Quantum Walks and Quantum Resetting

Dhruva Sambrani

A dissertation submitted for the partial fulfilment of BS-MS dual degree in Science

Indian Institute of Science Education and Research, Mohali

May 2023

Certificate of Examination

This is to certify that the dissertation titled **Quantum Walks and Quantum Resetting** submitted by **Dhruva Sambrani** (Reg. No. MS18163) for the partial fulfillment of BS- MS Dual Degree programme of the institute, has been examined by the thesis committee duly appointed by the institute. The committee finds the work done by the candidate satisfactory and recommends that the report be accepted.

Dr. Manabendra Nath Bera Dr. Abhishek Chaudhary Dr. Sandeep Goyal

Dr. Manabendra Nath Bera
(Supervisor)

Dated: May 2, 2023

Declaration

The work presented in this dissertation has been carried out by me under the guidance of Dr. Manabendra Nath Bera at the Indian Institute of Science Education and Research, Mohali.

This work has not been submitted in part or in full for a degree, a diploma, or a fellowship to any other university or institute. Whenever contributions of others are involved, every effort is made to indicate this clearly, with due acknowledgement of collaborative research and discussions. This thesis is a bonafide record of original work done by me and all sources listed within have been detailed in the bibliography.

Dhruva Sambrani
(Candidate)
Dated: May 2, 2023

In my capacity as the supervisor of the candidates project work, I certify that the above statements by the candidate are true to the best of my knowledge.

Dr. Manabendra Nath Bera
(Supervisor)
Dated: May 2, 2023

Acknowledgements

I would like to acknowledge the financial support provided by IISc and INSPIRE-DST towards KVPY scholarship no. SA-1610027. Furthermore, I would like to also acknowledge the Staff and Administration of IISER Mohali for their support.

I would like to express my sincere gratitude to Dr Manabendra Nath Bera (Manab da) for his invaluable guidance and support throughout the course of this research as my project guide. His vast knowledge and expertise in Quantum Resetting has been instrumental in shaping the direction and scope of this study. I am also grateful for his encouragement, patience, constant motivation and the occassional necessary criticism which, even though difficult to hear, helped me stay focused and committed to this project. The freedom he allowed me while exploring different aspects of the project has made me fall in love with the topic and provided a space for my approach to the problem flourish. This has allowed me to take possession of my work, which gave me an innate sense of responsiblity towards it.

I would like to thank Dr. Arnab Pal (Arnab Da) for his invaluable support and willingness to share his expertise in Stochastic Resetting. Our discussions often ran overtime, and led to many great insights being shared from both directions. His willingness to learn from me gave me a sense of confidence and strengthened my understanding of the topic. I am thankful for his support within and outside this project inspite of his busy schedule and personal commitments.

I would like to thank the members of the QIQP group who have helped me learn a variety of topics through our weekly meetings. I would like to especially thank Anubhav for explaining concepts and helping me troubleshoot my code when we got unexpected results.

I could have hardly finished this project without the constant support from my friends. I will be forever indebted to the emotional support given by Gowri at my lowest points, which supported me to carry on with the project. She has pushed me when necessary, and provided a soft landing at other times, being there in every emotion that this endeavor brought along. Heated but stimulating discussions with Aabhas, Akshay, Aalhad and Kunal which often ran late into the night, and explaining my work to them has given insights I would have otherwise missed. Their "friendly" teasing has grown into a friendship that I've never had before. Knowing that I could drop by Aarya's lab at anytime and speculate how elephants may have died allowed me to handle the stress of a Master's Thesis. I would also like to thank Atharva, Shashi bhaiyya and Yezad for making space for me and making me feel at home. I would also like to thank Vinita for checking my mathematical proofs.

My teachers at Prakriya, Primus and Neev, and more recently those at IISER have had a profound impact on how I approached my work. While an entire list of names would need an appendix, I would like to specially thank Jaya aunty and Durga Aunty, who told me things which I only appreciated much later. I am also grateful for Dr. Neeraja Sahasrabudhe, who taught me everything I know about Markov chains, allowing me to see this work in a new light.

Finally, I would like to thank my family. Without the constant support and love of my parents, I would hardly be where I am, and I can never imagine life without them.

Abstract

While quantum walks are faster than their classical counterparts in searching a node, the transient nature of quantum walks leads to a non-zero failure rate. By classically resetting the walk, it has been shown that the walk can be made recurrent, and thus the asymptotic failure rate goes to 0 without sacrificing the quantum speedup. In this work, we attempt to define a quantum resetting mechanism, and probe its effect on the mean hit time of the walk. Such a protocol is necessary for many quantum algorithms, where classical resetting may not be possible. We define two resetting protocols, a non-unitary quantum reset motivated by the coined walk formalism, and a unitary reset via the Szegedy walk.

First we propose a controlled reset-evolve operation, drawing inspiration from the coined quantum walk formalism. We show computationally, that for a range of parameters, there is no apparent speed up gained by this protocol. Furthermore, an additional coin and non-unitary operation set renders the protocol difficult to analyse. Thus, we require a quantum reset mechanism which remains unitary. This brings us to our second resetting proposal based on Szegedy walks. We can quantise the stochastic reset classical walk, thereby achieving a unitary quantum reset protocol. We further show computationally that a speedup for the Grover-like search protocol is achieved by the unitary reset walk for a range of parameters. Finally, we also use eigenvalue analysis to analyse the space and time complexity of protocol, and show a clear advantage in running the protocol.

List of Figures

List of Tables

Table of contents

Introduction

Random walks are a commonly used tool in the arsenal of algorithms on Classical Computers to solve a variety of problems which do not have a known easy solution. The power of such methods can be attributed to the fact that while the space of possible solutions is vast, we only need to sample few solutions to come to a close solution. This power has been routinely exploited in the past to sample from Markov Chains using the MCMC algorithm which can be found in any introductory textbook[1] of Markov processes, and they have recently been used in the fields of Financial engineering[2], fluid mechanics[3], fitting black hole images[4] and most famously in the Los Alamos project[5].

With the advent of quantum algorithms, quantum walks have arisen as an obvious extension of classical walks in the quantum domain. The applications of quantum walks are just as many: ANN training[6], Random Number Generation[7], List coloring (Grover)[8], collision finding[9], link prediction[10]. There has even been a recent foray into classifying and using a quantum-classical walk to speed up certain classical algorithms, namely the Google PageRank Algorithm[11]. The increased interest in quantum walks can be attributed to the quadratic speed up which it grants the solution a-la the Grover algorithm, which itself can be considered as a Quantum Walk[12].

In this master thesis, we will first define the classical (Chapter 1) walk, and specifically, the 1D walk. The classical 1D walk has a standard deviation which goes as $\mathcal{O}(T^{1/2})$, and is recurrent.

Then we define quantum (Chapter 2) walk, particularly the coined walk on a 1D grid. The quantum 1D walk has a standard deviation which goes as $\mathcal{O}(T)$, and is transient.

We define these walks with a view of using these walks in search algorithms. We formalise the search problem and show how the quantum walk is faster, but its transient nature leads to a non-zero asymptotic failure rate in Chapter 3.

Then, we look at previous attempts at solving this by resetting (Chapter 4). By resetting a Markov chain, we can convert a transient chain to a recurrent one. We explore the effect of resetting in the quantum case, attempting to recreate past work in continuous time quantum walks in the discrete time quantum walks. We also look at possible shortcomings of the solution.

In Chapter 5, we introduce a new mechanism for quantum resetting based on superposition of the evolution and reset operations and show numerically that it does not have a clear advantage in the search problem (Chapter 5). Due to the complexity of the formalism, we approach quantum resetting unitarily.

Finally, we introduce a second protocol for reset quantum walks by quantising the Markov process via Szegedy walks (Chapter 6). We also present a Grover like search

algorithm, which is easily analyzable by eigenvalue analysis. The results and discussions of this protocol are presented in Chapter 7. We then conclude and provide future directions of this work in Chapter 8.

Along with the theory, certain aspects of the implementation of the walks computationally are also added as necessary. This is done inline instead of in an appendix, which is the norm, since the author believes that such a presentation solidifies the readers' understanding of both, the model and the implementation. We use Julia[13] and its standard libraries, Plots.jl[14], QuantumInformation.jl[15], Luxor.jl[16], Yao.jl and YaoPlots.jl[17].

1. Classical Walks

Classical random walks are defined on a graph, with the walker being on some initial node i_0 with a probability λ_i, and "hopping" from node i to j in each time step with a probability given by p_{ij}. We can thus define a transition matrix $P := P_{ij} = p_{ij}$ which is row stochastic. Thus, the probability mass function of the walker at time step t is given by λP^t. Of course, the exact structure of the graph and the probabilities will decide the properties of the walk and there exists a vast amount of literature devoted to this analysis.

We however will restrict our discussion to the symmetric walk on the 1D chain. This is often called the 1D-Simple Symmetric Random Walk, and defined in the following way.

1.1. Definition

Consider an infinite 1 D chain (Figure 1.1), with nodes marked by $\mathbb{Z}$.

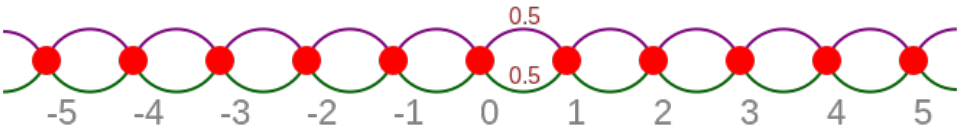


Figure 1.1.: A schematic representation of the 1D Chain. Nodes are marked by integers, and edges exist only between neighbor nodes. The simple symmetric random walk is when the walker chooses to jump along either edge with equal probability.

Define the probability of hopping from node i to node j

$$p_{ij} = \begin{cases} 1/2 & |i-j| = 1 \\ 0 & \text{otherwise} \end{cases}$$

And the initial state as

$$\lambda_{ij} = \begin{cases} 1 & i = 0 \\ 0 & \text{otherwise} \end{cases}$$

Thus, the hopper can only move to it nearest neighbors, starting from node 0.

Note that $P(X_t = i|$ HISTORY $) = P(X_t = i|X_{t-1})$, which means that the walk is a Markov chain.

1.2. Multiple Walkers

The SSRW is clearly a stochastic process, and each run of this process will lead to different paths being chosen by the walker, and we are my interested in what the walker does on an average rather than what happens in a particular instance. Thus, we can observe multiple walks, and plot their paths to visualize how they would spread in Figure 1.2.

In order to simulate a walk, we sample $X_i \in \{-1, 1\}$ with equal probability, and add it to the previous position S_{i-1} to get S_i. Note $S_0 = 0$. Thus, this is equivalent to sampling from `[1, -1]` uniformly `t` times and performing a cumulative sum. For `n` walkers, we can sample `(t, n)` such random numbers, and do a cumulative sum along the first dimension.

```
bm = cumsum(rand([1, -1], (200, 15)), dims=1);
```

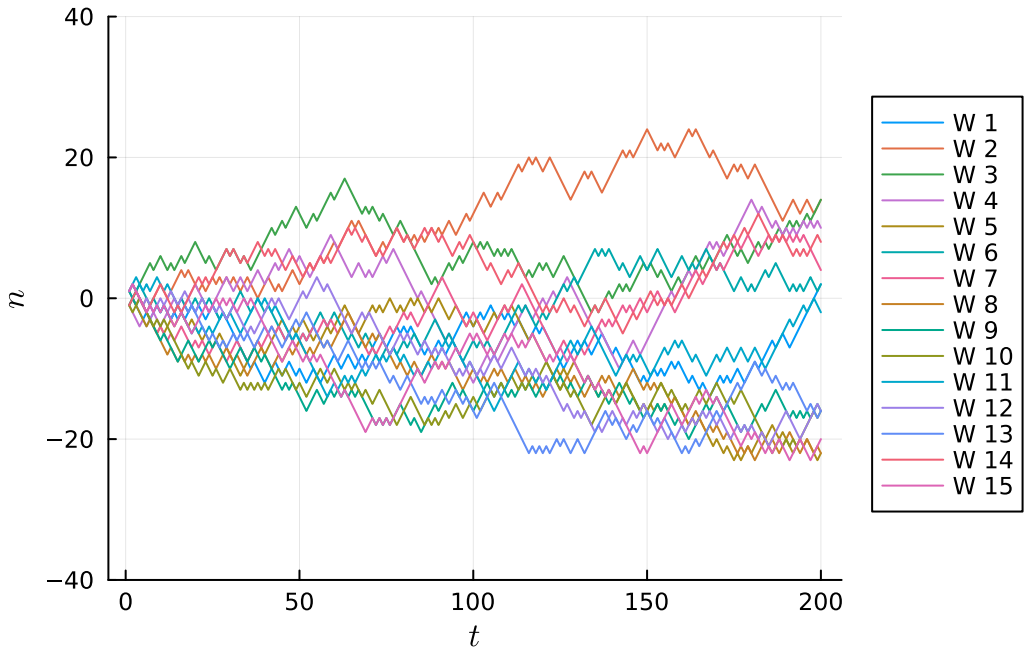


Figure 1.2.: An ensemble of classical walkers. Since the walk is stochastic, different walkers have different trajectories.

1.3. Probability mass function

Another common way to visualize the walk is to plot the probability that the walker is on node i at time t. For this, we define the transition matrix and λ appropriately

and find λP^t. Note however that the transition matrix for the SSRW is Tridiagonal with the Upper and Lower diagonals as 0.5 and the diagonal as 0, and there exist efficient storage and multiplication routines. Also, since we can only store a finite matrix, we limit the walk to some size. At the boundaries, we set open conditions, because this is the easiest to implement. Therefore, the walk can be seen as an evolution of a probability distribution, and the way that the spread occurs can be seen in Figure 1.3.

```
cps, cps_t = let
    t = 21
    U = SymTridiagonal(fill(0., 31), fill(0.5, 30))
    λ = fill(0., 31)
    λ[16] = 1
    ps = accumulate(1:t, init=λ) do old, _
        U * old
    end[t÷3:t÷3:t], t÷3
end;
```

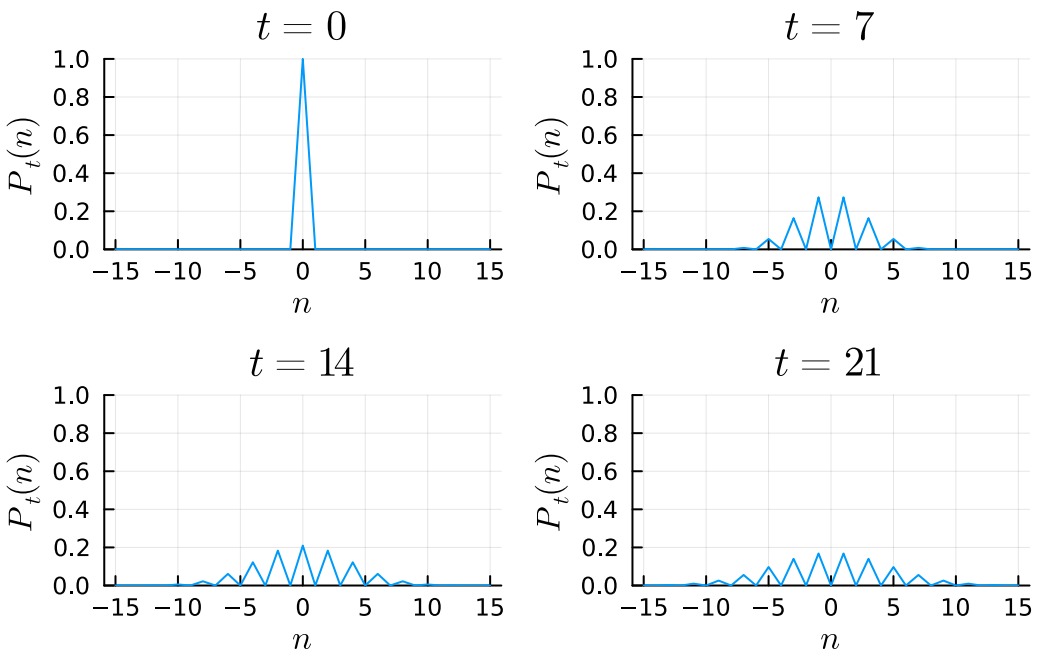


Figure 1.3.: Probability distribution of a classical walker in time. Note the well known diffusive spread of the walk. Although the walker eventually scans the entire space, it is mostly concentrated around the origin, and spreads slowly($\sigma \propto \sqrt{t}$)

1.4. Properties of the walk

Certain properties of the walk that are interesting for the problem we pose in the subsequent sections. Primarily, we are interested in the mean, standard deviation and recurrence of the graph.

