Classical and Quantum Simulations via Quantum Algorithms

By

PEDRO CONTINO DA SILVA COSTA

ADVISOR: DR. FERNANDO DE MELO

BRAZILIAN CENTER FOR RESEARCH IN PHYSICS

Thesis submitted to the Brazilian Center for Research in Physics in accordance with the requirements of the degree of DOCTOR OF PHYSICS.

MARCH 2018

Dedication and acknowledgements

These four years I pursued my doctorate were the most challenging of my life. Indeed, this path is very difficult and without the support of my advisor, collaborators, and friends, I would not be where I am today.

I would like to begin with a special thank you to my advisor, Fernando de Melo. When I started my doctorate, I had no idea what to focus my research on. Thus, it would be accurate to say that Fernando introduced me to a career in academics. I learned alongside him, and he helped me to ask more pertinent questions, as well as to think more about the problems I encountered. He went far beyond what I expected from an academic advisor. He taught me how to conduct a seminar presentation, and also helped me to improve my writing and speaking skills. Fernando also gave me the freedom to collaborate with other researchers, a detail that has proven to be vital to my career. In addition, he also was a great friend, giving me support along these fours years. It was an honor to me to be his first student.

I am also very grateful to Stephen Jordan, who gave me a special opportunity to work. I have been collaborating with Stephen almost two years. I could never learn too much with him. During my days at the University of Maryland, he welcomed me tremendously and gave me the necessary support to allow me to develop my work and also to enjoy the experience. I am also grateful to all staff, students, postdocs and researchers of the Joint Center for Quantum Information and Computer Science to all their support.

Thanks to Renato Portugal for providing me with a place to work in the National Laboratory for Scientific Computing-LNCC. During the year that I spent working in LNCC, I learned a lot from Renato and I am thankful for the opportunity to work with him.

Thanks to Bruce Sánchez for the period that we worked together and to the excellent discussions about cellular automata.

I would like to thank many friends, colleagues and professors of the Brazilian Center for Research in Physics, which was like a home to me. I especially want to thank Pedro Cavalcanti Malta, Linneu Holanda, and the professors and researchers Sebastião Alves Dias and José Abdalla Helayël for their advice and discussions.

Thanks to João Calvacanti, a great guy that gave me great support during tough times in these years.

Thanks to Joel Duberstein, my best friend overseas, for all his support during the time that I spent in Maryland, and also after I returned to Brazil.

I would like to thank Felipe Figueiredo Rocha, Ana Paula Mussel, and Juliana Menezes de Sousa to all their support.

Finally, I would like to thank my mother, Isabel Contino, for all her support and advice during my life, in particular during these last and difficult periods.

Abstract

The interest in the quantum information area is increasing fast. We can point the new quantum computers that are appearing as the main reason. People now seek to see the potentialities of these quantum machines to investigate their problems. With these new architectures, we need to develop quantum algorithms and use quantum models of computation in order to explore these new quantum devices. Going to this direction the main goals of this thesis are to do improvements in a specific the quantum method of computation, contrasting with its classical counterpart, and to develop new quantum algorithms.

A few results for classical and quantum models of computations are presented, with a particular focus on partitioned cellular automata and their quantum counterpart. We presented a new definition for this model of computation that we believe to be a more clear definition than the previous ones. Moreover, we show a general recipe to translate the main classes of quantum walks, as the coined model and the staggered quantum walk with Hamiltonian, to quantum cellular automata. We expect that this result within our new definition will increase the research activity in this area. For the classical cellular automata part, we developed coarse-graining, a tool widely explored to study emergent processes. In this part, we saw stochastic processes emerging from deterministic ones.

Still in the paradigm of discrete space and time, we propose a quantum algorithm for simulating the wave equation via Hamiltonian simulation. We move from a classical complexity that has a lower bound that depends exponentially on the lattice dimension to a quantum complexity that has an upper bound that depends quadratically on this same parameter. In this algorithm, we apply a sophisticated discretization method, via graph spectral theory, in order to achieve the discretization of the lattice. We show how to use this method of discretization to establish the Dirichlet and Neumann boundary conditions in our algorithm. We also work with a few practical examples to confirm that our method works.

We present a quantum lattice gas model via quantum walks. We generalized the first model of gas collision, called HPP, to code our quantum version. We employ a quantum walk with two interacting walkers in a two-dimensional lattice to perform our analyses. We show how this simple interaction generates entanglement between the particles and how it increases with time.

