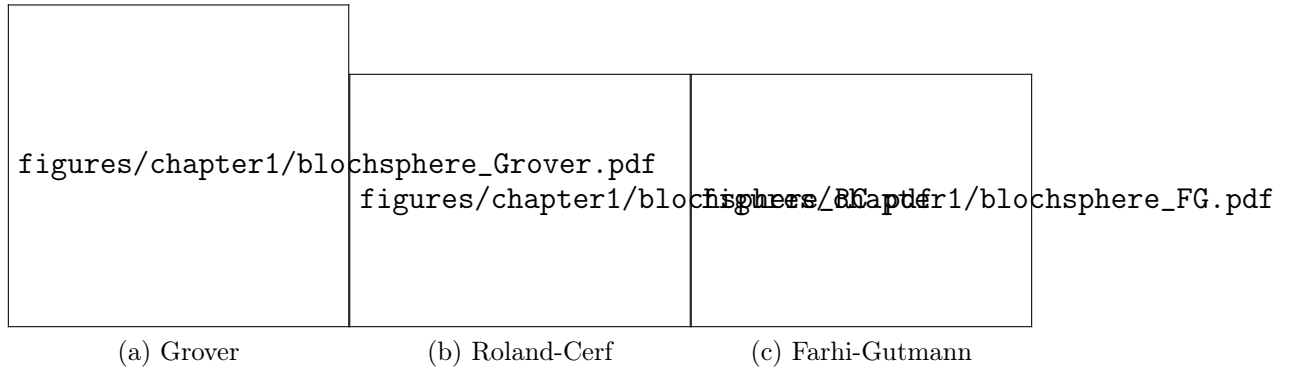(a) Grover   (b) Roland-Cerf   (c) Farhi-Gutmann

Figure 1.6: **Evolution of the quantum algorithms on the bloch sphere**. The figure shows the evolution of quantum algorithm on the Bloch sphere with marked vertex $|w\rangle$ at the North Pole and the equal superposition of the unmarked vertices $|r\rangle$ at the South Pole, and $N = 1024$: (a) Grover's original discrete-time search algorithm, (b) Roland and Cerf's local adiabatic evolution analogue and (c) Farhi and Gutmann's quantum walk analogue. *Wong et al. (2016)* [10]

This substantial difference of the path taked by the different algorithms is exacly what Wong et al. investigate, in particular with the goal of determine which

adiabatic algorithm does follow the same path of the quantum walk algorithm of Farhi and Gutmann in order to implement an adiabatic quantum walk based search algorithm.

To do so they constructed an hamiltonian starting from the following states, with the first represents the ground state and the latter the first (and only) excited state - rememeber that the Grover's search rotates the initial state over the final state, thus the first excited state is the orthogonal of the ground state.

$$|\psi_0(t)\rangle = \alpha(t)|w\rangle + \beta(t)|r\rangle \tag{1.47}$$

$$|\psi_1(t)\rangle = \beta(t)|w\rangle - \alpha^*(t)|r\rangle \tag{1.48}$$

where the coefficients are given by the components of $|\psi_0(t)\rangle$ of eq. (1.26)[4]. The hamiltonian is then given by the linear combination of $|\psi_0(t)\rangle$ and $|\psi_1(t)\rangle$:

$$H(t) = \lambda_0|\psi_0\rangle\langle\psi_0| + \lambda_1|\psi_1\rangle\langle\psi_1| \tag{1.49}$$

which allows to determine the final time-dependent *adiabatic* hamiltonian - where the time-dependence is given by the interpolating schedule $s(t)$:

$$H(s) = \sqrt[4]{\frac{s(1-s)}{4\varepsilon^2 N}}\left[(1-s)H_0 + sH_f + \sqrt{s(1-s)}H_e\right] \tag{1.50}$$

where the interpolating schedule is given by $s(t) = \sin^2\left(\frac{t}{\sqrt{N}}\right)$. Indeed the system evolves from $t = 0$ to $t = \frac{\pi}{2}\sqrt{N}$ as we would expect. Looking closely the Hamiltonian H_0 is the beginning hamiltonian, H_f is the final hamiltonian - encoding the solution of the search problem - and H_e is an *extra* hamiltonian, a common tecnique for manupulating the evolution of the path of adiabatic algorithms[5]. Let's now look at the consequences of this Hamiltonian, looking at H_0, H_f and H_e individually.

H_0 has ground state $|s\rangle$ and excited state $|s^\perp\rangle$ with respective eigenvalues -1 and $+1$, thus the hamiltonian can be written as:

$$H_0 = |s^\perp\rangle\langle s^\perp| - |s\rangle\langle s| \tag{1.51}$$

---

[4]Note that the global phase e^{it} is dropped.

[5]The extra hamiltonian commonly appears as $s(s-1)$, while the one considered in this scenario is given by the square root of such value [10].

Similarly, the final Hamiltonian H_f has ground state $|w\rangle$ and excited state $|r\rangle$ with eigenvalues $+1$ and -1:

$$H_f = |r\rangle\langle r| - |w\rangle\langle w| \tag{1.52}$$

It's immediate to notice that these two hamiltonian look fairly similar to the initial and final hamiltonian of the standard adiabatic quantum search algorithm of Equations (1.36) and (1.37). Although the similarities look at first sight promising, the extra Hamiltonian H_e and the interpolating schedule $s(t)$ change completely the evolution of the system. In particular, if we look at the Hamiltonian H_e - given by:

$$H_e = 2i\sqrt{\frac{N-1}{N}}\big(|r\rangle\langle w| - |w\rangle\langle r|\big) \tag{1.53}$$

we see that rather than having the standard yes/no oracle given by $|w\rangle\langle w|$, this extra hamiltonian introduces a more powerful structure, driving the evolution between $|w\rangle$ and $|r\rangle$ insted of just applying a phase to the targer state $|w\rangle$ like in the standard Grover's algorithm. This is a confirmation that the oracle is no more the usual Grover's yes/no oracle, and althought this a proof that an adiabatic evolution following the quantum walk path does exists it abandons the usual notion of the oracle and does not solve the search problem itself as originally posed by Grover.

The difference between the Quantum Walks and the Adiabatic Evolution approach is not a measure of the performance of the algorithms, since both, individually, solve the Grover's problem in $O(\sqrt{N})$ time, but illustrate that the "*how* they compute is different, even though *what* they compute is the same" [10].

Althought the worke by Wong et al. showed that is not possible to implement an adiabatic quantum walk algorithm that solves the search problem, it leaves space to a time-dependent algorithm *inspired* by the adiabatic evolution but that it's free from the constrains of the adiabatic condition. From this last considerations we develop our thesis. In the next chapter we summarize the work done and gives some interesting insights and properties of the quantum walks search algorithm with time-dependent hamiltonians.