The probability that a walker is on node n at time t is given by the expression

$$p(n,t) = \binom{t}{\frac{t+n}{2}} \frac{1}{2^t}$$

This equation is valid only if $t+n$ is even and $n \leq t$. If $t+n$ is odd or $n > t$, the probability is zero It should be obvious from the even symmetry of $p(n,t)$ that the mean $\mu(t) = \sum_n np(n,t) = 0$. This comes from the fact that the walk is symmetric. From Figure 1.3, we see that the walker spreads diffusively, and is more heavily concentrated in the middle of the chain rather than the edges. Mathematically, this follows from the binomial distribution.

The standard deviation of the walker position $\sigma(t) = \sqrt{\langle n^2 \rangle - \langle n \rangle^2} = \sqrt{\sum_n n^2 p(n,t)} = \sqrt{t}$[18]. Therefore, this means that the walker reaches about double the space in about four times the amount of steps.

2. Quantum Walks

2.1. Introduction

Quantum walks are defined analogous to the classical walks. First, our walker is quantum mechanical, and the position of the walker can be a superposition of the nodes. Thus, we define the state of the walker to be a superposition of the node states $|i\rangle$.

$$|\psi\rangle = \sum_i c_i |i\rangle$$

Let this Hilbert space be denoted as $\mathcal{H}_W$. The projection of the state on some node i given by $|\langle i|\psi\rangle|^2 = |c_i|^2$ is understood as the probability that the walker will collapse to the state $|i\rangle$ on measurement.

To define the evolution of the node, we look back at the transition matrix P_{ij}. Thus, we would define an operation in the following manner -

$$O|i\rangle = \sum_j \sqrt{p_{ij}}|j\rangle$$

The walk proceeds by repeated application of O.

While this expression is enough to define the walk, it is not immediately clear how one would explicitly realize such an operation. Multiple formalisms define such walks, but we shall use the coined quantum walk formalism which is very natural for regular graphs, as is the case for the 1D chain. Note that for the symmetric walks on nD lattices, like the 1D chain, the tight-binding model is already a well understood continuous time quantum walk. However, we prefer a discrete time formalism.

2.2. Coined Quantum Walk for 1D Chain

For the 1D chain, given a specific node, there are only 2 other nodes connected to it. Thus, we can add a 2 level system, which "decides" which node the walker jumps to. More formally, we attach a 2 level qubit system to the walker whose bases are denoted by $|0\rangle$ and $|1\rangle$. Let us denote this Hilbert space as $\mathcal{H}_C$

Thus, we can define the shift operation as

$$S|0\rangle|i\rangle = |0\rangle|i-1\rangle; S|1\rangle|i\rangle = |1\rangle|i+1\rangle$$

How would we define such an operation explicitly?

Controlled shift operation

$$S = |0\rangle\langle 0| \otimes S_L + |1\rangle\langle 1| \otimes S_R$$

Where S_L and S_R are defined as

$$S_L|i\rangle = |i-1\rangle,\ S_R|i\rangle = |i+1\rangle$$

Note that S operates on $\mathcal{H}_C \otimes \mathcal{H}_W$, whereas S_L and S_R operate on $\mathcal{H}_W$ only.

The superposition in the two choices at each step is recovered by putting the coin into a superposition of its basis states. This is achieved via a coin operator, which is commonly defined as $H \otimes I$, where H is the single qubit Hadamard operator.

Quantum Walk

Thus, one step of the quantum walk is defined as $S \circ H \otimes I$, operating on a two level coin and an n level system, with the state of the whole system lying in $\mathcal{H}_C \otimes \mathcal{H}_W$

Let us explicitly write down two steps of the walk

$$H \otimes I(|0\rangle|0\rangle) = \frac{|0\rangle|0\rangle + |1\rangle|0\rangle}{\sqrt{2}}$$

$$S\left(\frac{|0\rangle|0\rangle + |1\rangle|0\rangle}{\sqrt{2}}\right) = \frac{|0\rangle|-1\rangle + |1\rangle|1\rangle}{\sqrt{2}}$$

$$H \otimes I\frac{|0\rangle|-1\rangle + |1\rangle|1\rangle}{\sqrt{2}} = \frac{|0\rangle|-1\rangle + |1\rangle|-1\rangle + |0\rangle|1\rangle - |1\rangle|1\rangle}{2}$$

$$S\frac{|0\rangle|-1\rangle + |1\rangle|-1\rangle + |0\rangle|1\rangle - |1\rangle|1\rangle}{2} = \frac{|0\rangle|-2\rangle + (|1\rangle + |0\rangle)|0\rangle - |1\rangle|2\rangle}{2}$$

2.3. Repeated application of Coin Operator

Note specifically the repeated application of the coin operator. If the coin operator is not applied in step 3, our state will end up in $\frac{1}{\sqrt{2}}(|0\rangle|-2\rangle + |1\rangle|2\rangle)$ which is not what we wanted. This is because the application of the controlled shift operation entangles the coin and the walker systems, and thus there is no superposition in each term of the system

2.4. Computational Implementation

We show 2 ways to implement the Quantum walks, which are both interesting in their own ways. But the foremost thing to tackle would be how to store an infinite vector and an infinite dimensional operator. Since we cannot do either of these simply, we instead limit our walk to a node space of $2n$, and use periodic boundary conditions.

> ⚠ Other Boundary Conditions
>
> One could pick other boundary conditions too, such as the open or the absorbing boundary conditions, but both of these BCs lead to non-unitary operations, which complicate the definition and application of the gate.

2.4.1. Matrix formalism

The first and more immediate method is to simply write down the matrix equivalent of the above operations. For the visualization of the implementations, we will assume that the walk occurs in a 1 D chain of size 4.

The coin is preferred to be in the $\frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$ state when we start so that the walk proceeds symmetrically[18].

```
init_coin = 1/√2 * (ket(1,2) - 1im * ket(2,2));
```

The coin operator is trivial to write,

```
H = KrausOperators([sparse(hadamard(2)⊗I(4))]);
```

The left and right shift operators can be defined in the following manner.

```
R = collect(Tridiagonal(fill(1., 3), zeros(4), zeros(3)))
R[1, end] = 1
L = collect(Tridiagonal(zeros(3), zeros(4), fill(1., 3)))
L[end, 1] = 1
```

Thus, the shift operator is defined as

```
S = KrausOperators([sparse(proj(ket(1, 2)) ⊗ L + proj(ket(2, 2)) ⊗ R)]);
```

Thus, we can repeatedly apply $S \circ H$ to the initial state and accumulate the results.

```
init_state = proj(init_coin ⊗ ket(2,4))
ψ = [[init_state]; accumulate(1:40, init=init_state) do old, _
    H(S(old))
end];
```

Now we can plot the readout statistics by partially tracing out the coin, and plotting the diagonal. Figure 2.1 is a walk on 61 nodes.

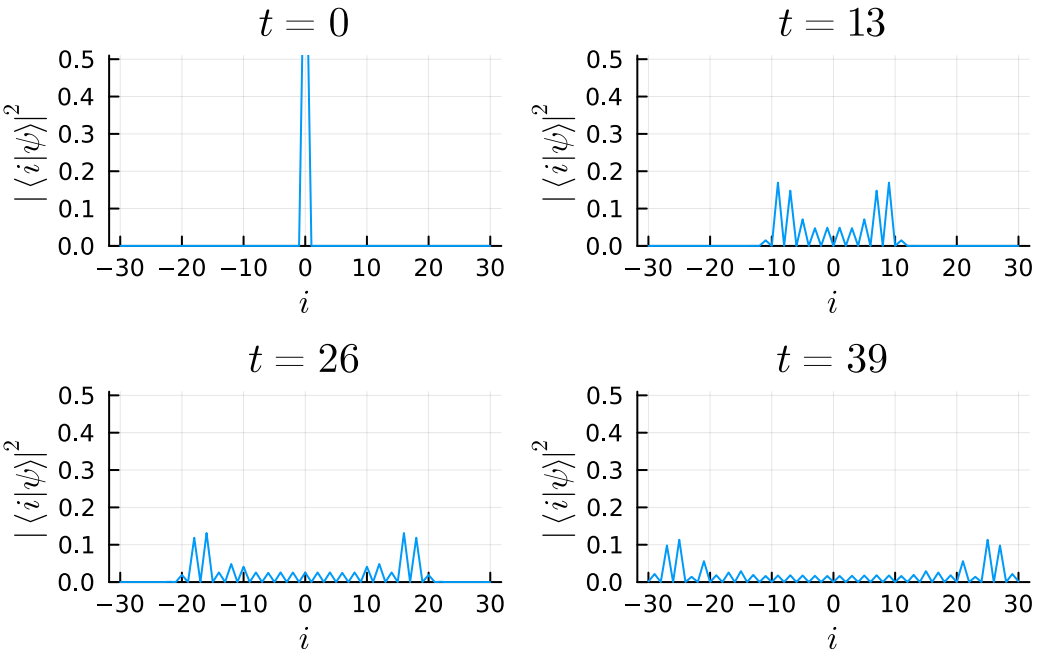


Figure 2.1.: Probability distribution of a Quantum Walker on a 1D chain in time. Note the ballistic spread ($\sigma \propto t$)of the walk, compared to the diffusive spread of the classical walk in Figure 1.3.

> **Quantum Walks are not random**
>
> Note, this is a single walker, not an ensemble of walkers as is the case in classical random walks. Quantum walks (without measurement), are completely deterministic.

2.4.2. Circuit Formalism

While the matrix definition of the Quantum Walks are easy to formalise and understand, finally these need to be simulated on some sort of device which may require us to reformulate it. Further, we try to ensure that the reformulation allows for easy additions of other dynamics which we may want to study.

The advantages of using a Quantum Computer over a classical computer for a quantum walk should be obvious. The problem however is that the quantum walk is over a high dimensional space, and we rarely have access to such high dimensional systems which we can control easily. Instead, we need to simulate such a system using the accessible 2 level systems available in quantum computers.

Let us define the basic components of our walk.

2.4.2.1. Nodes

- Each numbered node is then converted into its 2-ary n length representation denoted by (x_i). n is chosen such that $2^n > N$

- These bitstrings are encoded into an n qubit computer, where each basis state in the computational basis corresponds to the node with the same bitstring.
- The amplitude of a particular basis corresponds to the amplitude of the walker in the corresponding node

2.4.2.2. Coin

- The Walk Coin is a two level system, as usual

2.4.2.3. Edges and Shifts

- Since shifts are only to adjacent nodes, the left (right) shift is equivalent of subtracting (adding) 1 from the bitstring of the state.
- From the Quantum adder circuit, we can set one input to be $(0)_{n-1}1$ and reduce the circuit to get the Quantum `AddOne` circuit. Shown below is the circuit for $n = 4$ ($N = 16$)
- We can similarly construct the `SubOne` circuit, but that is simplified by noting that the `SubOne` circuit is simply the inverse of the `AddOne` circuit, and this corresponds to just inverting the circuit (all gates are unitary).
- The circuits for the left (and right) shift operation is shown in Figure 2.3 (and Figure 2.2)

```
rightshift(n) = chain(
    n,
    map(n:-1:2) do i
        control(1:i-1, i⇒X) end...,
    put(1⇒X)
)
leftshift(n) = rightshift(n)';
```

```
YaoPlots.plot(rightshift(4))
```

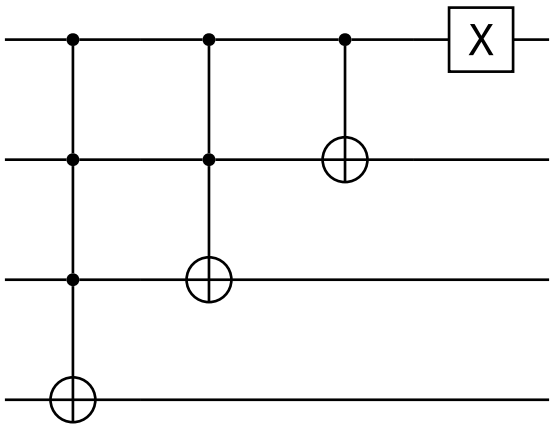


Figure 2.2.: A diagrammatic representation of the right shift circuit.

```
YaoPlots.plot(leftshift(4))
```

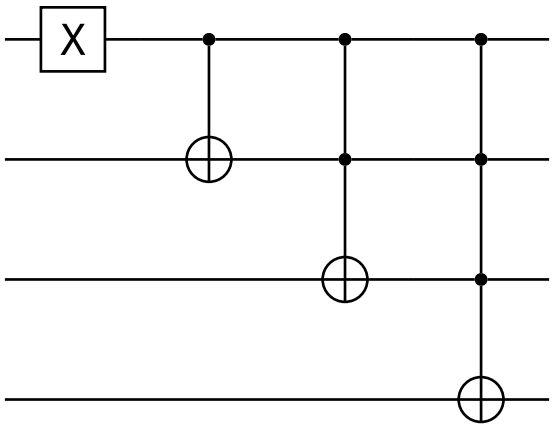


Figure 2.3.: A diagrammatic representation of the left shift circuit.

- The controlled shift operation is encoded by controlling on the coin qubit as seen in Figure 2.4.

```
shift(n) = chain(n+1,
    control(1, 2:n+1 ⇒ rightshift(n)),
    put(1 ⇒ X),
    control(1, 2:n+1 ⇒ leftshift(n)),
    put(1 ⇒ X),
)

YaoPlots.plot(shift(4))
```

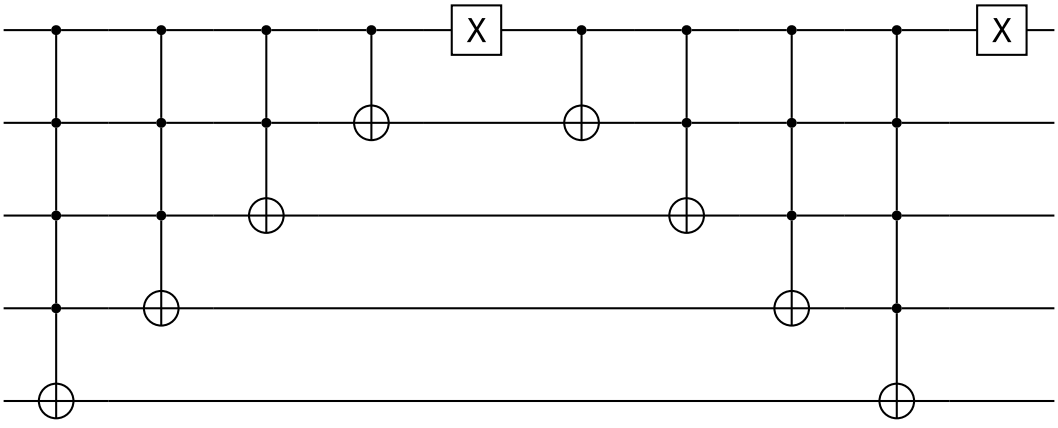


Figure 2.4.: A diagrammatic representation of the controlled shift circuit. This acts as the Controlled shift operator.

2.4.2.4. Coin operator

We can add the coin operator as usual on the first rail (Figure 2.5).

```
coin(n, c=Yao.H) = chain(n+1, put(1 ⇒ c))
YaoPlots.plot(coin(4))
```

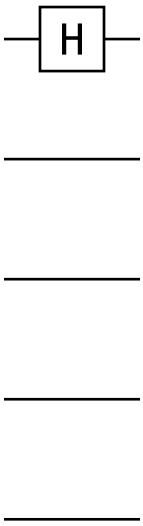


Figure 2.5.: A diagrammatic representation of the coin operator circuit. This acts to introduce superposition in the coin.

2.4.2.5. Evolve Circuit

Putting these together, we get the operation for a single step of the evolution as below. Note that the top qubit rail is that of the coin, and the rest are those of the simulation of the system.