Resumo

O interesse pela área de informação quântica vem aumentando rapidamente nos últimos anos. Podemos dizer que o principal motivo pelo aumento desse interesse é os novos computadores quânticos que estão aparecendo. Há um grande interesse de diversos grupos com diferentes propósitos de testar as potencialidades dessas máquinas quânticas para investigar seus problemas. Neste novo cenário de computação precisamos desenvolver algoritmos quânticos e usar modelos quânticos de computação para explorar esses novos dispositivos. Indo para esta direção, os principais objetivos desta tese são propor melhorias para um método quânticos de computação, contrastando com seu análogo clássico, e desenvolver novos algoritmos quânticos.

Alguns resultados para modelos clássicos e quânticos de computação são apresentados, tendo como foco automatos celulares particionados e seu análogo quântico. Apresentamos uma nova definição para este modelo de computação, o qual acreditamos ser uma definição mais clara do que as anteriores. Além disso, mostramos uma prescrição geral para traduzir as principais classes de passeios quânticos, como o modelo da moeda, para automatos celulares quânticos. Esperamos que este resultado junto com a nossa nova definição aumente a atividade de pesquisa nesta área. Para a parte clássica dos automatos celulares, desenvolvemos o "coarse graining", uma ferramenta amplamente explorada para estudar processos emergentes. Nesta parte, vimos processos estocásticos surgindo de processos determinísticos.

Ainda no paradigma de espaço e tempo discretos, propomos um algoritmo quântico para simular a equação de onda via simulações de Hamiltonianas. Passamos de uma complexidade clássica que tem um limite inferior que depende exponencialmente da dimensão de rede para uma complexidade quântica que possui um limite superior que depende quadraticamente desse mesmo parâmetro. Neste algoritmo, aplicamos um método sofisticado de discretização, via teoria espectral de grafos, a fim de alcançar a discretização da rede. Mostramos como usar este método de discretização para estabelecer as condições de contorno de Dirichlet e Neumann em nosso algoritmo. Também trabalhamos com alguns exemplos práticos para confirmar que nosso método funciona.

Apresentamos também um modelo de redes de gases quânticos através de caminhadas quânticas. Nós generalizamos o primeiro modelo de colisão de gases , chamado por HPP, para codificar nossa versão quântica. Empregamos o modelo de caminhantes quânticos com dois caminhantes interagentes em uma rede bidimensional para realizar nossas análises. Mostramos como essa interação simples gera emaranhamento entre as partículas e como ele aumenta com o tempo.

List of articles related to the thesis

1. Pedro C.S. Costa, Stephen Jordan, Aaron Ostrander. **Quantum Algorithm for Simulating the Wave Equation**. arXiv:1711.05394, 2017. Results presented in chapter 9 of this thesis.

   The paper was submitted to journal *Quantum Information and Computation* in 28/11/2017.

   Current state: we are doing the corrections pointed out by the referees and adding new sections in order to answer some relevant questions made by them. We will resubmit it in the beginning of April.

2. Pedro C.S. Costa, Renato Portugal, Fernando de Melo. **Quantum Walks via Quantum Cellular Automata.** arXiv:1803.02176, 2018. Results presented in chapter 7 of this thesis.

   Current state: Published in Quantum Information Processing, https://doi.org/10.1007/s11128-018-1983-x.

3. Pedro C.S. Costa, Fernando de Melo, Renato Portugal. **Quantum HPP**. Results presented in chapter 8 of this thesis.

   Current state: we are converting the results shown in this thesis into an article.

4. Pedro C.S. Costa, Fernando de Melo. **Coarse Graining of Partitioned Cellular Automata**. Results presented in chapter 5 of this thesis.

   Current state: we are converting the results shown in this thesis into an article.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The age of the quantum computers has already arrived. Right now, there are some quantum computers available, for instance D-wave[1], IBM [41] and others to come by Microsoft and Google [31]. Even the general public can already access one of those in the cloud, IBM quantum computers [2]. Basic questions one asks are: "what we will be able to do with these machines?" and "what we will be able to learn with them?" We can be more specific with these questions, for instance: can we simulate nature more efficiently with quantum computers than with classical computers (in terms of time and memory)? Can we simulate systems that we could not do before, in such a way that large experiments will not be necessary anymore? Can quantum computers help us to answer fundamental questions, studies about the emergence of the classical world from the quantum one?