Quantum walks with time-dependent Hamiltonians and their application to the search problem on graphs

The main goal of this thesis is to study quantum walks with time dependent Hamiltonians, focusing in particular on their application to the spatial search problem on graph. The general idea is trying to improve a time-independent implementation of quantum walks search using a time-dependent Hamiltonian analogous to the one used in the adiabatic evolution. In order to determine whether this new approach produces successfull results we study two selected graph topologies: the circular graph, for which the time-independent approach is not able to solve the search problem, and the complete graph for which the search problem is solved for both the time-independent and adiabatic evolution approaches.

We compare the two methods for the *optimized-search*, *localization* - which represents a search without needs to optimize the time - and a measure of *robustness*.

2.1. Search with time-dependent Hamiltonian

In Section 1.4 we discussed the use of quantum walks for the spatial search problem [2] and the application to the complete graph where the search problem is solved. We then illustrated how the adiabatic theorem can be used to solve computational problems with an adiabatic evolution of a quantum system [3]. Efforts to combine the two approaches showed that the search problem can be solved only with structures stronger than the usual Grover's oracle [10], thus making an *Adiabatic-Quantum-Walk-Search* impossible. However this leaves

space to a merely time-dependent quantum walk search, that takes advantage of a time-dependent implementation similar to the adiabatic evolution but that is not bounded to the strict adiabatic-theorem conditions and the limitation of the standard Grover's oracle.

2.1.1   Time-dependent quantum walks

Following from the adiabatic evolution discussed in the preliminaries, we consider a search Hamiltonian that interpolates between an initial Hamiltonian, i.e. the Laplacian of the graph L, and the final oracle Hamiltonian $H_w = \gamma|w\rangle\langle w|$

$$H(s) = (1-s)L - s\gamma|w\rangle\langle w| \tag{2.1}$$

where the interpolation schedule $s = s(t)$ goes from 0 to 1 as the time t goes from 0 to the runtime T.

In order to find the evolved ground state of the beginning Hamiltonian we need to determine the evolution operator that for a time-dependent Hamiltonian is given by:

$$S(t, t_0) = \text{T} \exp\left\{\frac{1}{i}\int_{t_0}^{t} dt' H(t')\right\} \tag{2.2}$$

where T is the time-ordering operator. Since we are only interested in the evolved state, having the exact evolution operator is somewhat irrelevant. Additionally for the circular graph - which represents the main topology studied in this thesis - the search Hamiltonian does not show any particular properties making an analitical derivation of the operator not possible. We therefore proceed by solving the differential Schroedinger equation:

$$i\frac{d}{dt}|\psi(t)\rangle = H|\psi(t)\rangle \tag{2.3}$$

Recalling that we are dealing with matrices and vectors in an N-dimensional Hilbert space, we solve N-differential equations of the form

$$\frac{d}{dt}|\psi_i(t)\rangle = \sum_j H_{ij}|\psi_i(t)\rangle \tag{2.4}$$

with the boundary conditions $|\psi_i(0)\rangle = |\psi_i\rangle$.

2.1.2 Interpolating schedule s(t)

As pointed by Wong the interpolating schedule s(t) plays a crucial role in the evolution of the system and in the overall scaling of the algorithm. The original adiabatic evolution by Farhi and Gutmann [3] uses a linear interpolating schedule defined as $s_L(t) = \equiv \frac{t}{T}$. Roland and Cerf show that in order to obtain a quadratic speedup for the complete graph a non linear schedule is essential [8][5].

Thus, from a linear interpolating schedule we consider also quadratic and cubic schedules:

$$s_S(t) \equiv \sqrt{\frac{t}{T}} \qquad s_C(t) \equiv \sqrt[3]{\frac{t}{T}} \qquad\qquad (2.5)$$

We then consider the interpolating schedule analitically derived by Roland and Cerf for the unstructured search (complete graph) given by :

$$t = \frac{1}{2\epsilon} \frac{N}{\sqrt{N-1}} \left[\arctan\big(\sqrt{N-1}(2s-1)\big) + \arctan\sqrt{N-1} \right] \qquad (2.6)$$

By inverting the function, $s(t)$ is obtained as plotted in fig. 2.1(a)
As we have already mentioned in Section 1.5.3 the shape of this interpolating schedule follows from gap $g(s)$ between the lowest two eigenvalues, changing faster when the gap is large, while it evolves slower when the gap is small. We therefore take these key aspects of $s(t)$ - derived for the unstructured search - and consider a similar interpolating schedule defined as follow

$$s_R C(t) = \frac{1}{2}(2(\frac{t}{T} - 1)^3 + 1) \qquad\qquad (2.7)$$

2.1.3 Multiple runs for one search

To give a complete picture of the usefulness of the time-depedent approach, we consider the possibility of repeating the search multiple times. If the probability at which the solution is found is $p = 1$ the search is *perfect* and the problem is solved. However, if the proabability is less than one, i.e. *imperfect search*, the problem can be solved by searching multiple times . Since the results of the search are checked independently, a single successfull search is sufficient, and this kind of routine is efficient as long as the probability $\left|\langle\psi(T)|w\rangle\right|^2$ is greater than $1/poly(N)$ (where N is the dimension of the graph) [5] - which as we shall later see is verified for all the scenarios considered.

./figures/interpolating_schedules/cerf.png

./figures/interpolating_schedules/our_ce

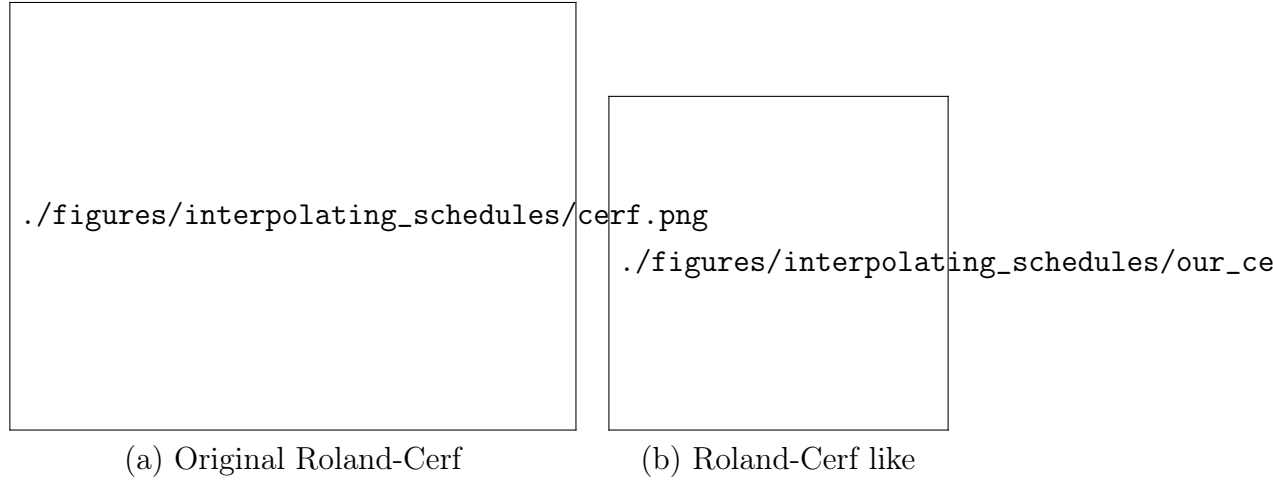(a) Original Roland-Cerf   (b) Roland-Cerf like

Figure 2.1: **Roland and Cerf's interpolating schedule (a) and our Roland-Cerf like schedule (b).** These figures show the difference between the original interpolating schedule derived by Roland and Cerf for the unstructured search (left) and our Roland-Cerf like schedule(right). Although quite different we study the effects of this particular non-linear interpolating schedule and compare it with the linear one used by Farhi and Gutmann [3].