This circuit (Figure 2.6) can be repeated to achieve any number of steps.

```
evolve(n) = shift(n) * coin(n)
YaoPlots.plot(evolve(4))
```

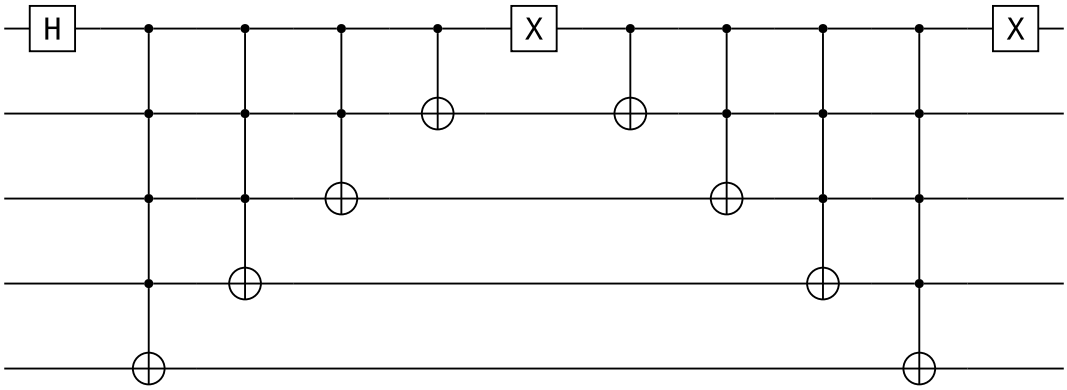


Figure 2.6.: A diagrammatic representation of the Quantum walk evolve circuit, composed of a Coin operator followed by a Controlled shift. This acts as the Walk operator

2.4.2.6. Prepare circuit

While we can already simulate the walk, a very useful helper function that we can define is the prepare circuit.

Quantum registers are often initialized to the 0 state, and it is also easy to restart the walk from the 0 state. However, we often like to start the walk in the center of the chain instead of at node 0. Also, the coin is preferred to be in the $\frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$ when we start so that the walk proceeds symmetrically[18].

Hence, to perform these steps, we define the following `prepare` subroutine (Figure 2.7).

```
prepare(n) = chain(
    n+1,
    put(1⇒Yao.H),
    put(1⇒Yao.shift(-π/2)),
    put(n+1⇒X)
)

YaoPlots.plot(prepare(3))
```

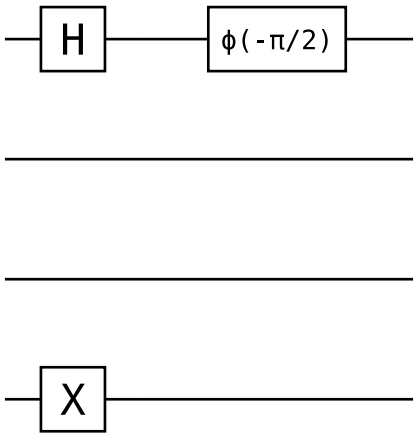


Figure 2.7.: A diagrammatic representation of the initial state preparation circuit.

Plotting the walker distribution (Figure 2.8) as before, we see that the two implementations are equivalent.

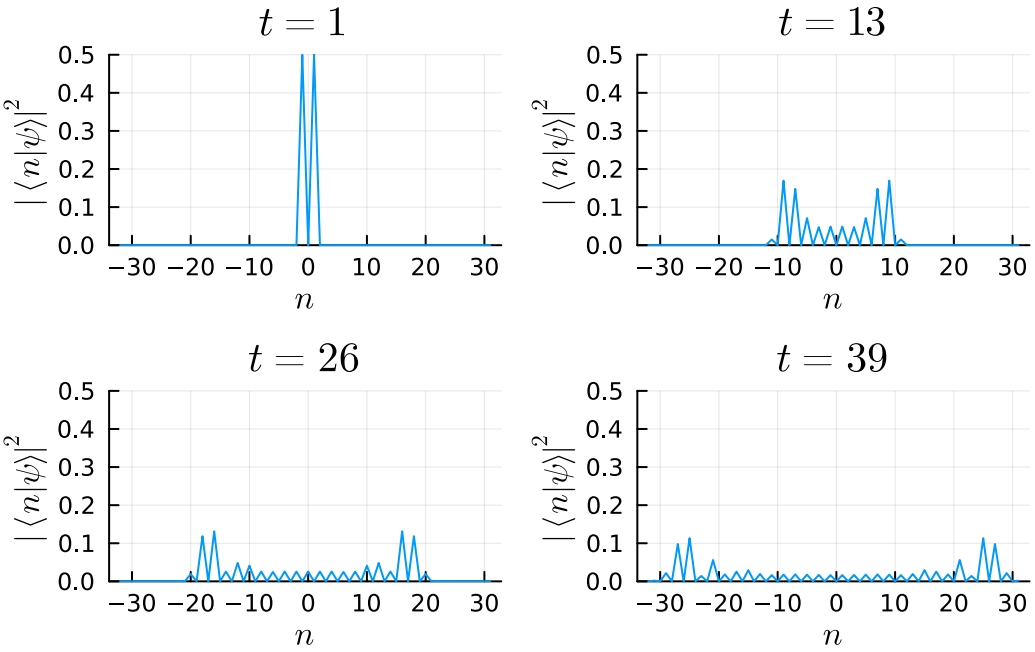


Figure 2.8.: Quantum walk using circuit formalism. For this simulation, we use 6 qubits ($2^6 = 64$ nodes) to represent the walker system. See that the probability distribution changes similarly to Figure 2.1

2.5. Properties of the walk

Similar to the classical random walk, the mean $\mu(t) = 0$, if the initial state of the coin is $\frac{1}{sqrt2}(|0\rangle - i|1\rangle$. If the coin is in another state, then the walk is biased towards a direction.

A more interesting feature is that the standard deviation $\sigma(t) = 0.54t$[18]. Compare this with the classical random walk (Figure 2.9), the quantum walk has a quadratic speed up. This is the reason for the quadratic speedup commonly seen in the Grover search and other Monte Carlo problems.

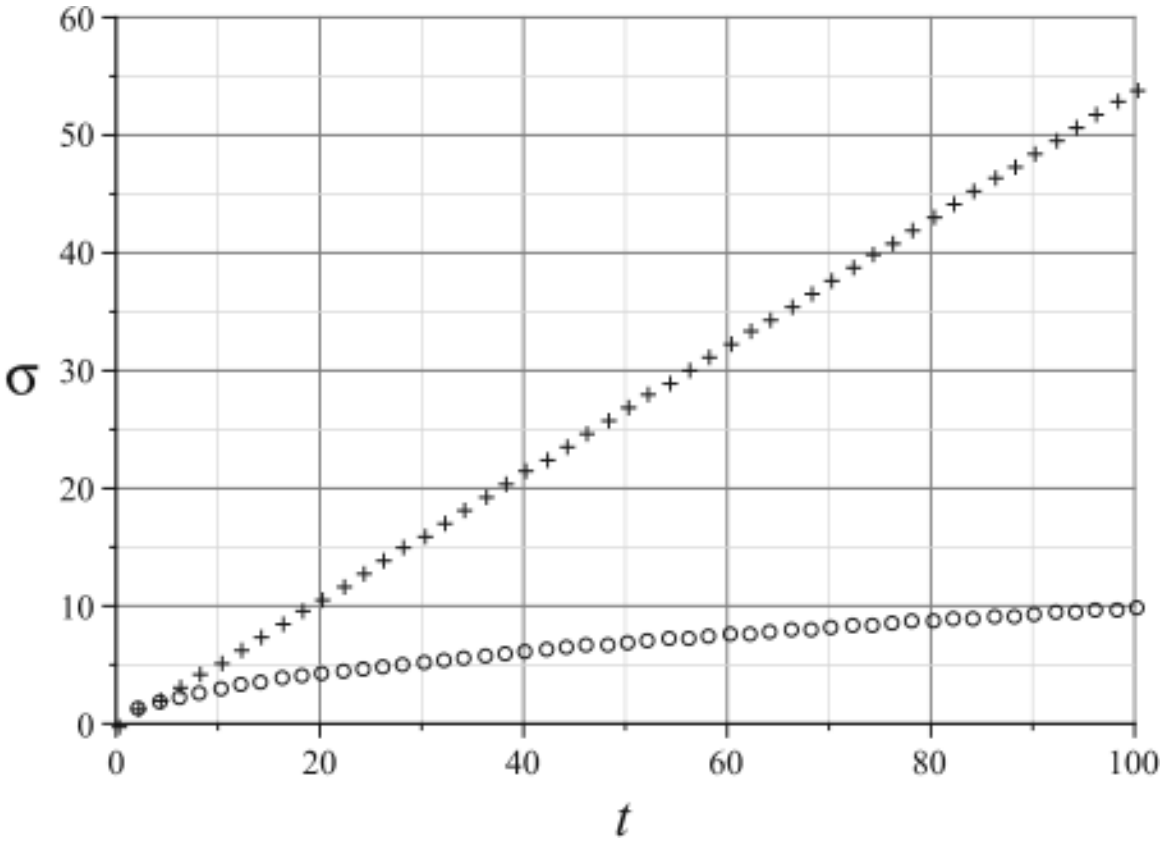


Figure 2.9.: A comparision between the standard Deviation with time for the quantum(*crosses*) and classical random(*circles*) walks[18]. It is clear that the quantum walk spreads ballistically, whereas the classical walk spreads diffusively.

2.6. Why the speed up?

We have seen that the quantum walk spreads ballistically and therefore shows a quadratic speed-up over the classical walk. This can be explained much more clearly from the continuous time versions of these walks[19].

In the classical case, we have the master equation -

$$\frac{\partial P(x,t)}{\partial t} = \gamma \left[P(x+1,t) + P(x-1,t) - 2P(x,t)\right]$$

In the quantum case, we have the Schrodinger equation -

$$-i\frac{\phi(x,t)}{\partial t} = \gamma \left[\phi(x+1,t) + \phi(x-1,t) - 2\phi(x,t)\right]$$

These look almost the same, but the most important difference is that where the classical walk is an evolution of probability distributions, the quantum walk is an evolution of *probability amplitudes*; where probability distributions must be positive, probability amplitudes may be negative, or even complex. This is the key ingredient that allows us to introduce interference within the forward-moving and backward-moving parts of the wave, such that the walker concentrates on the edges rather than the middle, thereby exploring the space faster.

3. Search Algorithms

A common application of walks is in search problems. In search problems, we generally have a black box function

$$f(x) = \begin{cases} 0 & x \in G \\ 1 & x \in G^C \end{cases}$$

where $G \cup G^C = S$ which is the search space. We are interested in developing an algorithm to output some $x \in G$ by querying f.

Naively, one can run a stochastic process over the domain $G \cup G^C$ and observe the system until we see an element in G. The average time to succeed in this protocol is called the hitting time of G, and denoted as $HT(P, G)$. Naturally, we not only want high success rates, but we also want low mean hit times.

In the particular case of Markov chains, hitting times are a kind of stopping time (Appendix A), and are well studied in their own regard.

3.1. Formalism

Define the following

- Denote the readout at the n^{th} measurement (at $t = n\tau$) as X_n.
- Select a target node δ
- Probability of first hit in n steps $F_n = P(X_n = \delta | X_i \neq \delta \forall i \in [0, n-1])$
- Mean hit time $\langle t_S \rangle = \sum_{i=0}^{\infty} iF_i$
- Success probability in n steps $S_n = \sum_{i=1}^{n} F_i = P(\exists i \in [0, n] | X_i = \delta)$
- Survival probability = Failure probability $= \mathcal{S}_n = 1 - S_n$
- Asymptotic versions of these terms are given by taking $n \to \infty$

3.1.1. Readout in Walks

To identify whether a walker has hit the target node, we need to track the location of the walker as it evolves in time.

For the classical case, this poses no problem, as measurement does not disturb the system. In the quantum case however, we need to be a bit more careful.

A constantly measured walker will freeze the dynamics of a quantum walker. This is known as the Quantum Zeno effect. The solution for this is to measure after every τ steps.

! Semi-quantum walk

This results in what is technically a semi-quantum walk[20], since measurement leads to a complete loss of coherence in the system. However, the walk between two measurement events is quantum in nature, and increasing τ leads to the quantum features of the walk dominating. We shall however keep this at the back of our minds and refer to this walk as the quantum walk, until we reach Chapter 6.

i Reduction to classical random walk with $\tau = 1$

The quantum $\tau = 1$ case reduces the discrete time quantum walk to a classical random walk with $\tau = 1$. Effectively, we apply a dephasing operator on the density matrix, dropping all off diagonal terms. Thus, we lose all effects of superposition, causing the classical random walk.

Plotting the readout trajectories of walkers with different parameters in Figure 3.1, we see the very clear difference in the spread between the classical readout and the quantum readout for same τ: The quantum walker spreads much faster than the classical walker, and thus scans a larger area. Similarly, note the spread between the quantum readouts for different τs: A larger τ value leads to the walk behaving more quantum-like, corresponding to a larger spread for the same number of measurements.

3.2. Markov Chains and Walks

In the classical case, it is clear that the 1D SSRW is a Markov process. In Appendix B, we show that the quantum walk with measurement is also a Markov process. Consequently, we can use certain results from the theory of Markov processes to compare the two walks.

3.2.1. Irreducibilty and Recurrence

Under the usual definition of irreducibility (Appendix A), it is trivial that in the 1D chain, $n = m = |i - j|\tau$ satisfies the condition. It is a well known fact that the 1D SSRW is recurrent. It is just as well known a fact that the 3D SSRW is transient[1].

In the quantum case, it is not as clear whether any of the available definitions of the quantum Markov process is more or less good than the others. Thus, the transience of quantum walks depends on the exact definition we are working under[21]. In our definition however, it can be shown using symbolic computation[22] that the walk is indeed transient. This poses an interesting problem which we shall discuss below.

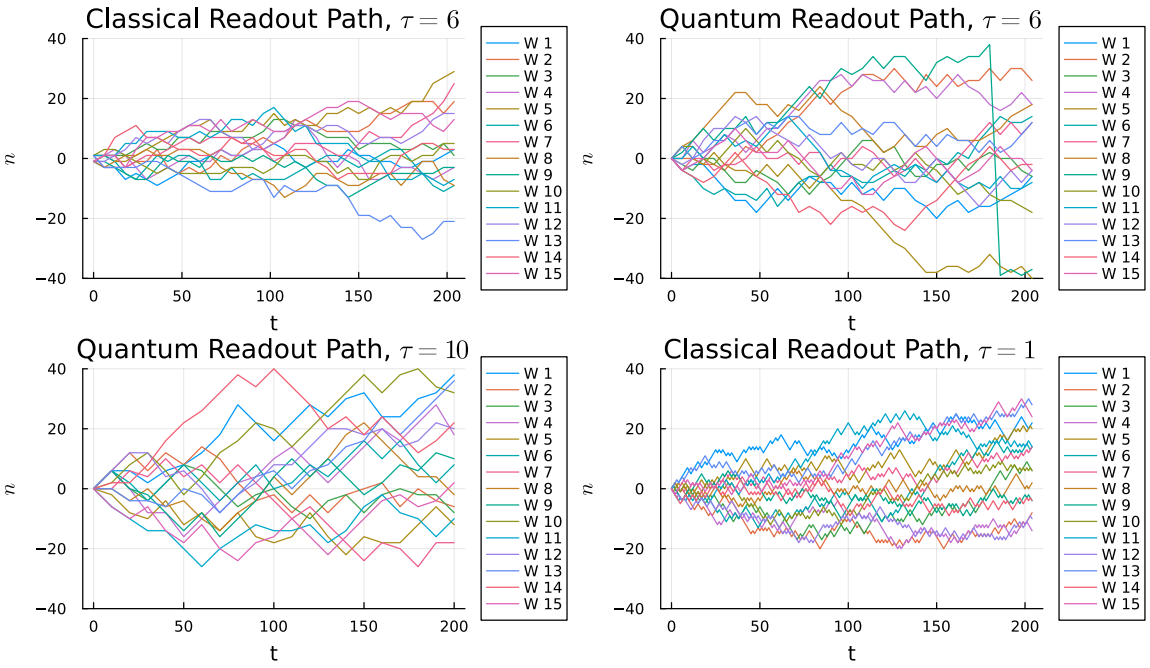


Figure 3.1.: Trajectories of readouts in quantum and classical walks. The dependence on the spread of the walk on the time between measurement is clear, with larger τ values spreading more, and the quantum walks spreading faster than their classical counterparts.

3.3. Survival probability

We can thus measure and plot the $S_n - n$ curve for the quantum and classical walks to compare the efficiency of these walks in the first hit problem.