Richard Feynman [30], who pointed out some of these questions above, was more interested in quantum computers for simulating quantum physics basically because of two reasons: the description of the quantum state increases exponentially with the number of quantum particles, and from the fact that ultimately we believe that the physical rules are quantum and then we should work with hardware structure that employs quantum logic in order to mimic nature more efficiently.

However, since the first quantum algorithms, we could see that quantum computers would go far beyond of what Feynman expected. Quantum computers might imply new protocols for cryptography [6], search algorithms [59], to speedup the solution of linear systems [38], results that can impact significantly multiple and distinct areas. These results related with these areas that strongly boosted the development and investment to construct a quantum computer, bringing many different companies to this area, as cited above. Furthermore we are aware about the high

[1] https://www.dwavesys.com/home

[2] http://researchweb.watson.ibm.com/ibm-q/

costs in carrying out large experiments, as the ones employed to study fundamental interactions. Then, if we have quantum algorithms that can reproduce the reality of the phenomenon so well that we can learn about the system investigated, we might replace these large experiments by simulations, which is a topic that also has economic interests since we would spend much less economic resources working with quantum computers instead with large experiments. Of course that for this last claim we should be more careful. First of all, currently our classical computers are used to investigate complex system when experiments can not be done or when math gets too hard. Besides, to replace experiments by simulations can not be done without some certification criteria. In the same way that in experimental science one resulted achieved by some group needs to be replicate by others groups, computers models also need to be replicated, and independent groups should be able to get the same results from the model proposed. Then, in this way, when we have access to large quantum computers employing thousands of controllable qubits, we might go beyond of our current technology can go for experiments, and maybe the only way to learn about some phenomena will be by simulations, that needs to be replicated.

Aiming at the simulation of complex systems the first step is to construct an algorithm for the problem, which in general is not a trivial task. Rather than employing classical algorithms with quantum computers, we want to employ quantum algorithms, that require quantum logic. During this process we can already learn more about the problem, as for instance its computational complexity. Currently, many efforts are being done in this direction with several results already presenting, some with exponential speedups when compared with their classical counterpart, [7, 8, 16, 21, 38, 85]. As an example, a quantum algorithm that aims at computing the scattering process between particles in high energy physics was reported in [70]. Results like that, point out to us the real possibility of taking simulations instead experiments to study fundamental physics.

Throughout my academic research, these main questions presented above guided me in my work, and I am pleased to say that I have been successfully collaborating toward some of their answers. I particularly focused on the models of computation called Quantum Cellular Automata [36, 57, 83], along with its classical part [17], and Quantum Walks [1, 59, 77], focusing on new quantum algorithms to explore quantum and classical simulations and to explore emergent phenomena. Simultaneously I had the opportunity to help build a new quantum algorithm to solve second order partial differential equations, in particular the ones that solve the wave equation [21], just applying the well-known Schrödinger equation in a convenient way.

In this chapter I will give a general view about the work presented in this thesis, pointing out the motivations and open questions of each one.

1.1 Cellular Automata

Since Von Neumann [78], the theory of cellular automata (CA) was extensively explored in several distinct areas, such as biology [35], cryptography [52], and fluid dynamics [32]. It is not only

because its simplicity that CA are attractive to physics, it is also because their local formulation, which is strongly related with what we expect from physics: local interactions. During my PhD, I could see by myself the power of this model, chapters 2 and 3, applying it to a few problems in physics, such as the simulation of the diffusion equation [17] and particles that can collide [37]. After seeing the power of this model, it was quite natural for me to expect the same strength from its quantum counterpart, quantum cellular automata (QCA) chapter, 6.

Employing the QCA, I could work with some of its previous results and then learn how this model can be useful to simulate the Schrödinger and the Weyl equation. Likewise, in the classical version, the simulation is done with discrete time and space; the method to confirm if it is being done properly is to obtain the continuous limit of their discrete motion equation, which is not a trivial task. Therefore, I could see within a few examples that working with the QCA, which employs local operators and whose description is done in terms of qubits, we can simulate quantum systems properly. Thus, QCA is indeed an excellent platform for quantum algorithms and a possible hardware structure for quantum computers.