Repeating the search multiple times does however come at a cost. It is indeed necessary to take into account for a non-zero *initialization* time t_{init} to prepare the system in the correct state as well a physical time for the measurement. Therefore, computing multiple searches with small T becomes less efficient than less searches with larger T, where the quantity t_{init} makes a lesser contribution to the overall T. Clearly this consideration is particularly relevant for increasing graph size.

2.2. Selected topologies: circular and complete graph

Throughout our analysis we will focus on two selected graph topology, the cycle graph $Cy(N)$ and the complete graph $C(N)$.

As previously discussed in Section 1.4.1 and **??** the **complete graph** represents the best case scenario since it has been shown to solve the search problem both for the standard time-independent quantum walk approach [2] and the local adiabatic evolution [8], with a quadratic speed up. An adiabatic implementation of the quantum walk search does not work with the usual Grover's oracle, requiring a more elaborate structure [10], however as we shall later see a merely time-dependent approach might give promising results in terms of robustness.

The **circular graph** on the other hand is not able to solve the search problem with the time-independent approach, and can give some interesting insights with the application of the time-dependent quantum walk search.

Short statement needed to summarize the motivation for the selected topologies. Additionally it is necessary to decide whether to introduce the complete graph first thing or lastly.

2.3. Characterization of the results: Search, Localization and Robustness

Firstly it is necessary to define what type of results we are looking for. We begin by showing the difference between **search** and **localization**. Then we introduce a (not-so-rigorous) measure for the **robustness** of the search algorithm.

2.3.1 Search vs Localization

In order to study the performance of the search algorithms we have to characterize two particular classes of results, the (optimized)**search** and the **localization**, that help us decide whether the time-dependent approach brings any advantages.

- the (optimized) **search** describes the usual search, namely the finding of

the solution with high probability (possibly unitary) for the smallest time as possible. As previously mentioned we also take into account the possibility of repeating the search multiple times.

- we call **localization** the finding of the solution with high probablity without the need to optimize the time. This description becomes necessary if we take into account the adiabatic nature of the time-dependent approach, that guarantees unitary probability for large T.

2.3.2 Robustness

We've seen that the probability of a search problem depends on the time T at which the quantum state is measured and the parameter γ. The optimal highest probability clearly is given by the optimal combination of T and γ. These two parameters might be affected by noise or perturbation, leading to variation from the maximum probability. Therefore it is interesting to define a quantity that is able to quantify this fenomenon.

Following from [9] we define the **robustness**, a quantitative measure representing the variation on the probability due to some perturbation/noise on γ. We begin by finding the highest probability p, evaluated with the single or multiple run for one search approach. For the corresponding (T, γ) combination we evaluate the robustness as follows:

$$R^{\pm} = p(T, \gamma) - p(T, \gamma \pm \delta) \tag{2.8}$$

where δ is some positive perturbation of the γ parameter. As we shall later see this quantity is given in terms of some percentage of γ. To find a unique value for the robustness an average of $R^{\pm}$ is done:

$$R_{\gamma} = \left[\frac{R^{+} + R^{-}}{2} \right] \tag{2.9}$$

The quantity R should be positive[1], since the (T, γ) combination corresponds to the hightest probability. Additionally we also notice that the perturbation on the parameter has equal probability of being positive or negative, thus the average ponderates between these two possibilities.

---

[1] Although not considered in our work, it is also possible to evaluate the robustness for non-maximal probability. In that particular scenario the value of the robustness could be negative.

As we mentioned, the variation in probability could also be due to some error on the time T at which the state is measured. We can extend the measure of robustness to this particular scenario, where the value is similarly evaluated:

$$R = p(T, \gamma) - p(T \pm \tilde{\delta}, \gamma) \tag{2.10}$$

and R follows directly from Equation (2.9). In order to differentiate from this two measure of robustness we call them **γ-robustness** and **T-robustness** respectively.

Although the value of R does not have any absolute physical significance, it fits well for our specific scenario where the interest is focused on the comparison between two specific approaches. Therefore this quantity will be solely used to characterize a particular approach as **more** or **less robust**, where the first is characterized by having a smaller value of R compared to the latter.

2.4.   Results for the Circular Graph

We now address the results for the circular graph. We begin by evaluating the probabilities for the time-independent and the time-dependent Hamiltonians. We compare the localization of the two approaches, followed by the search - which requires the introduction of a new quantity that takes into account the possibility of doing multiple runs for one search. Lastly we study the robustness of the time-dependent approach and determine whether the newly introduced Hamiltonian makes the search more robust than the time-independent one.

2.4.1   Time-Independent Benchmarks

The first step of the analysis is to compute time-independent benchmarks. We computed the probability over a (T, γ) grid, with $T \in [0, N]$ and $\gamma \in [0, 2.5]$, where N is the dimension of the graph considered. An initial run of the probability distribution shows that the probability does not increase with time (Figure2.2), thus the need to evaluate for T greater than $T = N$ proved to be unnecessary. In addition, Grover's algorithm for the unstructured search has a time scaling of $O(\sqrt{N})$ and the Farhi and Gutmann's global adiabatic evolution is $O(N)$, thus focusing on $T \leq N$ suffices.
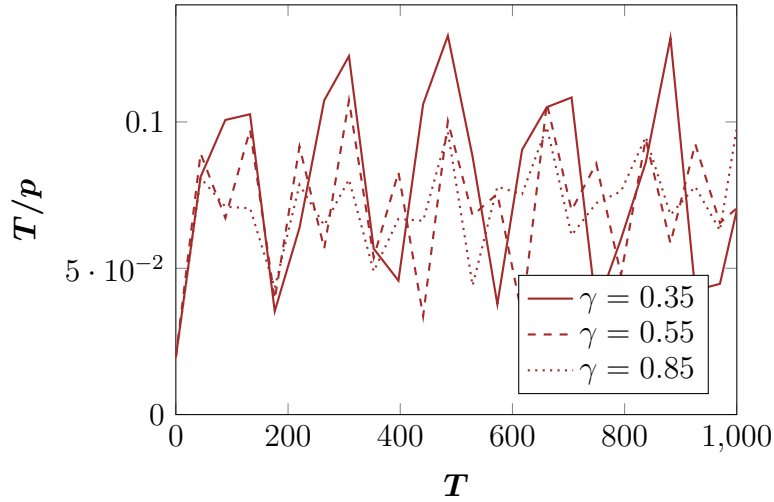

Figure 2.2: **Probability distibution for a $Cy(51)$ with sampled γ**: The figure shows the probability distribution for the cycle graph $Cy(51)$, evaluated with the time-independent Hamiltonian and a few sampled γ. We can see that the probability does not increase significantly with time, therefore we can focus our analysis on $T \leq N$.