In previous work[19], the analytical solution of the success rate as a function of time is found. An analytical result for discrete time walks is harder due to the nature of the walk. However, we reproduce the results for the discrete time case computationally. Figure 3.2a[19] shows the success rate for the continuous time quantum and classical walks, and Figure 3.2b shows our results for the discrete time case.

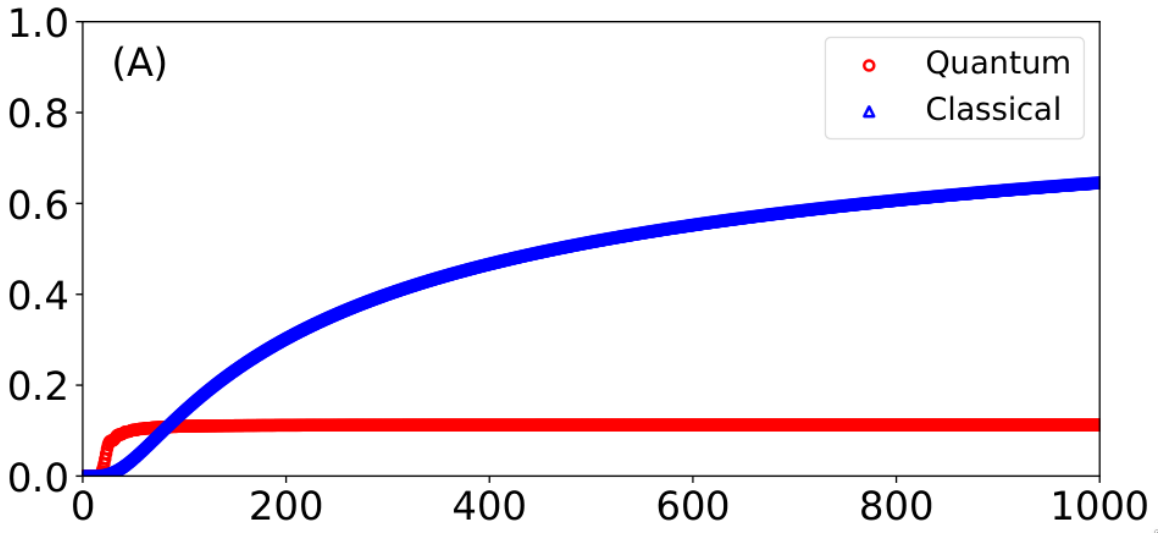
3.3.1. Observations - The Tortoise and the Hare

In the *continuous time* case:

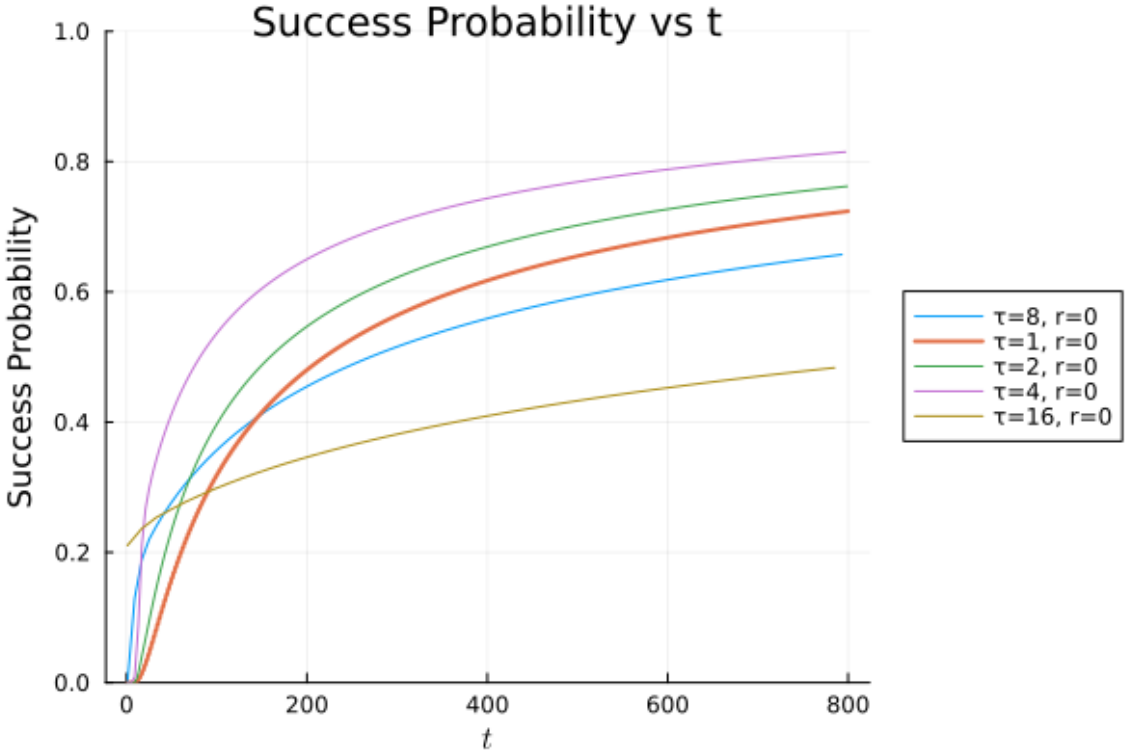
- The quantum walk has a fast rise in the initial phase but saturates at ~0.1
- The classical walk has a slow rise, but eventually reaches 1

In the *discrete time* case, solved computationally:

- Both walks do eventually reach 1
- The quantum walk shows a saturation for a while before suddenly rising again
- The classical walk shows a slow rise in the beginning phases, in contrast to the sharper rise of the quantum walk.



(a) Success probability vs Time for both walks[19], continuous time. $\tau = 0.25, \delta = 10$. Asymptotic success rate of the quantum walk is approximately 0.1, whereas the classical walk approaches 1.



(b) Success probability vs Time for both walks, discrete time. $\delta = 10$. See that more "quantumness" in the walk leads to a faster initial rise, but saturation at lower asymptotic success, allowing the slower classical walk to overtake.

Figure 3.2.: Success probability vs Time for both walks, continuous and discrete time.

There seems to be an apparent difference between the discrete and the continuous time walks, where in the discrete walk, the asymptotic success reaches 1 even in the quantum case, but this is only an artifact of the finite size of the walk space. It is well known that any irreducible finite chain is recurrent[23]. This claim can be verified by simply increasing the size of the state space, and noting that the saturation phase in the quantum walk lasts longer.

The observations in the continuous time case can be explained by the recurrence of the walk. While the quantum walker is faster (See Chapter 2), the transient nature (See Section 3.2.1) of the walk leads to a non-zero asymptomatic failure rate. Clearly this is a problem. What this means practically, is that for the first hit problem, if the quantum walk hits the target node, it does so faster than the classical walk, but a majority of the times, it doesn't hit the target node at all. This is reminiscent of the age-old fable of a race between the Tortoise and the Hare. The Tortoise-like classical walk is slow, but trudges on to the finish line, whereas the Hare-like quantum walk speeds away in the beginning, but falls behind.

But we want a walk that is both fast, and with an asymptotic success rate of 1. Can we give something to our Hare so that it doesn't fall asleep?

4. Stochastic Resetting

The crux of the matter is this. The quantum walk is fast in the initial phase, but eventually saturates asymptotically to a success rate of < 1. This means that some of the walkers hit the target, and others do not. If that is the case, is it possible to restart the walk when it starts saturating? That is, can we restart the walk for the walkers that have not yet hit the target after $r\tau$ time?

For such dynamics, we will need to reformulate the walk slightly.

4.1. Formalism

A reset of the walker (classical or quantum) implies that the walker returns to its initial state, and the walk dynamics continue from there.

However, we can vary when we reset the walker by considering multiple reset processes. A common reset process is the Poisson reset, where the reset times are modelled as a Poisson process with some parameter r. This is particularly convenient in the case of continuous time walks[24].

For the discrete walks, the geometric distribution is more natural.

$$t \sim \text{Geom}(\gamma)$$

, that is, at each step, there is a γ probability of reset. This results in different dynamics, which can be seen in subsequent subsections, but it has an equally drastic effect on the recurrence of the walk.

4.1.1. Recurrence and Resetting

In the geometric resetting case, it is clear that $P_{00}^n \geq \gamma$, and hence $\sum_n P_{00}^n \geq \sum_n \gamma$ which diverges as $n \to \infty$. Thus, regardless of the initial walk, the final walk will definitely be recurrent. Thus, the motivation for resetting the quantum walk which is transient should be immediately clear.

4.1.2. Stochastic Reset Classical Walk

In the classical case, the transition probabilities change to

$$p_{ij} = \begin{cases} (1-\gamma)\cdot 1/2 & |i-j| = 1 \\ \gamma & j = r_0 \\ 0 & \text{otherwise} \end{cases}$$

Thus, the new transition matrix is modelled as

```
γ = 0.2
U1 = sparse(SymTridiagonal(fill(0., 31), fill(0.5, 30)))
R = fill(0., (31, 31))
R[21,:] .= 1
U = sparse((1-γ) * U1 + γ * R);
```

where `U1` is the unchanged walk matrix and `R` is the reset matrix. We plot the probability distribution of the stochastic reset classical walk in Figure 4.1.

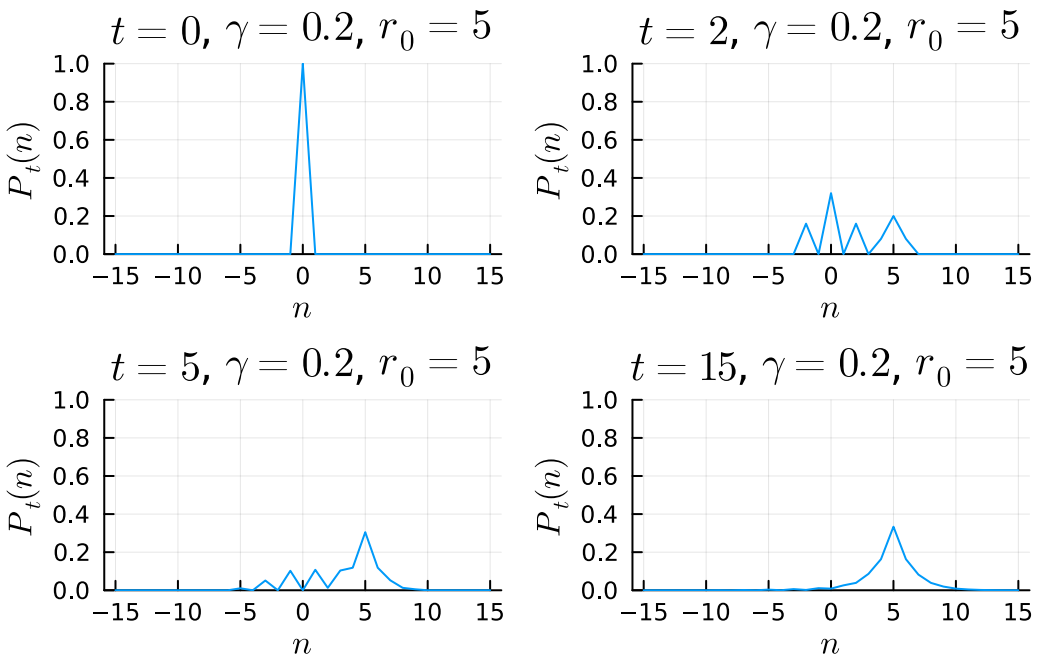


Figure 4.1.: Probability distribution of the Stochastic Reset Classical Walk. See that the walker is now stably localized near the reset node. At intermediate times, we see features of the walk and the resetting.

4.1.3. Stochastic Reset Quantum Walk

In the quantum case, the evolution changes from unitary dynamics to a non-unitary CPTP map of the following form.

$$O_{SR}(\rho) = (1-\gamma)\left(U\rho U^{\dagger}\right) + \gamma|r_0\rangle\langle r_0|$$

where U is the walk unitary. Plotting this walk in Figure 4.2, we see certain similarities with the stochastic reset classical walk.

```
function swqw(n, γ)
    R = collect(Tridiagonal(fill(1., n), zeros(n+1), zeros(n)))
    R[1, end] = 1
    L = collect(Tridiagonal(zeros(n), zeros(n+1), fill(1., n)))
    L[end, 1] = 1
    U = KrausOperators(
        [sparse(proj(ket(1, 2)) ⊗ L + proj(ket(2, 2)) ⊗ R)]
    )
    init_coin = 1/√2 * (ket(1,2) - 1im * ket(2,2))
    H = KrausOperators([sparse(hadamard(2)⊗I(n+1))])
    init_state = proj(init_coin ⊗ ket(n÷2 + 1, n+1))
    ψ = [[init_state]; accumulate(1:40, init=init_state) do old, _
        (1-γ)*H(U(old)) + γ*proj(init_coin ⊗ ket(n÷2+6, n+1))
    end]
    map(enumerate(real.(diag.(ptrace.(
        ψ, Ref([2, n+1]), Ref([1])
    ))))) do (t, ps)
        Plots.plot(
            -n÷2:n÷2, ps,
            ylims=(0, 1), xlabel="\$i\$",
            ylabel="\$\\langle i|\\psi\\rangle\$",
            title="\$t=$(t), \\gamma=$(γ), r_0 = |5\\rangle \$",
            legend=false
        )
    end
end;
```

4.2. Effect of Resetting on First Hit problem

For this analysis, we will consider the simpler "sharp reset" formalism[19], where the reset time is sampled from a distribution

$$t_r \sim \delta(t - r\tau)$$

Therefore, we restart after r measurement events (at $t = r\tau$). Once we understand the effect of this kind of reset, gaining intuition for reset times sampled differently is easier.

The success probability versus time can now be plotted (Figure 4.3a)[19] for the reset case.

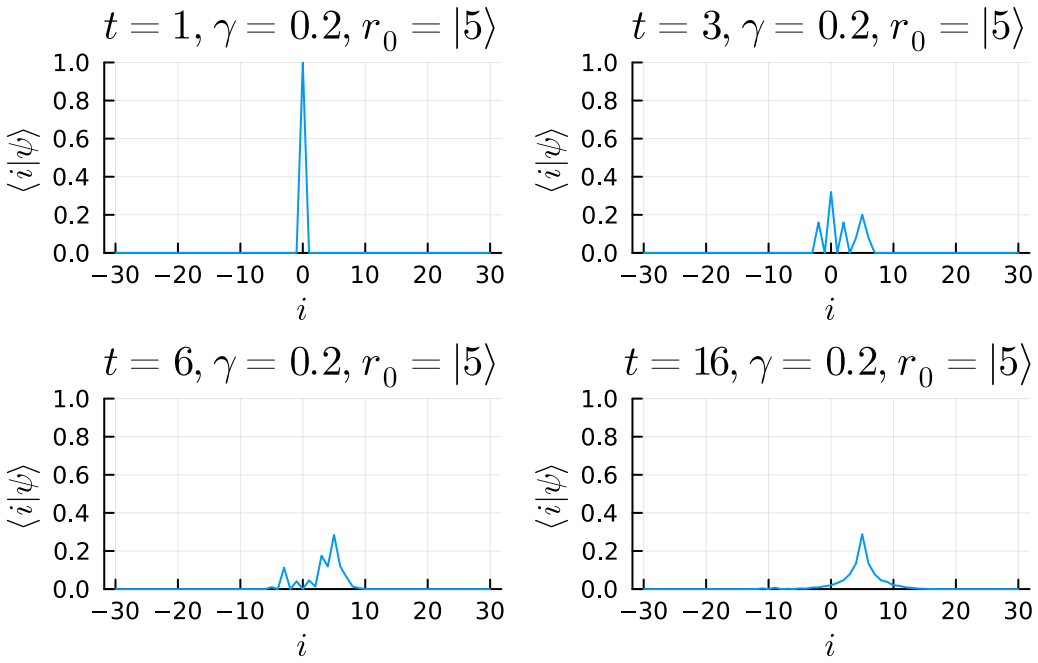


Figure 4.2.: Probability distribution of the Stochastic Reset Quantum Walk. See that the walker is now stably localized near the reset node, but is much more spread out compared to the classical case(Figure 4.1) as an obvious result of the faster quantum walk. At intermediate times, we see features of the walk and the resetting.