Another interesting aspect of the CA is the nature of its computation structure that allows us to study emergent phenomena easier than with other computational tools, as done in [39] (chap.5), which is another rich topic with several applications and questions. Typically, an emergent process in a physics context occurs when we move from a microscopic description to a macroscopic one, where the collective dynamics of a large assemblage of interacting parts emerges. Frequently, due to the weak sensibility of our detectors associated with the lack of information about the complete system, our recorded dynamics do not allow us to capture the full reality of the microscopic world. In general, when we move between these two worlds we start with more degrees of freedom and move to fewer in the macroscopic world. Despite not having complete access to all information, the main properties of the system are still present and we can describe them effectively. In several cases, it is exactly what we expect, and working with fewer degrees of freedom catches all the essential information, which demands fewer resources. The tool that usually allows us to explore this passage between these two worlds is known as coarse graining (CG).

Although some of these questions were addressed in [39], these results were reported only for the Wolfram's elementary CAs [84], which are not immediately important to physics. Since, during our incursions in the CA theory we learned about others classes of CA, as the "Partitioned Cellular Automata (PCA)", the one employed to simulate several diffusion processes and differential equations [17], we noticed that this class is more interesting to physics. Furthermore, after spending some time working with the PCA, we realized that it is this model - and not the models of Wolfram's classification - that must be quantized to achieve its quantum counterpart, the QCA. Therefore, if we were able to do an analogous study to the one done in [39] in this class of CAs, it may find several applications. Moreover, if we have access for the coarse graining tool for the PCA we can generalize it to the QCA. Finally, with this tool in hands for the QCA, we can, for instance, use it to study the quantum-to-classical transition.

Once we realized the potential of coarse graining techniques to the PCA we started to develop this tool to this CA class. We successfully obtained a coarse graining tool to the PCA and we established interesting results reported in chapter 5. For example, by starting with a deterministic PCA, we achieved a probabilistic PCA after we have applied our coarse-graining tool. The more interesting thing about this result is the fact that the transition function of this probabilistic PCA is equivalent to the discrete motion equation that describes the random walk. Therefore, we can see randomness as an emergent process, and we obtained a tool to understand and predict the emergence of large scale behavior in a system, starting from its microscopic description.

Despite of all the results for classical cellular automata, and even with these previous results - the Schrödinger and the Dirac equations - which are well known in terms of QCA, the quantum cellular automata do not share the same amount of research activity as its classical counterpart. The phenomenology and applications of quantum cellular automata are very scarce. One possible reason for the QCA to not be yet widely employed is that a clear definition was unavailable until recently [83]. Then, after we realize about the potentialities of the QCA model and the lack of its application we proposed another QCA definition, chap.6, that allowed us constructed a bridge between this model and another well known model of computation widely explored: *quantum walks*. These results might bring the community attention for the QCA, as will briefly comment in the next section.

1.2 Quantum Walks

While Schrödinger and Dirac equations simulations were done with QCA, I simultaneously did the same with another well known model of computation, widely applied to simulate quantum physics that also works with discrete time and space, the quantum walks. This quantum model, the quantum counterpart of the random walks, is being employed in various applications, most notably in quantum simulations [3, 26, 50] and in quantum algorithms [59]. Quantum walks, like its predecessor, comes in various flavors – coined [1], Szegedy [72], staggered [62], and staggered with Hamiltonians [61], to cite a few – each with its own specificity and tuned to better deal with a given problem at hand. Foremost, some models of quantum walks were shown to form a universal platform for quantum computation [14, 15]. All this versatility and possible applications prompted an intense experimental activity with various realizations of quantum walks, for instance with trapped ions [90], optical lattices [27] and more [79].

Working with these two platforms for the same problems, I concluded that their equation of motion in terms of amplitudes are equal, as we expected, since they are describing the same problem. Thus, dealing with both models at the same time, I could learn how to translate the operators in terms of QW to the QCA systematically, in such a way that with each time step, the state described by the QW can be reproduced by the QCA. At the time, I was restricted to the coined model described on lattice structure. After I observed the existence of other flavors as the

Szegedy and the Staggered QWs, the natural question was if this translation between these two models of computations could be made also for these others flavors. Having this goal in mind, my advisor and I contacted an expert in quantum walks, Renato Portugal (LNCC) [59]. Since then, we have worked together with the goal to show how translate the several QW flavors to QCA, chap.8. Presumably, these results will increase the activity in the QCA model, once we will be able to bring all the previous quantum walks results, from simulations to search algorithms, to the QCA platform, which is more experimentally friendly and it is an excellent candidate to be a hardware structure for a quantum computer.