Throughout the analysis we will consider graphs up to N = 71, with only odd dimensions which makes a easier oracle placement in the center vertex of the graph[2]. Additional considerations and reasoning for the (T, γ) grid can be found in the Appendix.

To display the results in an intuitive way we used a heatmap plot, which gives a good idea on the probability distribution for varying combinations of (T, γ):
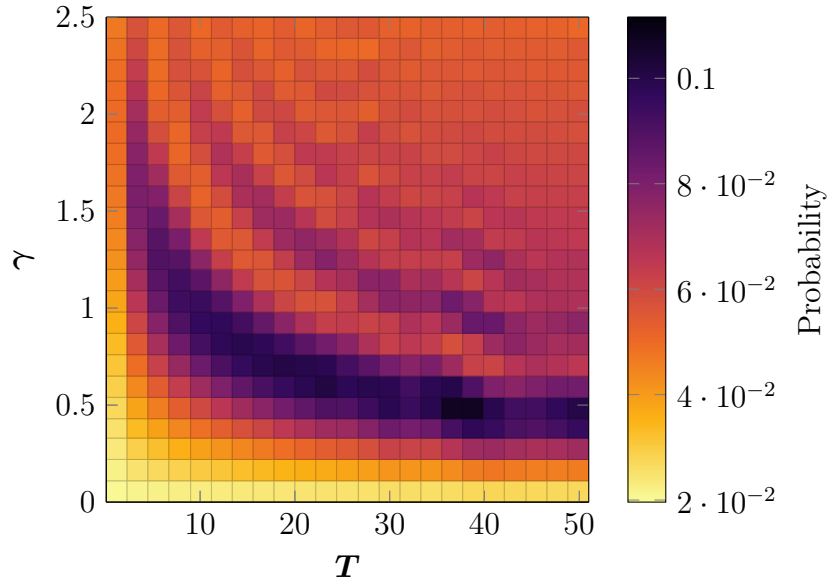

Figure 2.3: **Probability distribution for the time-independent Hamiltonian**. The figure shows the probability distribution for the cycle graph $Cy(51)$ evaluated using the time-independent Hamiltonian, providing the necessary benchmarks. Note that dark color regions do not represent probabilities close to one.

Comments on the probability distribution

From the heatmap plot we see that the probability distribution does not increase smoothly with increasing time, but shows peaks (dark regions) and valleys (lighter regions). Additionally high probability regions are scattered throughout the grid and do not appear necessarily for large T. We will later see that this is a weakness of the time-independent approach, since a small variation of the parameter γ leads to possibly great variation of the probability.

---

[2]The position of the oracle is somewhat irrelevant for the graph topologies considered, since every node is equivalent to all the $N - 1$ others. Nevertheless odd dimensions allows to place the oracle in the central vertex of the graph, namely $|\frac{N+1}{2}\rangle$.

2.4.2   Time-Dependent Results

Similarly we compute the probability with the time-dependent Hamiltonian using the interpolating schedules introduced in Section 2.1.2. To easily compare the two methods we consider the same time $T = N$ and γ used for the time-independent benchmarks. Indeed, from an initial run we see that the γ parameter affects the probability similarly to the time-independent approach, namely the probability tends to be higher for smaller values of γ.

The probability is then evaluated for all the interpolating schedules $s(t)$, and the results are again presented with an intuitive heatmap plot. **FIGURE 5**

Comments on the probability distribution

Compared to the time-independent hamiltonain approach we can clearly see that the probability distribution is smoother - has no valley and peaks - for both constant γ and T (any orizontal and vertical sections, respectively). In particular we notice that the probability increases for increasing time, and as we shall later see, for large enough T it reaches probability equal to one.

If we look at the different interpolating schedules it is immediately evident that $s_S(t)$ and $s_C(t)$ schedules perform poorly compared to the linear $s_L(t)$. On the other hand $s_{RC}(t)$ schedule performs similarly to $s_L(t)$, but has a significantly different probability distribution that might affect its robustness.

2.4.3 Comparison: Localization

We now compare the localization properties of the two algorithm. As we have already mentioned, from an intial look at the probability heatmap plots we discovered the following:

- **Time-Independent QW**: the time-independent based algorithm is not able to solve the search problem with a single iteration, making it necessary to run multiple searches. This implies that the approach does not show any localization properties, in fact the probability does not increase with time as seen in **??**.

- **Time-Dependent QW**: the time-dependent based search on the other hand solves the search problem with a single iteration, although that happens for large values of T, as can be seen in Figure?. This is indeed a consequence of the *adiabatic inspired* implementation, for which at large T the probability goes to one.
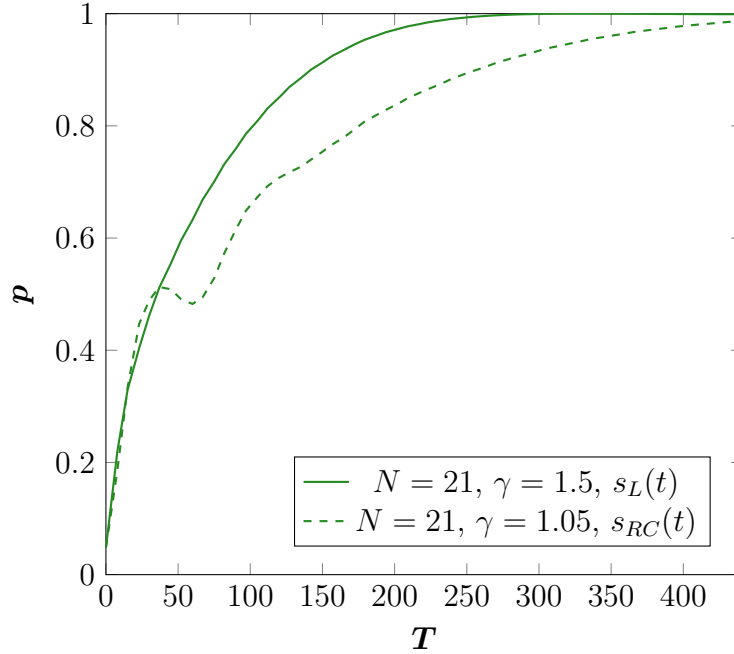

Figure 2.4: **Probability distibution for a $Cy(21)$ with sampled γ**: The figure shows the probability distribution for the cycle graph $Cy(21)$, evaluated with the time-dependent hamiltonian using the interpolating schedules (solid) linear s_L and (dashed) Roland-Cerf S_{RC}. We can see that the probability increases with time, as expected.