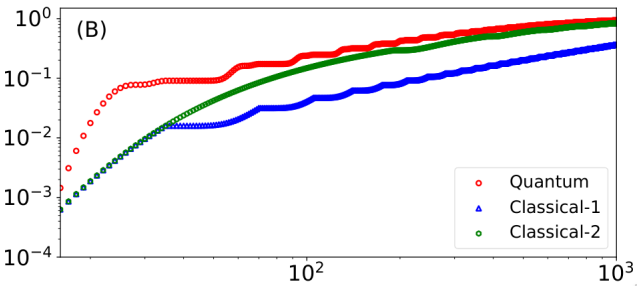
Now we see that the success probability is drastically increased for both cases, but due to the ballistic nature of the quantum walk, we see that the reset quantum walk performs much better than the classical walk.

For a better understanding of the performance of the reset quantum walk with reference to changing reset rates (r) and measurement times (τ), we can plot (Figure 4.3b)[19] the mean first hitting time versus these parameters.

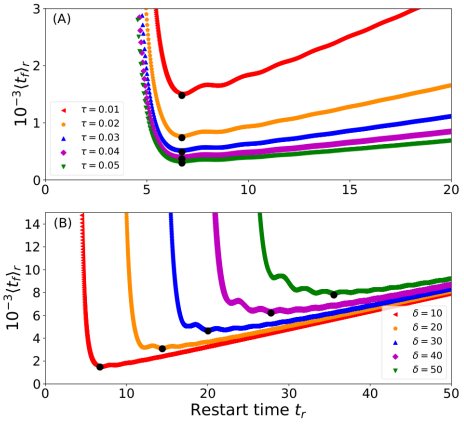
4.2.1. Observations

- Deterministic restart leads to zero asymptomatic failure rate
- Eager restarting leads to walker never reaching δ, reducing success rates drastically
- Cautious restart reduces the effect of restart, reducing success rates.
- There exists an optimal r, but this needs to be optimized, which is nontrivial for general τ and graph structures.

> **i** Can stochastic restarting be better than sharp reset?
>
> No, because even if $\langle r \rangle = r_{\text{optimal}}$, $\langle t_f \rangle > \langle t_f \rangle_{\text{optimal}}$ due to the non-monotonic nature of the curve



(a) Reset Success probability vs Time[19]. See that picking the correct r value can make the quantum walk outperform even the optimally reset classical walk.



(b) Effect of r and τ on mean hitting time[19]. See that very small and very large values of r lead to poor performance of the walk

Figure 4.3.: Sharp Reset Continuous Time Quantum Walk

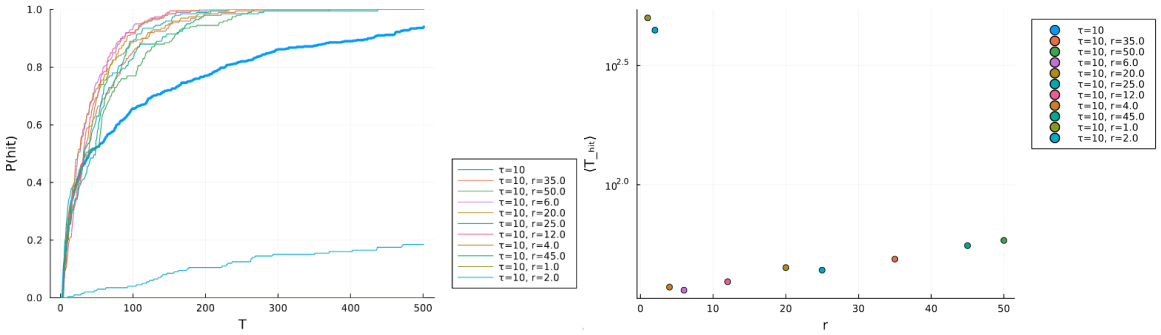
4.3. Sharp Reset for Discrete time walks

For discrete time walks, the sharp reset walk is given by

$$|\psi_t\rangle = U^{t-r_l\tau}(|c_{\text{init}}\rangle \otimes |0\rangle)$$

where $r_l\tau$ was the time of last reset.

Equivalently, in the circuit model, the reset circuit is considered to be a measure followed by post process where we apply the appropriate unitary rotation to rotate to $|\psi_0\rangle$.

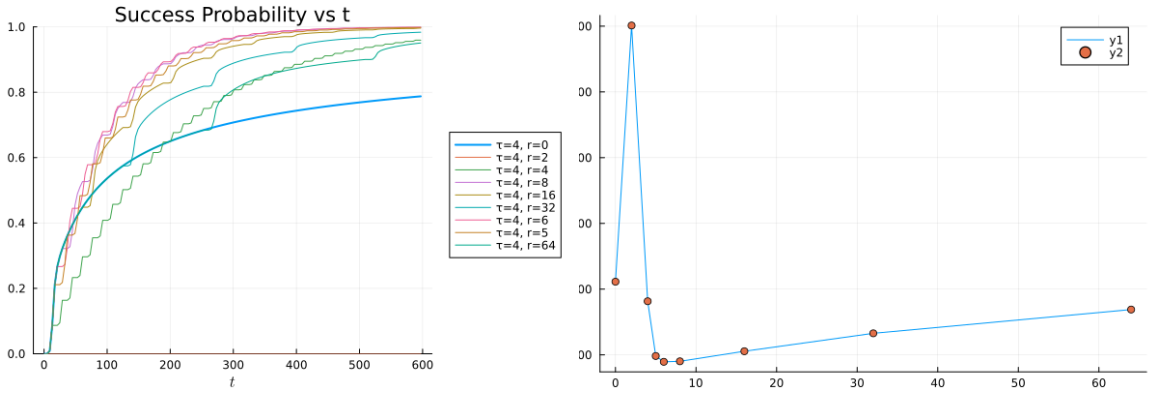


(a) P(hit) vs Time

(b) $\langle T_{hit}\rangle$ vs γ

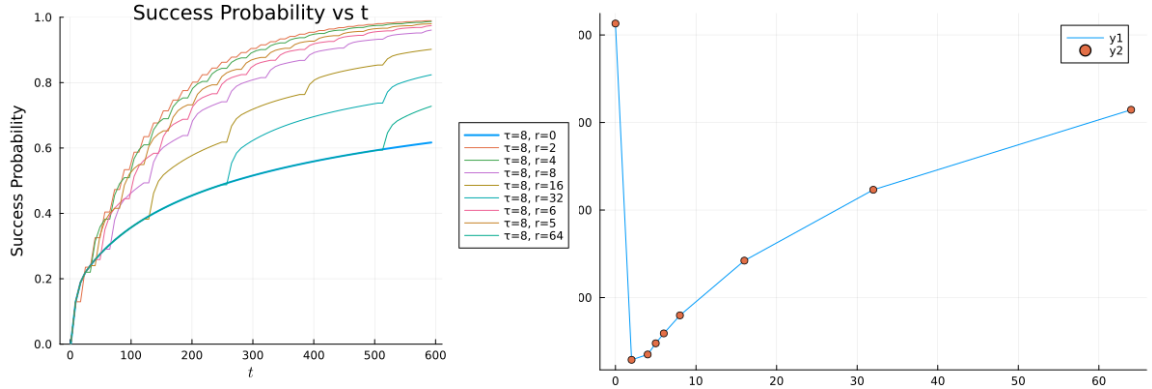
Figure 4.4.: Sharp Reset Quantum Walk - Multiple Sampling. Walk was performed by simulating the circuit formalism with $\delta = 10$ on a cycle of size of 256 nodes. Note that the success curves and the dependence of mean hitting time is similar to the continuous case (Figure 4.3)

Our results are plotted in Figure 4.4 where the circuit formalism is run multiple times, and in Figure 4.5, we pick the diagonal term from the matrix formalism as the infinite



(a) P(hit) vs Time $\tau = 4$

(b) $\langle T_{hit} \rangle$ vs γ $\tau = 4$



(c) P(hit) vs Time $\tau = 8$

(d) $\langle T_{hit} \rangle$ vs γ $\tau = 8$

Figure 4.5.: Sharp Reset Quantum Walk - Smooth Sampling. Walk was performed by simulating the matrix formalism with a dephasing operation to simulate measurement, and picking the diagonal term of the density matrix as the probability of hitting the target at every measurement event. **Note particularly that the $r = 0$ corresponds to a no reset case, and this value is simply an effect of the way it was coded.** $\delta = 10, n = 512$. Note that the success curves and the dependence of mean hitting time is similar to the continuous case (Figure 4.3)

limit. Note the similarity between the continuous and discrete curves. However, also note the difference between the non reset curve, where in the discrete case, success probability still reaches 1, this can be attributed, as before, to the finiteness of the walk space.

4.4. The Problem in the Solution

As discussed before, reset rates which are very high or low can end up being detrimental to the success times of the walk. Secondly, there is no clear path as to how to optimize the reset parameter for arbitrary graph structures.

A more fundamental problem with this protocol is the requirement of measuring at intermediate steps, and resetting accordingly. This measurement leads to a complete loss in coherence of the quantum walk. Thus, while the walk is fully quantum between measurements, measurement leads to a semi-quantum walk corresponding to a slowdown. Thus, fully quantum protocols such as the Grover search may not directly benefit from this resetting protocol. Therefore, we require a quantum resetting protocol as compared to the classical resetting mentioned here.

5. Quantum Resetting by Superposition

From the previous discussion, it seems fruitful to consider a quantum reset of quantum walks. Just as we harnessed the power of quantum superposition to speed up the walk, can we similarly have a superposition between the reset and evolution to increase the efficiency of hitting a node? The concept is similar to that of the quantisation of the classical walk: let the resetting occur on probability amplitudes rather than probabilities, allowing for interference in areas we do not want the walker to be on. In this case however, the superposition is on the level of operations, not the walker position directly.

Our first protocol is motivated by the quantum resetting of quantum systems introduced in Anubhav Srivastava's[24] master thesis, where a similar formalism was applied to a qubit system. The faster convergence of the quantum reset system acts as the primary indicator that such a speed-up may also be visible in quantum walks.

5.1. Formalism

On a finite 1D chain of length $2N+1$, define the following -

- Reset operation - $\mathcal{R}$ by the Kraus operators $\{\mathcal{R}_i = |r_0\rangle\langle i|\}_{i\in[-N,N]}$ on $\mathcal{H}_W$
- Evolve operation - $\mathcal{U}$ by the unitary operation $S \circ H$ on $\mathcal{H}_C \otimes \mathcal{H}_W$
- Attach another two level coin - states denoted by $|0\rangle$ and $|1\rangle$. Resulting state lies in $\mathcal{H}_R \otimes \mathcal{H}_C \otimes \mathcal{H}_W$
- Controlled reset operation - $\mathcal{E}$ by the Kraus operators $\{\mathcal{E}_i = |0\rangle\langle 0| \otimes I_2 \otimes \mathcal{R}_i + \frac{1}{\sqrt{N}}|1\rangle\langle 1| \otimes \mathcal{U}\}_{i\in[-N,N]}$. See Appendix C for proof that this set of Kraus operators represents a CPTP map.
- A reset coin operator $\Gamma(\gamma) = \begin{bmatrix} \sqrt{1-\gamma} & \sqrt{\gamma} \\ \sqrt{\gamma} & -\sqrt{1-\gamma} \end{bmatrix}$ on $\mathcal{H}_R$

Quantum Reset

One step of the resulting walk is defined as

$$\mathcal{E} \circ (\Gamma \otimes I_2 \otimes I_{2N+1})$$

Once again, the hitting protocol is found by measuring the walker position after τ time steps.

Although the protocol is similar to the motivation, the current problem we are considering of the first hit time has drastically changed the methods of exploration and the property we want to optimize. Where previously we only probed the convergence time, here we also want to maximize the probability of hitting the target node.

5.1.1. Interpretation of γ and dependence on initial condition of the coin

In the stochastic resetting γ is understood as the "rate" of resetting. However, this is no longer accurate for the quantum case. Consider the case for $\gamma = 0$. Then, the Gammamard operator $\Gamma = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$. This does not automatically imply that the reset never occurs; we also require that the initial state of the reset coin is $|0\rangle$. If the initial state of the reset coin is $|1\rangle$, this corresponds to the always reset case. Thus, γ is better understood as the probability that given the last step was a Reset, what is the probability this step is an Evolve, and vice versa. Thus, it is not immediately obvious how we can compare the reset mechanisms for a given γ. Nor is it obvious how the initial state of the reset coin finally affects the success probability and such, and needs to be numerically checked.

5.1.2. Resetting of the walker coin

In our specific formalism (Section 5.1), we have only applied the reset operation on the walker $\mathcal{H}_W$. One could also reset the walker coin $\mathcal{H}_C$ and see what happens. In our current formalism, there is no entanglement broken between the two coins, but there is no a priori understanding of how this may (or may not) affect the walk.

5.2. Computational Implementation

Implementations of $\mathcal{U}, U, H$ follow as before. The $\mathcal{E}$ operation is implemented as

```
function €(n, r₀)
    ks = map(1:n) do i
        (1/√n * [1 0; 0 0] ⊗ (S(n)*H(n))) +
        [0 0; 0 1] ⊗ I(2) ⊗ real(ket(r₀,n)*bra(i,n))
    end
    return KrausOperators(sparse.(ks))
end;
```

The gammamard operation Γ is defined as

```
Γ(γ) = [
    √(1-γ)  √γ;
    √γ      -√(1-γ)
```

```
];
```

We take the initial reset coin to be $(S(-\pi/2) \circ \Gamma(\gamma))|0\rangle$ for the same reason as in the quantum walk

```
reset_init(γ) = [1 0; 0 -1im]*gammamard(γ)*ket(1, 2);
```

5.3. Results

5.3.1. What the walk looks like

For the specific choice of initial reset coin as $1/\sqrt{2}\,(|0\rangle - i|1\rangle)$ and $\gamma = 0.2$, the quantum reset walk looks like Figure 5.1.

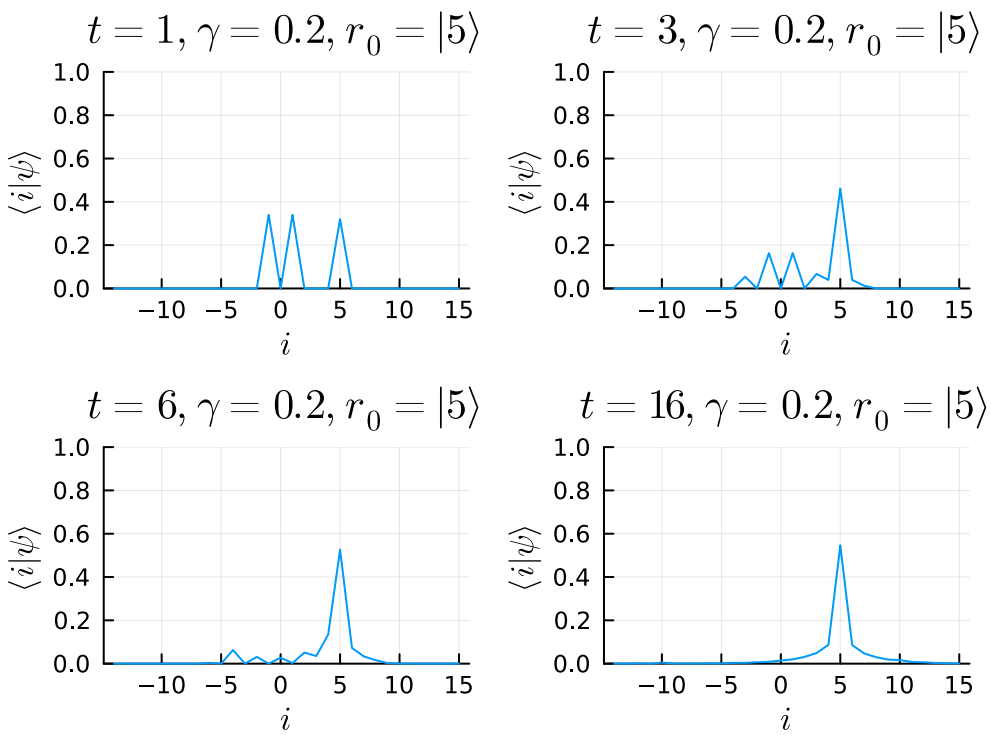
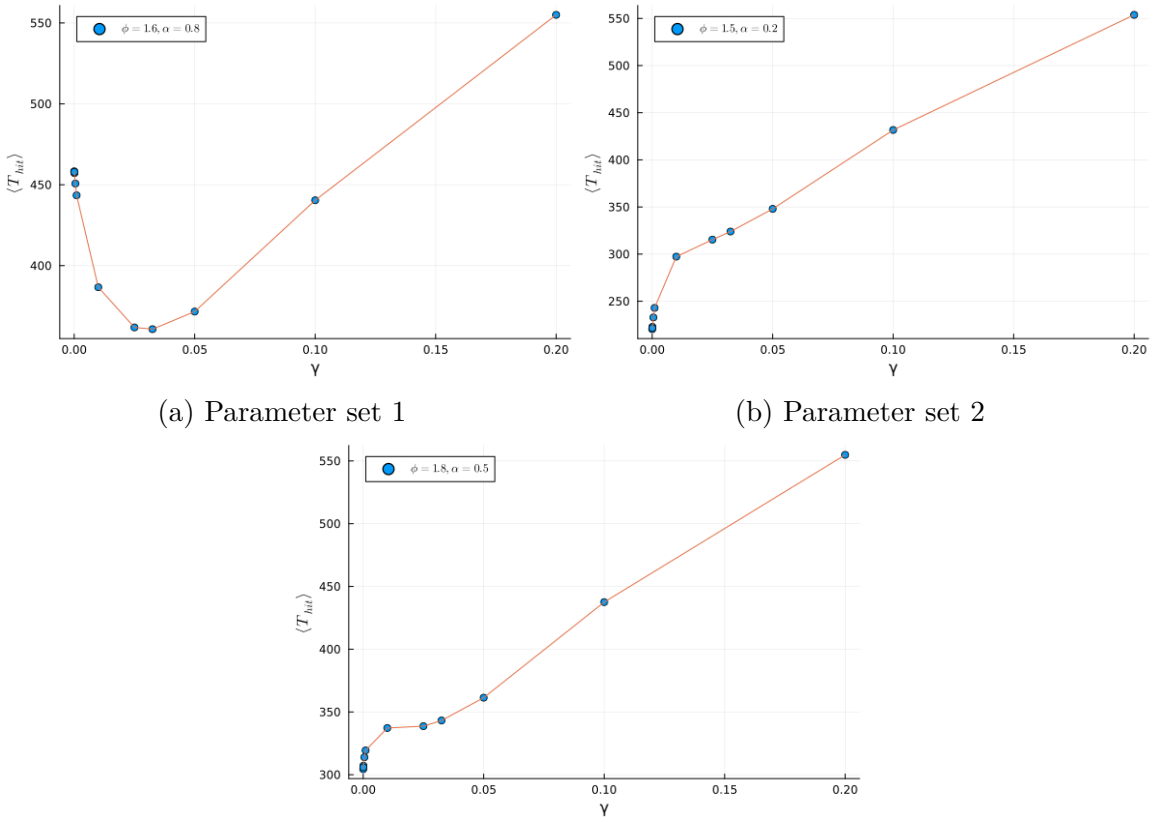