Another possible strength of the quantum walks is the ability to employ multiple walkers simultaneously with interaction between them, which can lead to more powerful applications [9, 15, 67]. However, due to difficulty of dealing with multiparticle quantum walks, since the dimension of the space state increases exponentially, this area was little explored and only has been initiated in [55]. Moreover, until now, we have only seen results with two walkers an one dimensional lattice or on finite graphs with small number of vertices. Although results are very scarce until now, we can glimpse how powerful these algorithms can be. For instance, in [9], two interacting and non-interacting quantum walkers are employed on arbitrary graphs to distinguish nonisomorphic strongly regular graphs. In fact, there is a large list of possible applications for QWs with more walkers. For example: we can employ these models of multiple walkers to study the entanglement dynamics between many particles, to study quantum thermalization processes, search algorithms, and so on.

Furthermore, in my studies I worked with the model proposed by Hardy, Pomeau and Pazzis called HPP, because of their names, which is a CA rule for multiple particles [37]. Thus, I was familiar with this kind of interaction between the particles. After we had joined forces with Renato Portugal and began my studies with QWs, we became interested in the problem with more walkers following the same type of the interaction as in HPP. The first challenge in this problem was that the HPP was presented in terms of CAs. However, from our previous results, which translated QWs to QCA, the inverse path was not a difficult task. Another barrier that we faced was the huge dimension of state space for two walkers in the two-dimensional lattice. To work around the problem, we took the advantage of the sparsity of the unitary operator for these two walkers, and we successfully implemented it in classical computers in order to do simulations of this new dynamics. Currently, we are doing several investigations, trying to understand their dynamics numerically and analytically while we study their entanglement dynamics. However, this study is just at its beginning - there are several directions to take; for instance to use walkers for search algorithms. In this thesis I will report in chap.8 the results that we have established so far.

1.3 Quantum Algorithms for Classical Simulations

Now we can return to the previous question: can we simulate nature more efficiently than the classical computers, in terms of time and memory? So far I have discussed algorithms for simulating quantum systems. However, quantum computers can help us to solve problems unthinkable by classical computers, problems that would normally take an unimaginable amount of time in order to see their outputs. One classical example is the Shor algorithm [63], for integer factorization. Shor obtained a polynomial-time quantum algorithm to do this task against the exponential algorithms available for classical computers.

There is huge interest from the industry in seeing applicable quantum algorithms that will demand less resources. In other words, the industry wants to apply better algorithms to solve their problems more efficiently. They mostly need algorithms for numerical problems, such as algorithms to solve differential equations or linear systems. Going in that direction, I had the opportunity to collaborate with Stephen Jordan (UMD/NIST) and Aaron Ostrander (UMD). Together, we proposed a quantum algorithm for simulating the wave equation, chap.9. Essentially, this algorithm employs the Schrödinger equation to solve the desired partial differential equation, [21]. Employing previous results of quantum computation, as in [8], we established an algorithm that has a polynomial complexity in terms the dimension of the lattice, instead of the exponential complexity of its classical counterparts, chap.4. There is a wide variety of problems where our algorithm can be useful, from several electrodynamic scattering processes up to relativistic problems.

This thesis is divided into two parts: in part one I report only the classical results, and in part two I present the quantum results. We will present four new results in this thesis, one that is in part I "Classical Models of Computation" and the others three are in part II "Quantum Models of Computation". In order to facilitate the understanding of readers in Fig.(3.1) we did a flowchart to guide them through these results. Showing the chapters that are recommended to be read first in order to get the necessary background for each new result.