Although the time-dependent approach is able to get to unitary probability for large T, it is able to produce large enough probabilities in much less time, as we can see from this plot. This is a consequence of the fact that the probability does not grow linearly with time, thus needing larger T closer it gets to $p = 1$.
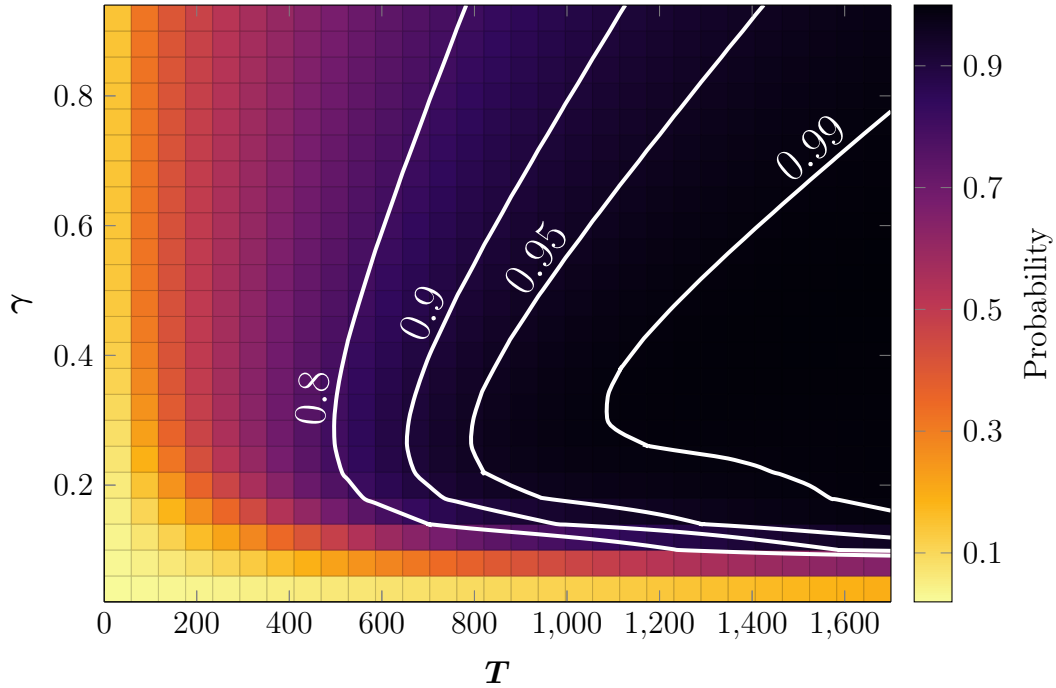Spiegare meglio cosa vuol dire large enough

Figure 2.5: **Localization at large T for the time-dependent Hamiltonian**. The figure shows the probability distribution for the time-dependent Hamiltonian using the linear interpolating schedule $s_L(t)$ up to large values of T. Although, as we can see from the white contour lines, the algorithm gets to $p > 0.99$ for a large value of T, it is able to produce $p = 0.9$ in $T/2$ and $p = 0.8$ in $T/3$.

2.4.4 Comparison: Search

In order to compare the two approaches for the search it is clear that we cannot simply consider the time at which the solution is found with unitary probability, since that particular T does not exist for the time-independent approach and is not optimized for the time-dependent one as seen for the localization results. Therefore, as previously mentioned, we consider the possibility of doing multiple runs for one search. For this reason we introduce the following quantity

$$\tau = \min \left(\frac{T}{p} \right)_{T,\gamma} \tag{2.11}$$

where the minimization is done over T and γ [3]. The quantity $1/p$ represents the number of runs necessary to get to unitary probability (statistically), and combining it with T gives the total time necessary get to $p = 1$. Minimizing over the combination of T and p gives the smallest time necessary to solve the search problem with unitary probability using the multiple runs approach.
The minimization thus consists in finding for fixed T the maximum probability, evaluating then the quantity T/p and finding the minimum.
Additionally, the number of runs performed - from now on referred to as **iterations** - given by

$$I = \min \left(p^{-1} \right) \tag{2.12}$$

might give some useful insights on the performance of the approach, in particular if you take into account the initialization time t_{init}, since large number of I are penalized by such t_{init}.

However, this approach poses a few problems. In fact if we look at how the quantity T/p varies for varying T, we discover that the minimum of such quantity will always be for the smallest T available, regardless of the type of Hamiltonian and interpolating schedule $s(t)$ considered. The following plot indeed shows, for a sampled γ, the shape of T/p with increasing time.

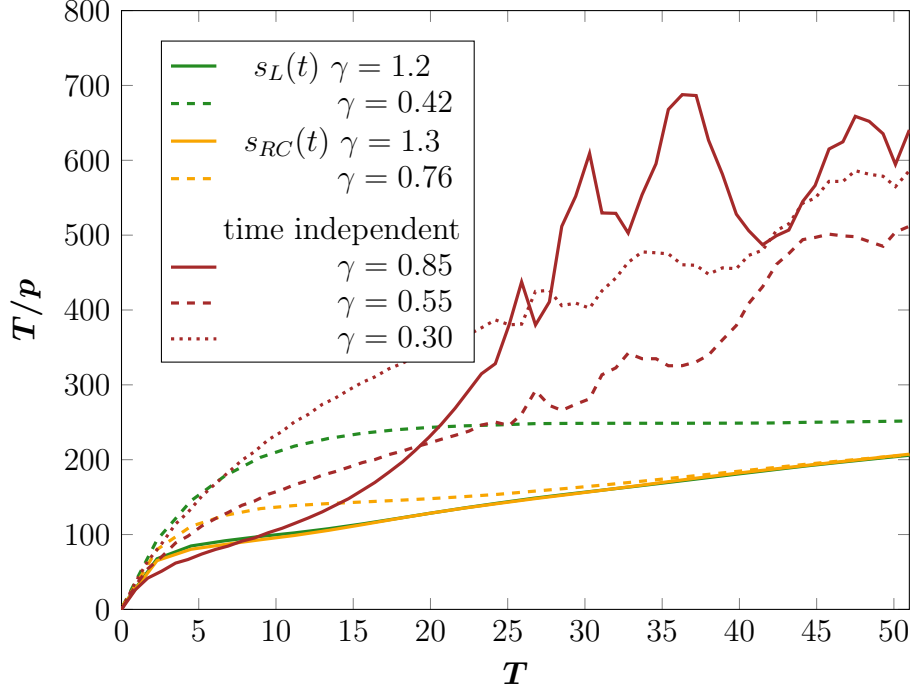[3]Remember that the probability depends directly on the combination of T and γ.