Figure 5.1.: Probability distribution of the walker in Quantum Reset Walk. We see a markedly different stable distribution as compared to the stochastic reset case Figure 4.2, with a much sharper cusp, but also a much more spread out walker.

5.4. Results and Discussion

As discussed in Section 5.1.1, we need to vary the three parameters, (α, ϕ, γ), and see what happens. Thus, we plot the success curves for a few sets of parameters

(Figure 5.2), and find the minimum mean time of hit for each fixed parameter (Figure 5.3).



(a) Parameter set 1

(b) Parameter set 2

(c) Parameter set 3

Figure 5.2.: Mean hitting time for some parameter sets. We see a nontrivial dependence on the initial state of the coin.

As we can see, the mean time of hit even for the best set of parameters (within the ranges explored) is not better than the optimal time for the stochastic reset protocol. This result however may not hold true for other values of τ, and more work is necessary to conclusively state that there isn't (or is) any speedup achieved here.

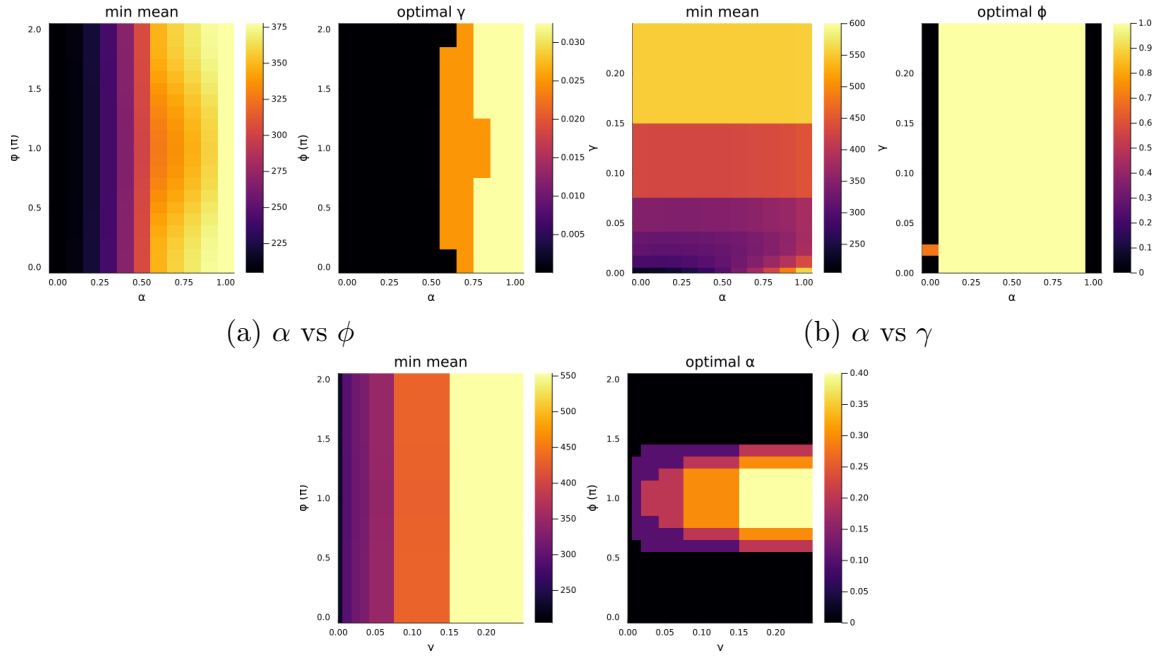
5.4.1. Optimal Parameter Values

Table 5.1.: Optimal Parameter Values

α	ϕ	γ
0.0	0.0	10^{-5}

5.4.2. The Issue of Non-Unitarity

One drawback of this formalism is its complexity. Whereas the stochastic reset proceeded by a unitary evolution between measurement events, the quantum reset protocol is inherently a non-unitary operation. This makes analysis more complicated.



(a) α vs ϕ

(b) α vs γ

(c) ϕ vs γ

Figure 5.3.: Value of the third parameter for optimal mean hitting rate, found by fixing the other two parameters. We surprisingly see that $\gamma = 0$ or 1 is the optimal value for all initial conditions. Furthermore, the effect of ϕ is symmetric about π

The difficulty doesn't only lie in analytical complexity, but also in simulations, where simulating the quantum reset quantum walk takes much longer. This can be attributed to the larger system size (size of $\rho = 16N^2$), and due to the increased number of matrix multiplications and additions ($2N$ matrix multiplications of size $4N \times 4N$ ($64N^3$ multiplications and $64N^3 - 16N^2$ additions) followed by N matrix additions ($16N^2 additions) which finally consists of $128N^4$ multiplications and $128N^4$ additions for one step of the quantum reset walk, as compared to $2 \cdot 8N^3$ multiplications and $2 \cdot 8N^3 - 4N^2$ additions, even ignoring the additions, the quantum reset walk is $\mathcal{O}(N^4)$ whereas the stochastic reset walk is $\mathcal{O}(N^3)$ to simulate. For any reasonably sized cycle (to avoid finite size effects), this cost makes simulating the walk for different parameters extremely tedious.

Therefore, although we have been able to introduce a superposition between the two operations, and we have decoupled the resetting from the measurement, we require a unitary protocol to efficiently perform the walk.

6. Quantum Markov Chains

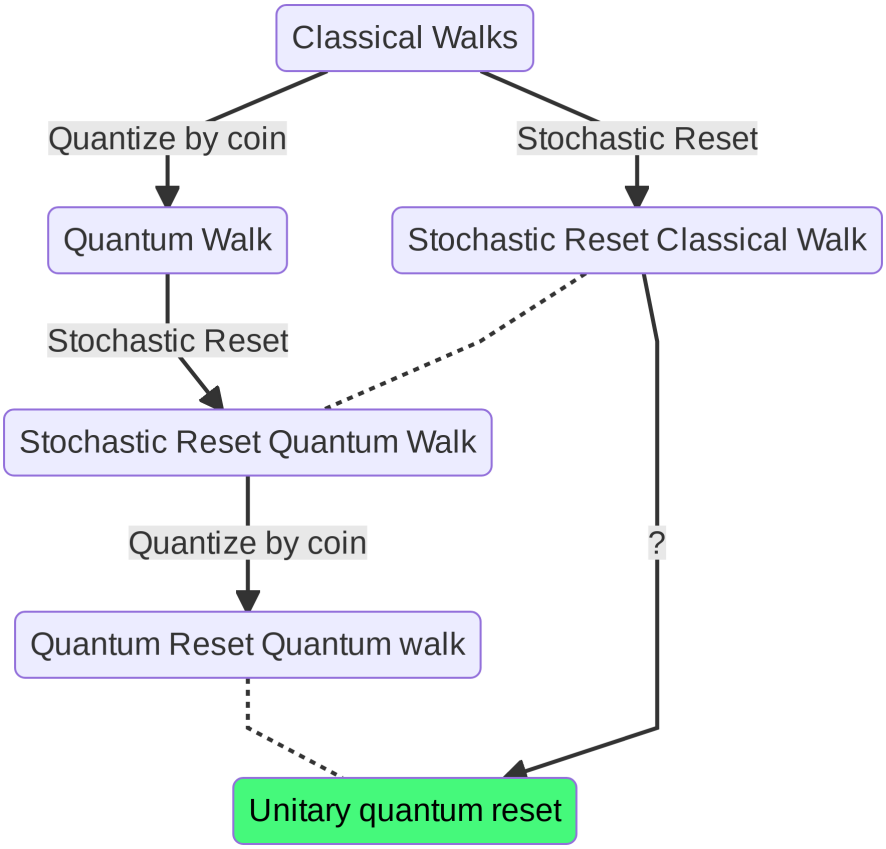


Figure 6.1.: Work until now: Our initial proposal to combat the transient nature of the quantum walk was by following a semi-quantum stochastic reset protocol. To further quantise the resetting mechanism, we propose a coined reset formalism. This proved to be too complex, prompting a unitary reset walk, which we shall introduce now.

As suggested in the earlier chapter, we would like to define a unitary quantum resetting protocol unitary for analytical and interpretational simplicity. For this, we reconsider our path until now (Figure 8.1). Instead of quantising the stochastic reset quantum walk via a coin, can we quantise the stochastic reset classical walk directly?

6.1. Generalized Coined walks

When we quantised walks in Chapter 2, we attached a 2 level coin whose states represented the selected edge, and by putting the coin into superposition, one can apply a superposition of jumps. We would like to quantise walks on arbitrary undirected graphs using the coined walk formalism.

Graphs can be partitioned into two classes, Class 1 and 2, based of the chromaticity of the graph.

> **Edge Chromaticity**
>
> The edge-chromaticity of a graph is the minimum number of colors that the edges can be colored with such that no two adjacent edges are similarly colored and is denoted by $\rho(G)$.

By Vizing's theorem[25], $\Delta(G) \leq \rho(G) \leq \Delta(G) + 1$, where $\Delta(G)$ is the max degree of the graph.

Graphs of Class 1 are those where $\Delta(G) = \rho(G)$ and graphs of Class 2 are the others. Walks on Class 1 graphs can be quantised by the coin - position formalism similar to the one introduced in Chapter 2, whereas walks on Class 2 graphs can be quantised by a walk on the edges rather than the nodes[18]. However, we're still restricted to undirected graphs.

6.2. Szegedy Walks

It is known that discrete time Markov chains do not naturally quantise via the coin formalism[26]. Szegedy[26] came up with a formalism to quantise symmetric irreducible Markov chains by using the bipartite double cover of the underlying graph (Figure 6.2). This was then generalized to ergodic chains by Magniez et al.[27]. What follows is a brief introduction to the generalized Szegedy walks.

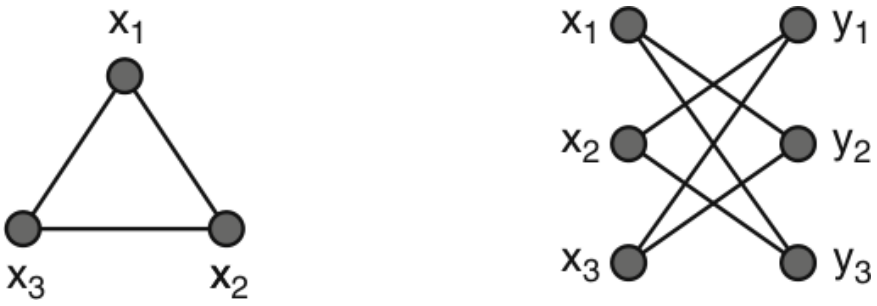


Figure 6.2.: Duplication Process on a graph[18]. The nodes are copied, and edges are drawn between the two sets if the pair of nodes had a connecting edge in the original graph.

6.2.1. Formalism

Let P be the transition matrix of a reversible Markov chain. Let P^* be the time reversed Markov chain of P.

For a state $|\psi\rangle \in \mathcal{H}$, let $\Pi_\psi = |\psi\rangle\langle\psi|$. For a subspace $\mathcal{K}$ of $\mathcal{H}$ spanned by a set of mutually orthogonal states $\{|\psi_i\rangle : i \in I\}$, let $\Pi_\mathcal{K} = \sum_{i\in I} \Pi_{\psi_i}$ be the projector onto $\mathcal{K}$ and $\mathcal{R}_\mathcal{K} = 2\Pi_\mathcal{K} - \mathrm{Id}$ be a reflection through $\mathcal{K}$.

Let $\mathcal{A} = \mathrm{Span}(|x\rangle|p_x\rangle : x \in X)$ and $\mathcal{B} = \mathrm{Span}(|p^*_y\rangle|y\rangle : y \in Y)$ be subspaces of $\mathcal{H} = \mathbb{C}^{|X|\times|X|}$, where

$$|p_x\rangle = \sum_{y\in X} \sqrt{p_{xy}}|y\rangle; |p^*_y\rangle = \sum_{x\in X} \sqrt{p^*_{yx}}|x\rangle$$

where p_{ij}, p^*_{ij} are elements of P, P^* respectively, and X is the set of nodes, Y is the set of nodes after duplication.

Quantum Markov Chain

The quantised version of the Markov chain P is defined to be the unitary operation $W(P) = \mathcal{R}_\mathcal{B}\mathcal{R}_\mathcal{A}$ and is called the Szegedy walk. Where defined, the Szegedy walk is equivalent to two steps of the Coined quantum walk[28].

The Discriminant matrix

For an ergodic Markov chain P with stable distribution π, we define

$$D(P) = \mathrm{diag}(\pi)^{1/2} \cdot P \cdot \mathrm{diag}(\pi)^{-1/2}$$

as the discriminant matrix.