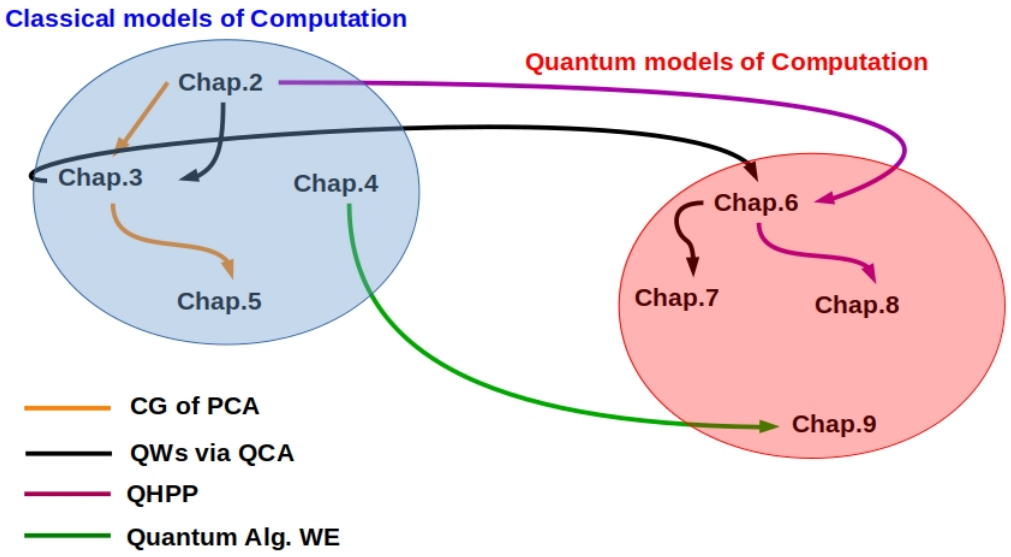


Figure 1.1: The aim of this figure is to guide the reader through the new results that are present in this thesis. Each sequence of arrows with the same colors, always starting from the lower to the higher chapter, indicates the chapter orders that are recommended to be read first. The four results are: coarse graining of partitioned cellular automata (CG of PCA), quantum walks via quantum cellular automata (QWs via QCA), quantum HPP (QHPP) and quantum algorithm for wave equation simulation (quantum Alg. WE).

Part I

Classical Models of Computation

CHAPTER 2

CELLULAR AUTOMATA

A model of computation widely explored in several areas, from biology to physics, is the cellular automata model. There are different reasons for the CA be so attractive and we could say that one of them is its simple formulation. A cellular automaton is a lattice of cells such that at any one moment in time each cell is in one of a finite set of states. At each discrete time step the state of each and every cell is updated according to some *local transition function*. The input of this function is the current state of the corresponding cell, and the states of the cells in some finite neighborhood. Although they have this simple structure different dynamics can be achieved by choosing distinct transition functions (rules), which is the name of the function that will update each cell accordingly with their neighborhood, or employing different finite alphabets, the part that has the information about the number of possible states that each cell has.

Cellular automata were originally proposed by John von Neumann in late 1940's [78]. The automaton originally described by him is a two-dimensional infinite array of uniform cells, where each cell is connected to its four neighbors (see figure (2.1)). The main purpose of von Neumann was to bring the rigor of axiomatic and dedutive treatment to the study of complex natural systems, such as the human nervous system. One of his motivations to study complex systems was to build large computers to solve very complex problems. His ambition was to construct an artificial cellular automata with the capacity of self-control and self-repair, like the nervous system. Von Neumann suggested that this system of cells, where each cell is characterized by an internal state, evolves in discrete time steps and the rule of evolution is the same for all the cells. Because of his premature death, von Neumann was unable to put in a final form the research he was doing in automata theory. But following the same line of research many other authors continued to develop his theory of automata. One of the greatest contributions came in 1970 by

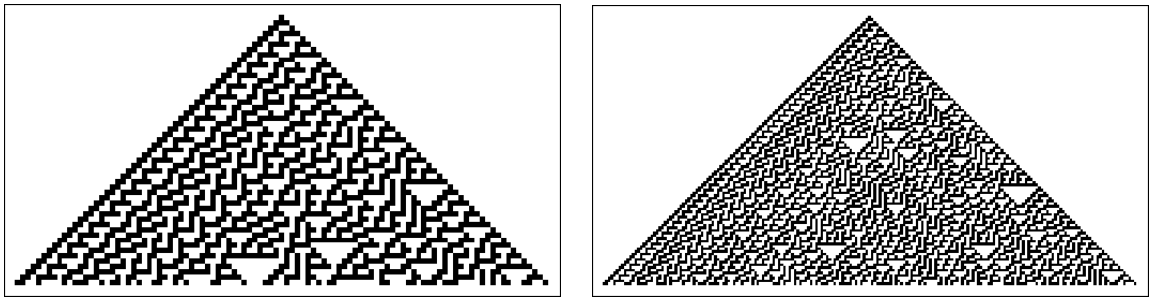
Figure 2.1: The von Neumann neighbourhood for a regular 2D CA.