Figure 2.6: **Distribution of T/p for sampled γ showing that τ will always be for the smallest T**. The figure shows the distribution of T/p for some sampled values of the γ parameter, using the linear s_L (green) and Roland-Cerf s_{RC} (orange) interpolating schedules for the time-dependent Hamiltonian and the time-independent Hamiltonian (red). It is clear that $\tau = \min(T/p)$ will always be for the smallest T available, regardless of the interpolating schedule and the type of Hamiltonian, requiring therefore a constrain on the time.

In addition, in Section 2.1.3 we mentioned that when dealing with multiple runs we need to take into account an initialization time t_{init}. It becomes therefore necessary to introduce a minimum value of time that can be considered, since for small T the contribution given by t_{init} can have a significant impact. We call this particular T as **lower bound time**, reffered to as $T_{\min}$, and study its effect on τ and the overall performance of the time-independent and time-dependent algorithms.

τ and I for increasing lower bound time $T_{\min}$

We begin by studying how τ and I vary with increasing lower bound time $T_{\min}$. In the following plot we show the shape of T/p with the time-independent approach and the time-dependent one; since we're only interested in the general effect of $T_{\min}$ on τ and I we only consider the linear interpolating schedule $s_L(t)$[4].

---

[4]We have seen in Section 2.4.2 the probability distribution for the different interpolating

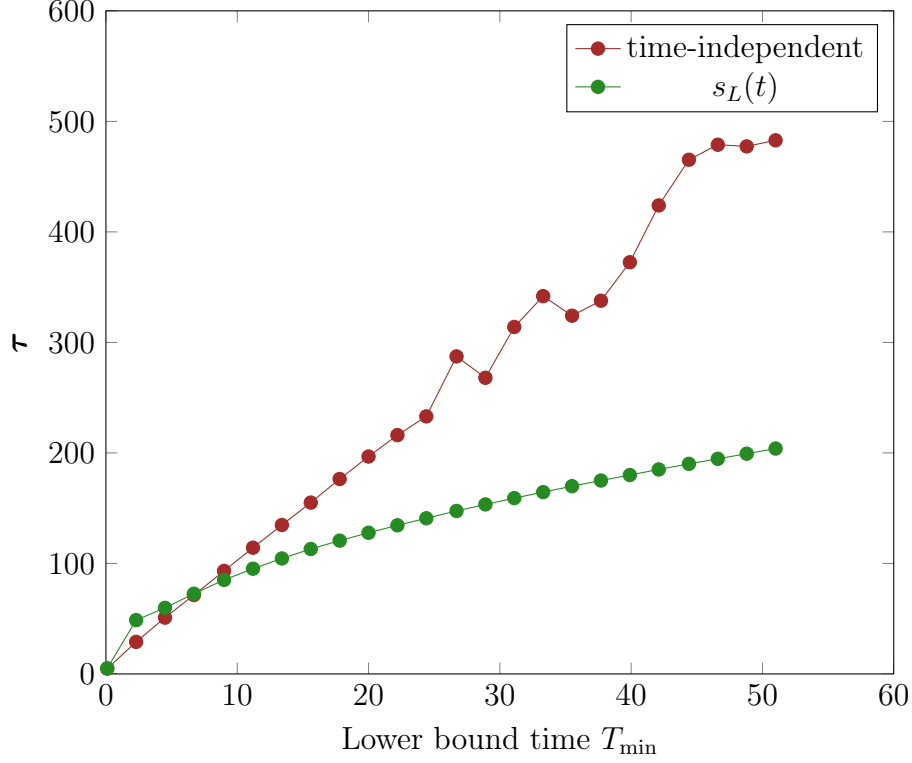For sake of semplicity we study the cycle graph $Cy(51)$ with time up to $T = N$

Figure 2.7: **τ distribution for increasing lower bound, Cy(51).** The figure shows the τ distribution for increasing values of lower bound time $T_{\min}$, using the time-independent Hamiltonian (red) and time-dependent Hamiltonian (green) with $s_L(t)$ and evaluated for a Cy(51). We see that for times smaller than a characteristic time T^* the time-independent approach performs slightly better, while for large time the time-dependent one performs significantly better.

As we can see from the plot, the distributions can be divided into two sections marked by a particular T^*, representing the time at which the two distributions - time-dependendent and time-independent - intersect (for the time being the value of such time is not of our interest):

- for $T_{\min} < T^*$ the time-dependent approach has a comparable performance with the time-independent one, although the latter has a slight advantage.

- for $T_{\min} > T^*$ however the time-dependent approach performs significantly better, in particular with increasing $T_{\min}$

The behaviour for large T is to be expected, considering that the time-dependent approach shows localization properties and the probability increases with increas-

schedules is quite similar, therefore looking only at $s_L(t)$ is sufficient.

ing time as we showed in **FIGURE 7** and in Section 2.4.3, in contrast with the time-independent approach that does not show localization properties.

What this shows is that the choice of $T_{\min}$ has a great impact on the outcome of our time-dependent approach, making it a successfull or unsuccesfull alternative. manca qualcosa che dia una conclusione a quest'ultima affermazione.

Let's now look at the distribution of the number of iterations I. The following plots shows I for increasing $T_{\min}$ up to N, using the time-independent Hamiltonian and time-dependent one with the linear interpolating schedule $s_L(t)$.
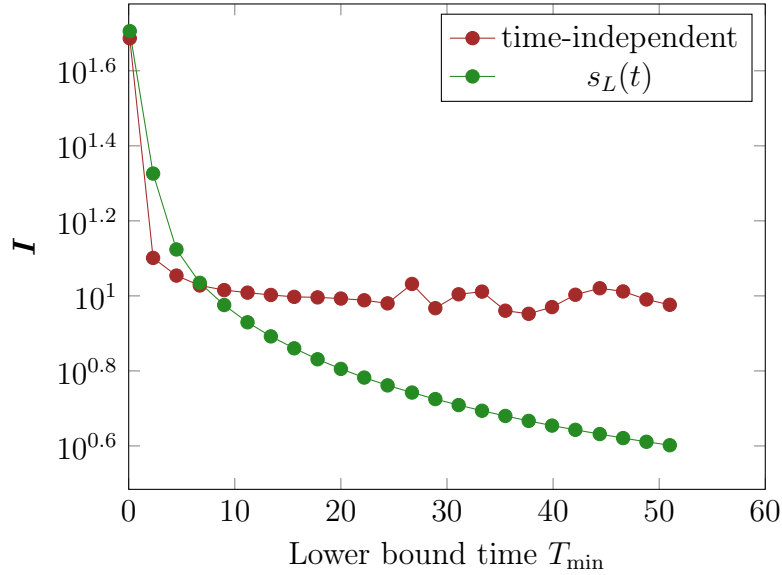

Figure 2.8: **I distribution for increasing lower bound time $T_{\min}$.** The figure shows the distribution of I for increasing values of lower bound time, using the time-independent hamiltonian (red) and time-dependent hamiltonain (green) with linear $s_L(t)$ and evaluated for a Cy(51). This distribution reflects the probability distribution of the two approaches: for the time-independent hamiltonian the probability does not increase with time, resulting in a (almost) constant I, while the time-dependent hamiltonian showing localization properties requires less iterations to get to unitary probability. Note that the plot is given with logscale y axis, which helps to highlight the time-independent trend.