6.2.2. Properties

1. On $\mathcal{A} + \mathcal{B}$, eigenvalues of $W(P)$ that have non-zero imaginary part are $e^{\pm 2i\theta_1}, \ldots, e^{\pm 2i\theta_l}$, with same multiplicity.
2. On $\mathcal{A} \cap \mathcal{B}$, $W(P)$ acts as the identity. The left (and right) singular vectors of D with singular value 1 span this space.
3. On $\mathcal{A} \cap \mathcal{B}^\perp$ and $\mathcal{A}^\perp \cap \mathcal{B}$, the operator acts as $-$Id. The $\mathcal{A} \cap \mathcal{B}^\perp$ (resp. $\mathcal{A}^\perp \cap \mathcal{B}$), is spanned by the set of left (resp. right) singular vectors of D
4. W(P) has no other eigenvalues on $\mathcal{A} + \mathcal{B}$; on $\mathcal{A}^\perp \cap \mathcal{B}^\perp$ it acts as Id

6.3. Szegedy Search or the Quantum Hitting Time

Of the many ways to define the search algorithm (refer to Chapter 3 for an introduction) for Szegedy walks, we shall look at two protocols, one which is easy to understand

and the other easy to analyse. The original paper by Szegedy[26] proposed the following protocol -

1. Modify the classical Markov chain to make all the marked vertices sinks (Figure 6.3).

$$p'_{xy} = \begin{cases} p_{xy}, & x \notin G \\ \delta_{xy}, & x \in G \end{cases}$$

2. Define $W'(P)$ as the quantum Markov chain by the usual protocol.
3. Define the initial state

$$|\psi(0)\rangle = \frac{1}{\sqrt{n}} \sum_{\substack{x \in X \\ y \in Y}} \sqrt{p_{xy}} |x\rangle |y\rangle$$

4. Finally, define the quantum hitting time such that

$$F(T) \geq 1 - \frac{g}{n}$$

where

$$F(T) = \frac{1}{T+1} \sum_{t=0}^{T} \Big\| |\psi(t)\rangle - |\psi(0)\rangle \Big\|^2 ; g = |G|$$

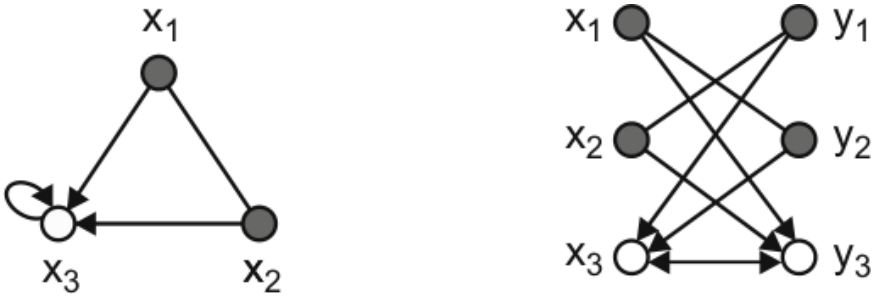


Figure 6.3.: Duplication Process of a graph with marked vertex 3[18]. All edges going outwards from x_3 and y_3 are broken, making the graph directed

Defined by this protocol, the quantum hitting time is quadratically smaller than the classical hitting time[26] for the 1D case.

A later modification[27] defined the search operation via a Grover-like oracle. This was easier to analyse, and the connection between the spectral gap of the discriminant matrix (Section 6.2.1) and the quadratic speedup attained by the walk is clearer. We leave the exact protocol to reference[27], but only outline the steps.

1. Prepare the initial state $|\pi\rangle|0^T ks\rangle$, where

$$|\pi\rangle_d = \sum_{x \in X} \sqrt{\pi_x} |x\rangle |p_x\rangle = \sum_{y \in X} \sqrt{\pi_x} |x\rangle |p_x\rangle$$

2. First apply the Grover oracle

$$\mathcal{G}(|x\rangle_d|y\rangle_d|z\rangle) = \begin{cases} -|x\rangle_d|y\rangle_d|z\rangle, & \text{if } x \in G \\ +|x\rangle_d|y\rangle_d|z\rangle, & \text{otherwise} \end{cases}$$

3. Apply a phase estimation circuit to the quantum walk, repeated k times.
4. Repeat steps 2 and 3 T times.
5. Observe the first register, by a projective measurement in the computational basis. Denote by $\bar{x}$
6. With high probability, output $\bar{x}$ lies in G

If the eigenvalue gap of the Markov chain is δ, and $\frac{|G|}{N} \geq \epsilon \geq 0$, the cost to perform this circuit is of order $\left(\frac{1}{\sqrt{\epsilon\delta}} \log \frac{1}{\sqrt{\epsilon}}\right)$ calls to the walk operation[27]. Contrast this with the classical search which requires $\frac{1}{\delta\epsilon}$ steps of the Markov walk, we see the quadratic speedup [1].

Therefore, one can find the spectral gap δ of P and compare the search speed of two chains. This Szegedy formalism beautifully sets up the stage for a truly unitary quantum reset quantum walk protocol which can be analysed for any graph structure without having to resort to simulation techniques.

[1]Technically, this is not a quadratic speedup due to the extra $\log \frac{1}{\epsilon}$ term, and it was only later shown[29] using a protocol of eigenvalue estimation to hold true for all ergodic chains with any number of marked vertices

7. Unitary Quantum Reset Quantum Walk

Armed with the Szegedy formalism from the previous chapter, we can define the unitary quantum reset quantum walk by quantising the stochastic reset classical walk (Section 4.1.2). It is obvious that the stochastic reset classical walk is not symmetric ($p_{i0} \neq p_{0i} \forall i \neq 0$), so we cannot use the original Szegedy walk, but we can use the generalized Szegedy formalism if we show that our process is ergodic.

7.1. Ergodicity of the Stochastic Reset Classical Walk

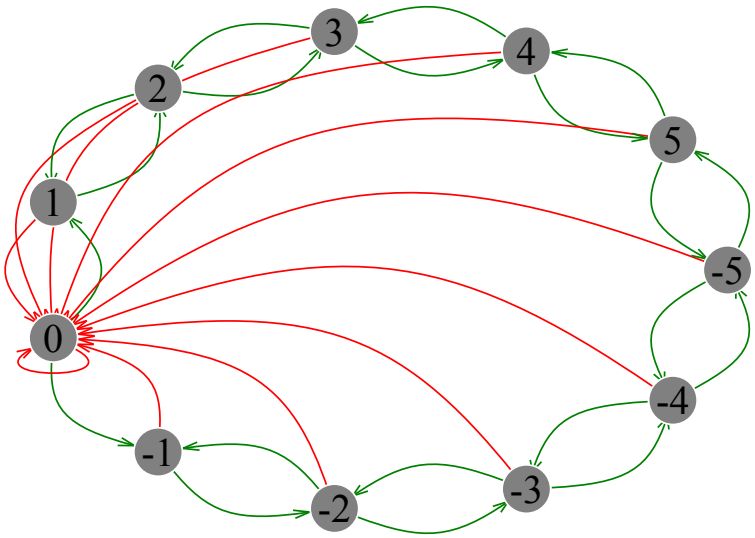


Figure 7.1.: Graph of the Stochastic Reset Classical Walk. Green edges represent walk edges, and red edges represent reset edges.

To show ergodicity, we need to show that the walk is irreducible, aperiodic and positive recurrent.

Irreducibility follows from the irreducibility of the 1D walk (Appendix A) $\forall \gamma < 1$. For the trivial case of $\gamma = 1$, the chain is not irreducible, and we cannot use the Szegedy formalism.

Aperiodicity (Appendix A) of the chain follows from the irreducibility of the chain and aperiodicity of node 0 which is obvious due to the existence of the self loop.

Recurrence is obvious from $p_{00}^{(n)} \geq \gamma \forall i$ and irreducibility. Positive recurrence is harder to show, but we can solve the recurrence relation (Chapter 5).

Thus, the stochastic reset classical walk is ergodic and is a viable candidate for quantisation via the Szegedy walk.

7.2. Implementation and Results

The implementation of the Szegedy walk is simplified by the use of `QuantumWalk.jl`[30] and `LightGraphs.jl`[31]

First we define a function that returns the underlying graph and the transition matrix

```
MyGraph(n, γ=0.5) =
    let
        temp = CycleGraph(n) ▷ adjacency_matrix ▷ collect
        reset = zero(temp)
        reset[:, n÷2] .= 1
        stochastic = ((1 - γ) / 2) * temp + γ * reset
        temp = sign.(temp + reset)
        DiGraph(temp), sparse(transpose(stochastic))
    end
```

Then we define and run the search algorithm.

```
function run_search_mygraph(n, δ, T, γ)
    graph, stochastic = MyGraph(n, γ)
    qwe = QWSearch(Szegedy(graph, stochastic), [n÷2 + δ])
    first.(measure.(Ref(qwe), execute_all(qwe, T), Ref([n÷2 + δ])))
end
```

`QuantumWalk.jl` uses the Grover-like search algorithm[28].

```
struct Simulation
    n::Int64
    δ::Int64
    T::Int64
    γ::Float64
    data::Vector{Float64}
    function Simulation(n, δ, T, γ)
        new(n, δ, T, γ, run_search_mygraph(n, δ, T, γ))
    end
end

mean_hit(s::Simulation) = mean(0:s.T, Weights(success(s)))
success(s::Simulation) = accumulate(s.data, init=0) do old, curr
```

```
    (1-old) * curr + old
end
```

We can scan over $\gamma \in [0, 1]$ and plot the instantaneous probability of measuring in δ, success probability and the mean hitting time in Figure 7.2.

> **Alternative Definition of the Mean Hitting Time**
>
> Although the definition of mean hitting time we have defined until now is only valid for a measure and continue semi-quantum walk, we adopt a similar definition for this walk, only without actual measurements. In this section, (particularly Figure 7.2 and Figure 7.3), we use a measure and rerun protocol, and define the success probability S_n on the basis of F_i of the unmeasured quantum walk until each time $i < n$

```
res = map(0:0.01:1) do γ
    Simulation.(100, 10, 100, γ)
end;
```

7.3. Results

7.3.1. Exceptional Quantum Walk on the Cycle

For the non reset ($\gamma = 0$) case, we see that the success rate does not increase with time. While this is initially surprising, this is a known result[32], caused due to the fact that the initial state evolves by phase flips, and the amplitude does not increase.

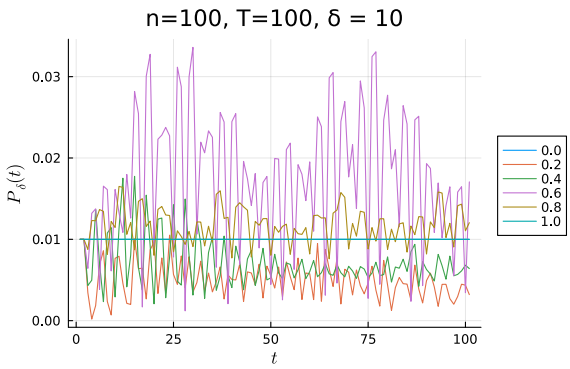
In the reset case, we see that the walk is no longer exceptional (at least, not in this sense), which allows for an amplitude amplification on the marked node.

7.3.2. Mean hitting time vs γ

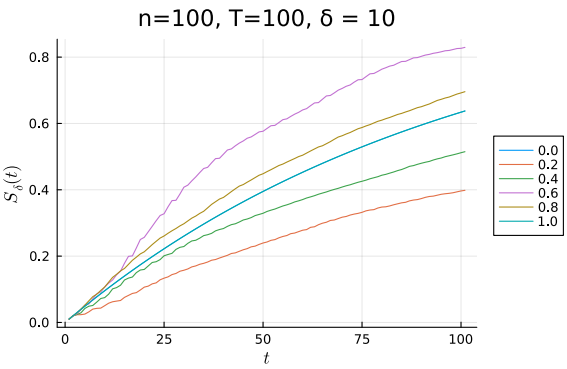
Plotting the mean hitting time (see previous callout on the alternative definition) versus γ for 500 nodes and 500 steps, we get the curve in Figure 7.3.

As we can see, there is a clear non-monotonous effect of γ on the mean hitting time of the walk. For values a bit more than $\gamma = 0.5$, we see that the mean hitting time is much lower than that for the no reset $\gamma = 0$ case. Also of note is that as $\gamma \to 1$ the mean hitting rate approaches that of the no reset case.

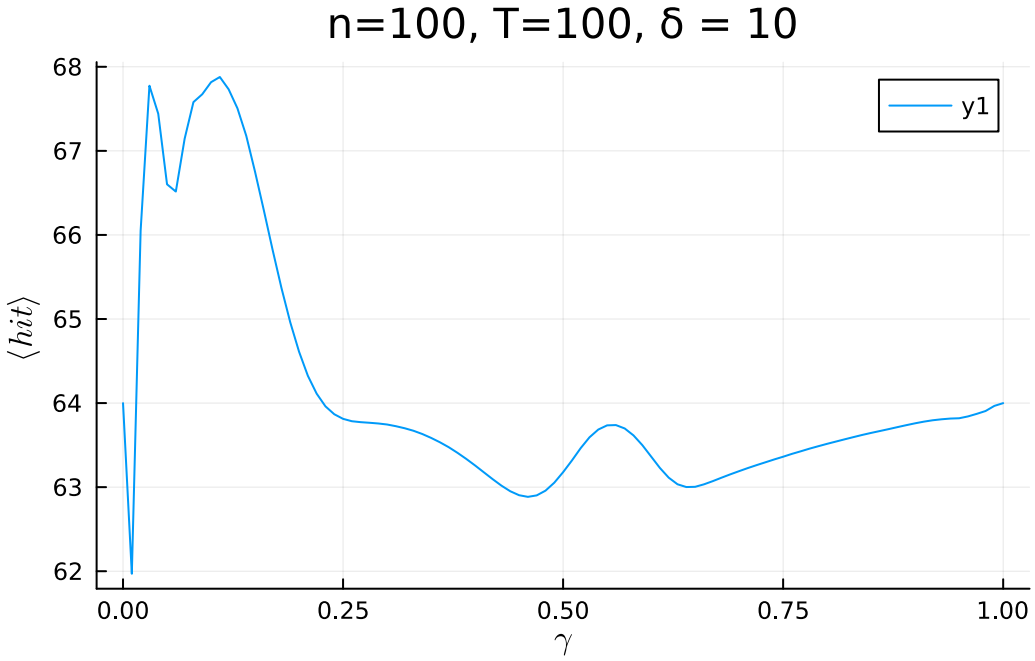
Numerically, the exact time for mean hit changes according to the amount of time that we run the simulation for. This is obvious from the way that the mean hit time is calculated, where if the walk has not succeeded within the time of running the simulation, we assume that it succeeds in the next step. However, we recover the same qualitative curve for large number of steps.



(a) Probability of measuring in δ



(b) Success Probability



(c) Mean hit time vs time

Figure 7.2.: Unitary Quantum Reset. These results are for a walk on $n = 100, \delta = 10, T_{max} = 100$. This is a very small state space, run for a short time, so results should only be considered qualitatively, but we can still see the effect of unitary resetting on the success probability.

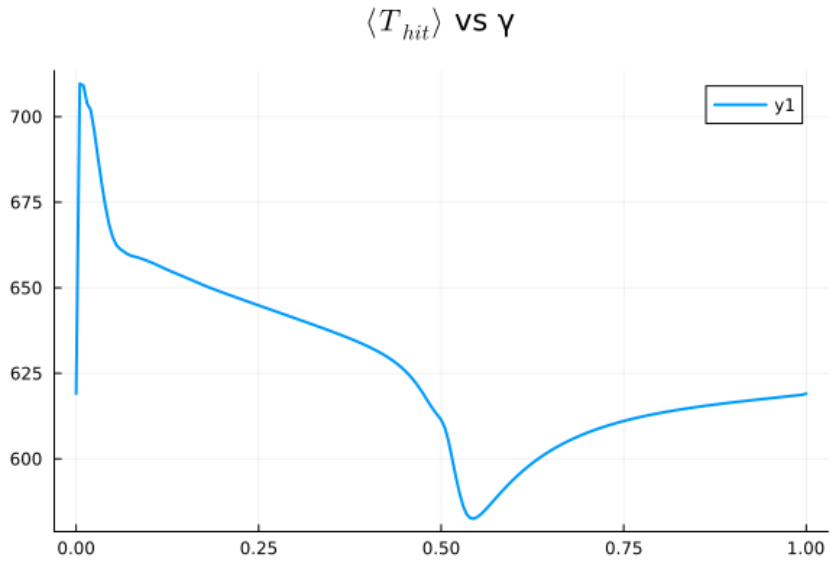


Figure 7.3.: Mean hitting time vs γ for the Unitary Quantum Reset. We observe a non-monotonic curve for the mean hitting time, and a speed-up compared to the non-reset case. This was run for $n = 500, \delta = 50, T_{max} = 1000$

7.3.3. Eigenvalue analysis

In Section 6.3, we saw how the quadratic increase in the eigen-gap (ΔP) leads to an associated speedup in the search problem. Thus, if we can show that the eigen-gap increases in for the reset case, we can show that the search protocol requires fewer steps to complete. Furthermore, the space cost of the circuit goes as $\lceil \log_2 \left(\frac{2\pi}{\Delta P} \right) \rceil$

We see that the transition matrix is extremely sparse, especially for larger system sizes (goes as $\mathcal{O}(3N)$, so the density goes as $\mathcal{O}(1/N)$), and that we do not require the entire eigen spectrum to find the eigen-gap. Thus, we can use specialized methods such as the Arnoldi method provided in Julia by the `KrylovKit.jl`[33] package.