the mathematician John Conway [33], who proposed his *Game of life*, a mathematical game in terms of cellular automata. His motivation was to try to find a simple rule that lead to complex behaviors. His idea was quite simple, he proposed a two-dimensional lattice CA, following the neighborhood scheme of (2.1) plus the four second nearest neighbors along the diagonals within two states: state one and state zero, states that he interpreted as alive and dead cell respectively. The updating rule (transition function) of the game is given as follows: a dead cell surrounded by three alive cells comes back to the life. On the other hand, a living cell surrounded by less than two or more than three neighbors dies of isolation. Even employing these simple rules the game of life has rich behavior and John Conway also could show that Life could simulate a universal computer. These results called attention to the public and more and more people started to become interested in the CA theory. However, despite a range of different interesting results for cellular automata got from different mathematicians and computer scientists, some say that the most significant CA contributions were given by the theoretical physicist, Stephen Wolfram in the early 1980's.

Stephen Wolfram a physicist working at the Institute for Advanced Study in Princeton, became fascinated by the CA ideas and the different patterns that it can do. Wolfram decided to investigate the CA dynamics. In order to do that, as a good physicist, he initiated his studies from the simplest CA case, **Elementary Cellular Automata** (ECAs), a model that himself proposed. The ECAs are one dimensional CAs, two-state in which each cell is connected only to its two nearest neighbors, (we will explain more formally this model in the next section). Wanting to understand better the CAs dynamics, Wolfram and his colleagues developed a special programing language, called Mathematica [1], a software that went far beyond its initial goals and current it is applied in distinct areas for different proposals. During his dynamics investigation in CAs, Wolfram became fascinated by rule 30 (there are 256 rules for the ECAs as we will see in the next section) and he would like to understand how the complex patterns emerged from the very simple CA rules, see Fig.(2.2). One of the greatest of Wolfram's results was his proof that the rule 110 is a universal model of computation, perhaps the simplest known example of a universal computer. However, we can say that the most important Wolfram contribution was given in his book *A new kind of Science* [84]. This title name refers to the idea that the universe and everything inside in it, can be explained by simple programs. In this book Wolfram showed how to apply CA in

[1] https://www.wolfram.com/mathematica/

different areas of science. Wolfram's contribution was extremely important for the development of CA understanding and called much public attention and until today we can see his ideas being applied in different areas, [17, 35, 52].



a) b)

Figure 2.2: In these figures we can see rule 30 in two different time instants Fig.(a) $t = 50$ and Fig.(b) $t = 100$, where the time flows downwards. Both dynamics have the same initial condition, with only one black cell in the first row. These figures were generated by Mathematica, which has special functions for CA dynamics.

After this brief introduction on the history of CA, we present a more formal CA definition:

Definition 2.1. [CA] A Cellular Automata is a 4-tuple $(L, \mathcal{N}, \Sigma, f)$ consisting of:

1. a d-dimensional lattice of cells indexed by integers $L \subseteq \mathbb{Z}^d$;
2. a finite neighborhood scheme $\mathcal{N} \subseteq L$;
3. a finite set Σ of cell states;
4. a local transition function $f : \Sigma^{\mathcal{N}} \to \Sigma$;

The transition function f simply takes, for each lattice cell position $x \in L$, the states of the neighbors of x, which are the cells indexed by the set $\mathcal{N}_x = x + \mathcal{N}$ at the current time step $t \in \mathbb{N}$ to determine the state of cell x at time $t+1$. There are two important properties of cellular automata that should be stressed. Firstly, cellular automata are space-homogeneous, in that the local transition function applies the same function at the neighborhood of all cells. Secondly, cellular automata are time-homogeneous, in that the local transition function does not depend on the time step t.

The main proposal in this chapter is to introduce different CA class and see some applications. Moreover, we will see and understand what is the best CA class to physics.

2.1 Elementary CA

Elementary cellular automata are the simplest CAs possible. They are 1D binary CAs with neighborhood size one. In this way the evolution of a given cell is dictated by its state and the

states of its left and right neighbors. This limits the number of elementary CAs to just $2^{2^3} = 256$, of which just 88 are distinct [2]. Having so few rules makes the study of the entire rule space practicable and wouldn't leave us reliant on sampling a small (and possibly unrepresentative) corner of the rule space.