The iterations distribution reflects the overall probability distribution of the time-dependent and time-independent Hamiltonian approaches. For small lower bound time the two approaches show a similar performance: the probability is very small, thus requiring a large number of iterations to get to unitary probability. As $T_{\min}$ increases we see two very different trends:

- The time independent approach requires an almost constant number of

iterations[5]. This reflects the non-localization properties of this particular approach, for which the probability does not increase with time. It also shows that the maximum probability found is (almost) equal for all T, provided that $T_{\min}$ is large enough.

- On the other hand the time-dependent approach requires less iterations to solve the search with unitary probability, as expected.

Taking into account that we are performing a multiple runs search and the initialization time t_{init} previously discussed, it is clear that the time-depedent approch performs better than the time-independent counter part in most of the scenarios, where imposing a lower bound time $T_{\min}$ becomes necessary.

τ and run iterations with constrained lower bound time

In order to show that the lower bound time does indeed have such a great impact on the performance of the time-dependent approach relative to the time-independent one, we study the distribution of τ and I with a constrain on $T_{\min}$. It should be noted that in this particular scenario τ will no longer be a minimization over T since the time is fixed, while the dependance on γ remains.

The choice of constrain is arbitrary and somewhat biased since the larger the constrain the better the performance of the time-dependent approach, as we've just shown in **FIGURE 9**. Therefore, to make the choice fair, we consider the lower bound time to be $T_{\min} = \frac{\pi}{2}\sqrt{N}$ which is the same order of magnitude of the standard Grover's time scaling, the time-independent quantum walk search on the complete graph and the unstructured local adiabatic search. In the best-case scenario we might discover that the number of iterations necessary to get to unitary probability remains constant regardless of the dimension of the graph, making this approach have the same time scaling as the ones just mentioned; in the most probable scenario we discover that the number of iterations increases with the graph size, thus adding a scaling factor that depends on some power of N.

We begin by investigating the effects of the interpolating schedule $s(t)$ on the τ distribution. The following plot shows the results for cycle graphs $Cy(N)$ with N up to 71, using the time-dependent hamiltonian and the interpolating schedules

---

[5]The numerical value of I is irrelevant since this distribution reflects only the Cy(51).
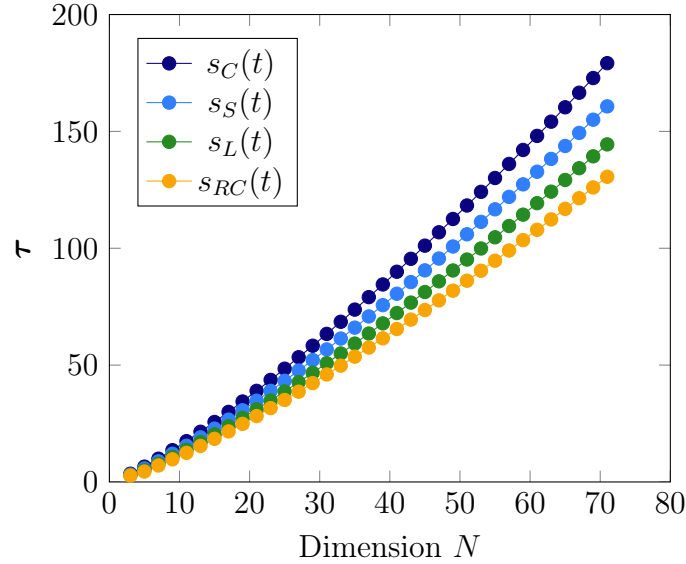
defined in Section 2.1.2.

Figure 2.9: **τ distribution with constrained time for the different intepolating schedules.** The figure shows the distribution of τ for cycle graphs up to N=71, using the time-dependent hamiltonian with different intepolating schedules: (orange) Roland-Cerf $s_{RC}(t)$, (green) linear $s_L(t)$, (light blue) $s_S(t)$ and (blue) $s_C(t)$. As expected $s_{RC}(t)$ is the best performing intepolating schedule, followed by the linear $s_L(t)$. The other two do not show any advantage compared to the others.

As we could have expected the Roland-Cerf interpolating schedule $s_{RC}(t)$ performs the best, compared to the linear one. The $s_S(t)$ and $s_C(t)$ show poor performance compared to all the others, therefore are ignored for the next analysis. This shows indeed that the shape of the interpolating function has a great impact on the performance of the algorithm, in particular when considering large graphs. We now compare the time-independent approach to the time-dependent one using the linear and Roland-Cerf interpolating schedules. Again, the time is constrained as in the previous plot. We begin by looking at the τ distribution.
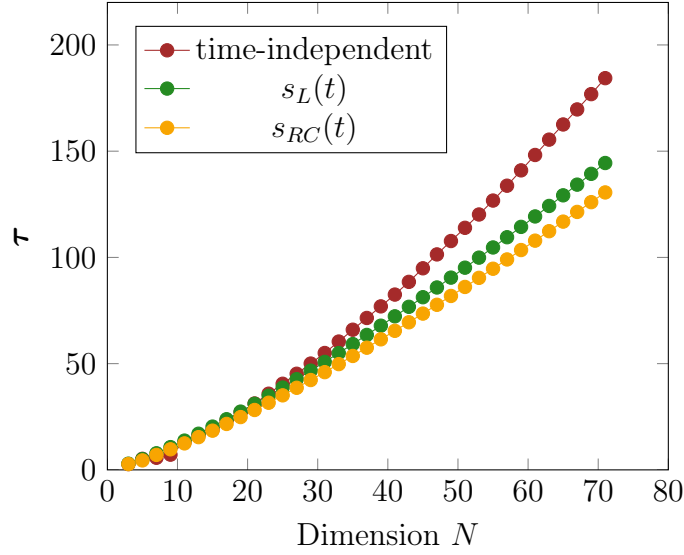
Figure 2.10: **τ distribution with constrained time at $\pi/2\sqrt{N}$.** The figure shows the distribution of τ for cycle graphs up to N=71, using the time-independent hamiltonian (red) and time-dependent hamiltonain with linear s_L (green) and Roland-Cerf s_{RC} (orange). We notice that for small graphs, up to $N \sim 20$, both approaches perform similarly. For large N however the time-dependent algorithm performs significantly better.