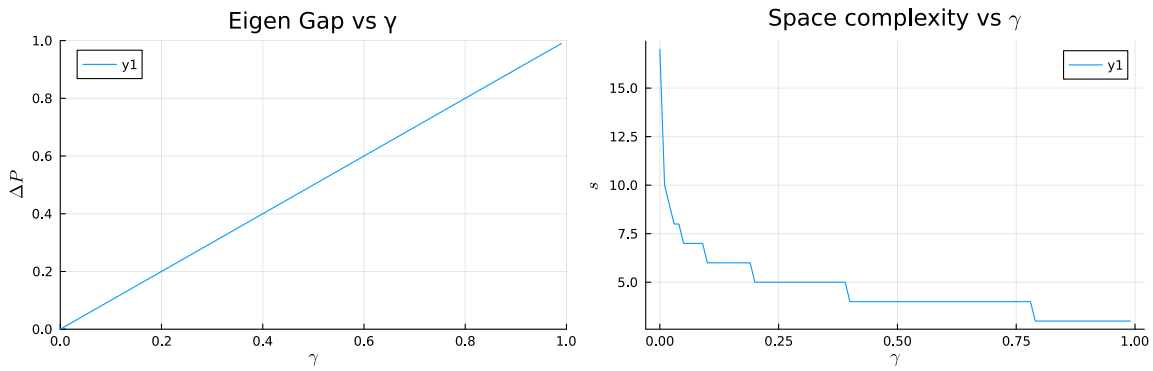
```
using KrylovKit

function eigen_gap(M)
    v = real(eigsolve(M, 2, :LR)[1]) # get real(eigval)
    v[1] - v[2]
end

γs = 0.:0.01:(1-0.01)

ΔP = eigen_gap.(getindex.(MyGraph.(512, γs), 2))
```

Plotting ΔP vs γ in Figure 7.4, we see that increasing γ leads to an equal increase in ΔP, which corresponds to a faster convergence under the Grover-like search protocol. Note however that we do not have a quantitative characterization of what the final probability of success is. Therefore, despite the faster convergence, the non-monotonic nature of the mean hitting time against γ can be explained by a loss in final success



(a) Eigen Gap (b) Space complexity

Figure 7.4.: Analysis by Eigen Gap of the Reset Walk. See that the eigengap increases with γ, and therefore, there is a corresponding decrease in the space requirement of the protocol. $n = 500, \delta = 50, T_{max} = 1000$

probability. The interplay of these two effects may be the cause of the existence of an optimal reset parameter.

7.4. Discussions

A particular advantage of the Unitary quantum reset is that we are not limited to only resets on 1D chains. We can define the unitary reset walk on any irreducible graph that one can come up with, even periodic graphs, since resetting a periodic graph leads to an aperiodic graph (Appendix D), and the corresponding walk can be quantised by the Szegedy formalism.

While it is clear that there is an obvious speedup compared to the walk without reset, we cannot directly compare these results with the stochastic reset quantum walk due to the different definitions of F_n. The quantification of the quantum hitting time for Szegedy walks is an area of very active research[34]. Another possible approach would be via the semi-quantum Szegedy walk[20]. These methods have been left as a possible future work.

8. Summary and Future directions

See the following schematic as a summary of our work, with the highlighted sections as new work.

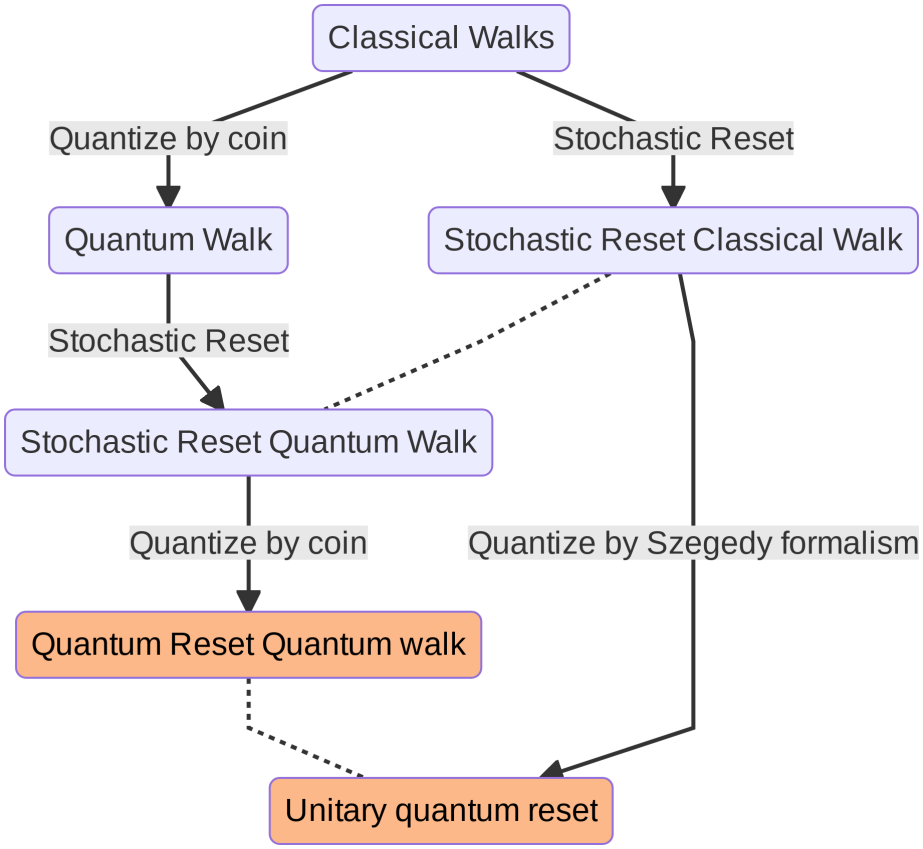


Figure 8.1.: A summary of the thesis

We have reviewed the definitions of classical and quantum walks, and a measurement based node hitting protocol. We further reviewed past work which fixed the transient nature of the semi-quantum walk by a classical (possibly stochastic) resetting protocol, leading to 0 asymptotic failure rates. Furthermore, we motivated the need for a true quantum resetting protocol.

We first proposed a quantum resetting protocol motivated by the superposition of the shift operations in the coined quantum walk, and produced a preliminary computational analysis. While we were not able to show conclusively an advantage over the classical reset, there is still a possibility that a speedup can be achieved for other

ranges of parameters. This protocol however became unwieldy to handle due to its complexity, and suggested a need for a unitary quantum reset.

We then proposed a unitary quantum reset by directly quantising the classical reset classical walk via the Szegedy walk formalism, which was also introduced. By running a Grover-like search protocol, we were able to show that the unitary reset quantum walk on the 1D chain does show a small decrease in the mean hitting time compared to the quantum walk for certain ranges of the reset parameter. Furthermore, we show that there is a definite speedup and a reduction in the computational space required in the search protocol via eigenvalue analysis, and propose that the reduction in the mean hitting time may be due to a corresponding reduction in the success of the protocol.

8.1. Future directions

We propose following questions for future work.

8.1.1. Quantum Reset

Does the quantum reset protocol show speedup even for other parameter ranges? Is there a framework to analyse the convergence time of the quantum resetting protocol and the probability of success? Can we propose a Grover-like search algorithm? What effect does the quantum resetting have on such a protocol?

8.1.2. Unitary Reset

We have left the analysis of the probability of success for the unitary quantum reset Grover-like search protocol for future work. Since the unitary quantum reset is a quantum walk, how does the semi-quantum version of this walk behave like? The unitary quantum reset protocol is not limited to the 1D chain, or even lattice like structures. Do we still see speedups for other graph structures? Finally, can we apply the classical reset to the unitary quantum reset walk to gain a speedup from both kinds of resetting?

References

[1] J.R. Norris, *Markov Chains*, 1st pbk. ed (Cambridge University Press, Cambridge, UK ; New York, 1998).
[2] P. Glasserman, *Monte Carlo Methods in Financial Engineering* (Springer New York, New York, NY, 2003).
[3] K.P. Griffin, S.J. Suresh, T.J. Flint, and W.H.R. Chan, (2019).
[4] D. Psaltis, F. Özel, L. Medeiros, P. Christian, J. Kim, C. Chan, L.J. Conway, C.A. Raithel, D. Marrone, and T.R. Lauer, ApJ **928**, 55 (2022).
[5] N. Metropolis, Los Alamos Science Spl. Issue (1987).
[6] L.S. de Souza, J.H.A. de Carvalho, and T.A.E. Ferreira, in *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)* (IEEE, Salvador, Brazil, 2019), pp. 836–841.
[7] M. Bae and W.O. Krawec, (2021).
[8] S. Mukherjee, IEEE Trans. Quantum Eng. **3**, 1 (2022).
[9] X. Bonnetain, A. Chailloux, A. Schrottenloher, and Y. Shen, (2022).
[10] M. Goldsmith, G. García-Pérez, J. Malmi, M.A.C. Rossi, H. Saarinen, and S. Maniscalco, (2022).
[11] S.A. Ortega and M.A. Martin-Delgado, (2022).
[12] Qiskit, Quantum Walk Search Algorithm (n.d.).
[13] J. Bezanson, A. Edelman, S. Karpinski, and V.B. Shah, SIAM Rev. **59**, 65 (2017).
[14] T. Breloff, (2022).
[15] P. Gawron, D. Kurzyk, and Ł. Pawela, PLoS ONE **13**, e0209358 (2018).
[16] JuliaGraphics, (n.d.).
[17] X.-Z. Luo, J.-G. Liu, P. Zhang, and L. Wang, Quantum **4**, 341 (2020).
[18] R. Portugal, *Quantum Walks and Search Algorithms*, 2nd ed. 2018 (Springer International Publishing : Imprint: Springer, Cham, 2018).
[19] R. Yin and E. Barkai, (2022).
[20] S.A. Ortega and M.A. Martin-Delgado, (2023).
[21] M. Štefaňák, I. Jex, and T. Kiss, Physical Review Letters **100**, 020501 (2008).
[22] H. Friedman, D.A. Kessler, and E. Barkai, Phys. Rev. E **95**, 032141 (2017).
[23] user940, (n.d.).
[24] A. Srivastava, Resetting Quantum Systems Through Superposition of Evolution , PhD thesis, IISER Mohali, 2021.
[25] Wikipedia contributors, (2023).
[26] M. Szegedy, in *45th Annual IEEE Symposium on Foundations of Computer Science* (IEEE, Rome, Italy, 2004), pp. 32–41.
[27] F. Magniez, A. Nayak, J. Roland, and M. Santha, SIAM Journal on Computing **40**, 142 (2011).
[28] T.G. Wong, Quantum Information Processing **16**, 215 (2017).
[29] H. Krovi, F. Magniez, M. Ozols, and J. Roland, Algorithmica **74**, 851 (2016).
[30] A. Glos and J. Adam Miszczak, (2018).

[31] S. Bromberger and O. Contributors, (2017).
[32] T.G. Wong and R.A.M. Santos, Quantum Information Processing **16**, 154 (2017).
[33] Jutho and Contributors, (n.d.).
[34] P. Boito and G.M. Del Corso, (2023).

A. Markov Chains

The following are some definitions related to Markov chains, which can be found in any introductory textbook to the topic[1].

Definition: Communicating Nodes

States $i, j \in S$, are connected if $\exists n > 0, m > 0 | P_{ij}^n > 0$ and $P_{ji}^m > 0$. Note that this is an equivalence relation.

Definition: Closed Communicating Class

If C is such that $\sum_{j \in C} p_{ij} = 1 \forall i \in C$, then it is called a closed communicating class.

Definition: Irreducibility

If a chain has only one closed communicating class, it is called irreducable.

Definition: Recurrence

If $i \in S, \sum_{n=1}^{\infty} P_{ii}^n \to \infty$, then the state is called recurrent. Equivalently, $\sum_n F_n \to 1$ for recurrent nodes. If a chain is irreducible and one of its states is recurrent, all its states are recurrent, and thus the chain is called recurrent. If a chain is recurrent, then it has a stable measure π such that $P\pi = \pi$. If π can be normalized to a stable distribution, the chain is called null recurrent.

Definition: Stopping time

A random variable $T : \Omega \to 0, 1, ... \cup \infty$ is a stopping time the event $\{T = n\}$ depends only on $X_0, X_1, ... X_n$ for $n = 0, 1, 2, ...$

Definition: Periodicity

For a state $i \in S$, the Period is $\gcd\{n \geq 0 : p_{ii}^{(n)} > 0\}$. If the Period of the state is 1, then it is called aperiodic. For an irreducible chain, if one state is aperiodic, then all states are aperiodic, and the chain is referred to as aperiodic.
The state space of all irreducible chains can be partitioned into p disjoint sets

$$S = C_0 \cup C_1 \cup \cdots \cup C_{d-1}$$

and set $C_{nd+r} = C_r$, such that

1. $p_{ij}^{(n)} > 0$ only if $i \in C_r$ and $j \in C_{r+n}$ for some r.
2. $p_{ij}^{(nd)} > 0$ for all sufficiently large n, for all $i, j \in C_r$, for all r.

Time reversed Markov chain

For a Markov chain P with stable distribution π, P^* given by $p_{ji}^* = \dfrac{\pi_i p_{ij}}{\pi_j}$ is the transition matrix of the time reversed Markov chain.

Definition: Ergodic Markov chain

A positive recurrent, aperiodic chain is called an ergodic chain.

B. Measured Quantum Walk is Markov

Let X_i be the ith readout in the measured quantum walk protocol. This implies that immediately after the ith measurement, the state of the walker is $|X_i\rangle$. After τ steps, the walker is in state $E^\tau|X_i\rangle$. The probability of measuring the walker in X_{i+1} is given by $|\langle X_{i+1}|E^\tau X_i\rangle|^2$. Thus, $P(X_{i+1} = X_{i+1}|X_i, \ldots, X_0) = P(X_{i+1} = X_{i+1}|X_i)$, which implies that the readouts in the measured quantum walk protocol is a Markov process.

C. Quantum Reset is a CPTP map

We prove that the set of Kraus operators defined in Chapter 5 form a CPTP map. We are required to show that

$$\sum_i \mathcal{E}_i^\dagger \mathcal{E}_i = \mathbf{1}$$

We start by evaluating

$$\mathcal{E}_i^\dagger \mathcal{E}_i = |0\rangle\langle 0| \otimes I_2 \otimes \mathcal{R}_i^\dagger \mathcal{R}_i + \frac{1}{N}|1\rangle\langle 1| \otimes I_n = |0\rangle\langle 0| \otimes I_2 \otimes |i\rangle\langle i| + \frac{1}{N}|1\rangle\langle 1| \otimes I_n$$

Summing over i

$$\sum_i \mathcal{E}_i^\dagger \mathcal{E}_i = |0\rangle\langle 0| \otimes I_2 \otimes I_n + |1\rangle\langle 1| \otimes I_2 \otimes I_n = I_{4n}$$

We are done.

D. Proof of Reset Markov Chains Being Aperiodic

Given an irreducible Markov chain with transition matrix P with state space S, $s_r \in S$ and $0 < \gamma < 1$, define the **reset Markov chain to s_r w.p γ** as the Markov chain given by the transition matrix $P^* = (1-\gamma)P + \gamma R$, where $[R_{ij}] = \begin{cases} 1, & j = s_r \\ 0, & otherwise \end{cases}$ on state space S. Note that this is a well-defined irreducible Markov chain.

Let the period of P^* be d. Therefore, we can partition the state space into disjoint sets.

$$S = C_0 \cup C_1 \cup \cdots \cup C_{d-1}$$

and set $C_{nd+r} = C_r$, such that $p_{ij}^{*(n)} > 0$ only if $i \in C_r$ and $j \in C_{r+n}$ for some r. Let $s \in C_k, s \neq s_r$ for some k. Since $p^*_{ss_r} = p_{ss_r} + \gamma > 0$, s_r must lie in C_{k+1}. Since this holds for all k, there can at most be two Cs, one with s_r and the other with the rest of the nodes. Finally, if $p_{ij} > 0$ for any $i, j \in S/\{s_r\}$, they must lie in different classes, or there must be only one class. Since the former would create a contradiction, we must have that $d = 1$ and P^* is aperiodic.

If no such i, j exist, then P must be a star network. While in the current formalism, the self loop at s_r renders the proof almost trivial, we have provided a proof which shows aperiodicity even if $P*$ be modified to remove the self loop. In such a case, barring the star network, all other irreducible graphs follow this proof.