The following plot shows the number of iterations I for the two considered approach. We are able to qualitatively describe the time scaling of the algorithms by superimposing the functions $\sqrt{N}$ and $\sqrt[3]{N}$: if the distribution of the iterations I lies below the $\sqrt{N}$ line the time-dependent algorithm has some speedup compared to the classical search. Remember in fact that the overall time scaling of the multiple runs search is τ, that is given by
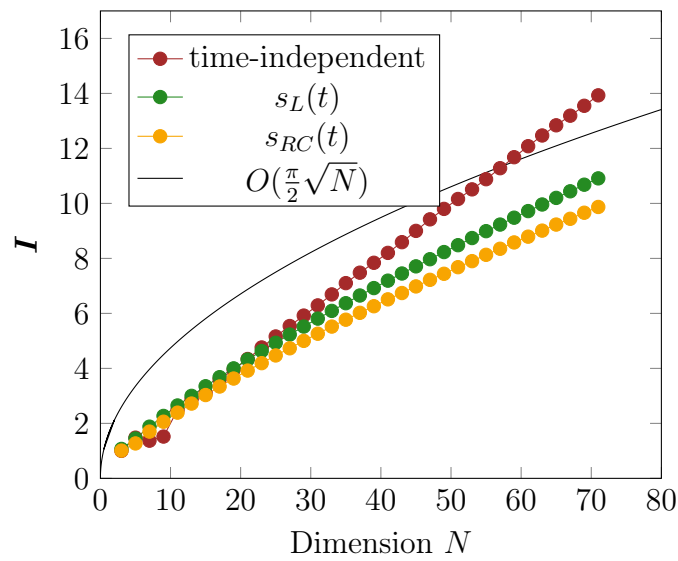
Figure 2.11: **I distribution with constrained time at $\pi/2\sqrt{N}$.** The figure shows the distribution of I for cycle graphs up to N=71, using the time-independent Hamiltonian (red) and time-dependent Hamiltonian with linear s_L (green) and Roland-Cerf s_{RC} (orange) interpolating schedules. The black line represents a time scaling of $O(\sqrt{N})$. Combining the value of I with the constrained time of $\pi/2\sqrt{(N)}$ we can estimate the performance of the algorithms: if the distribution is below the black line the algorithm performs better than the classical search $O(N)$. From this plot we can clearly see that for N up to 71 the performance is better than the classical search, although we are not able to make predictions for larger N.

2.4.5 Comparison: Robustness

We now address the robustness of the time-dependent and time-independent search. As we mentioned in Section? we are only interested in the comparison of the two approaches, and not an absolute measure of their robustness. Therefore we will use this measure solely as a comparison value.

We proceed by considering small variations on the parameter γ in the order of 1% and 5% (this numbers still need to be discussed). We begin by finding the quantity $\min(T/P)$ with time constrains $(T = 2\sqrt{N})$. For the corresponding (T, γ) combination we evaluate the robustness R as defined in Section? For the time-dependent search with only consider the linear interpolating schedule an the Ronal-Cerf(3). As we showed in the previous section the Roland-Cerf Hamiltonian performs better than the linear conterpart, while from a qualitative point of view the linear Hamiltonian has a smoother probability distribution (see **??**(a)-(d)). The following plots show the semi-quantitative robustness for the γ variations in the order of 1% - 5% for the time-independent search and the time-dependent one with linear and Roland-Cerf interpolating schedules s(t). **Computations are still needed**
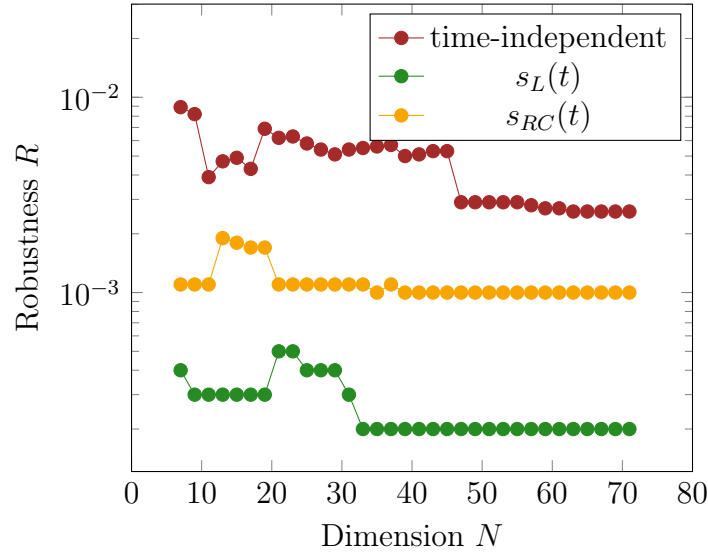
     FIGURE 15

Figure 2.12: **γ-Robustness for the time-independent and time-dependent approaches.** The figure shows the γ-robustness distribution for the time-independent approach (red), the time-dependent one with linear interpolating schedule $s_L(t)$ (green) and with Roland-Cerf $s_{RC}(t)$ (red). Recalling that the lower R value the highest the robustness, this distribution reflects the probability distribution seen in Section 2.4.2, where the probability distribution was smoother for $s_L(t)$ than $s_{RC}(t)$.

2.5. Results for the Complete Graph

2.5.1 Search results from Wong(2016)

2.5.2 Localization results

Bibliography

[1] A. M. Childs, E. Farhi, and S. Gutmann. An example of the difference between quantum and classical random walks. (2):1–4, 2001.

[2] A. M. Childs and J. Goldstone. Spatial search by quantum walk. *Physical Review A - Atomic, Molecular, and Optical Physics*, 70(2), 2004.

[3] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser. Quantum Computation by Adiabatic Evolution. 2000.

[4] L. K. Grover. Quantum mechanics helps in searching for a needle in a haystack. *Physical Review Letters*, 79(2):325–328, 1997.

[5] J. G. Morley, N. Chancellor, S. Bose, and V. Kendon. Quantum search with hybrid adiabatic?quantum walk algorithms and realistic noise. pages 1–24, 2018.

[6] O. Mülken and A. Blumen. Continuous-time quantum walks: Models for coherent transport on complex networks. *Physics Reports*, 502(2-3):37–87, 2011.

[7] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambrigde University Press, 2000.

[8] J. Roland and N. J. Cerf. Quantum search by local adiabatic evolution. *Physical Review A - Atomic, Molecular, and Optical Physics*, 65(4):6, 2002.

[9] S. Z. SH. Hung, S. Hietala. Quantitative Robustness Analysis of Quantum Programs (Extended Version). *Proceedings of the ACM on Programming Languages*, 3(January), 2019.

[10] T. G. Wong and D. A. Meyer. Irreconcilable difference between quantum walks and adiabatic quantum computing. *Physical Review A*, 93(6):1–8, 2016.