Consider rule 90. The rule number is calculated by finding the decimal representation of the binary values of the rule's output states, as shown in figure(2.3). This figure shows a pictorial representation of the CA transition function, as presented in [84].

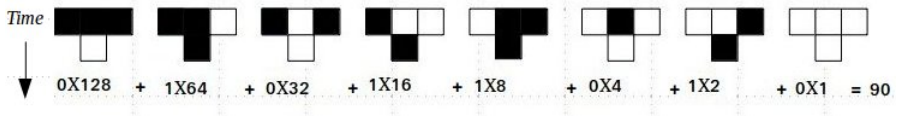


Figure 2.3: Elementary CA rule 90. Here the number one is represented by the black box, while the number zero is represented by the white box. As we have explained before, the second line is determined by the three cells above.

As ECAs are one dimensional, we can efficiently show the evolution of the CA over time by placing each subsequent generation underneath the last. Figure(2.4) gives a pictorial description of the automaton's evolution in time.

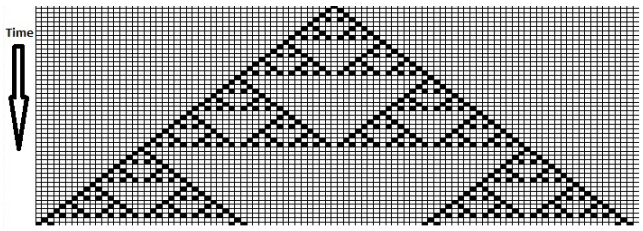


Figure 2.4: This figure represents the evolution in time of the CA described in Fig.(2.3).The initial configuration, shown in the first line, is a single cell coloured black, while all others are white. Time flows downwards. Each subsequent line depicts the current state of the automaton, and all stages since the initial one.

We can see the ECAs in a formal way, as described in the definition 2.1. The 4-tuple is $(\mathbb{Z}, \mathcal{N}, \Sigma, f)$, with $\mathcal{N} = \{x-1, x, x+1\}$ $\Sigma = \{0,1\}$ where we have illustrated these states with black and white, $0 \rightarrow$ white, $1 \rightarrow$ black. Then in this case the update rule is a function $f : \Sigma^3 \rightarrow \Sigma$ and

$$a_n(t+1) = f[a_{n-1}(t), a_n(t), a_{n+1}(t)], \tag{2.1}$$

where the indexing $n \in \mathbb{Z}$ gives the position of the nth cell and t is the time. At each time step, every cell in the lattice is updated accordingly to the updating rule, as we explained before. As a practical example we can see how the set of cells in the first line in Fig.(2.4) is updated using

[2]The others are either reflections or inversions. If a rule produces diagonal lines that move to the left over time, the reflection would produce a mirror image with lines that moved to the right. An inverse rule substitutes 1 for 0 in all cases.

Eq.(2.1). There we start at the time $t = 0$ and $a_0(0) = 1$ while the state of the all other cell is equal to zero, $a_i(0) = 0$, $i \in \mathbb{Z}^*$. From the rule 90 in Fig.(2.3) we can write the following evolution,

$$\begin{aligned} a_0(1) &= f[a_{-1}(0), a_0(0), a_{n+1}(0)] \\ &= f[0,1,0] \\ &= 0, \end{aligned}$$

there are only more three distinct configurations that we need compute, the first one is

$$\begin{aligned} a_1(1) &= f[a_0(0), a_1(0), a_2(0)] \\ &= f[1,0,0] \\ &= 1, \end{aligned}$$

the second is symmetric of the last one $a_{-1}(1) = f[a_{-2}(0), a_{-1}(0), a_0(0)] = f[0,0,1] = 1$ and finally the same output $a_n(0) = f[a_{n-1}(0), a_n(0), a_{n+1}(0)] = f[0,0,0] = 0$, for all $n \geq |2|$.

2.2 Reversible and partitioned CA

In this section we propose a special class of CA, that employs characteristics of two well known CA classes. These classes were introduced to facilitate the construction of reversible and conservative CA that also allow us to work with the concept of conservation laws, a quite important physical constraint. As many physical systems are invariant under time-reversal and employs conserved quantities, as the number of particles, in a microscopic level, the CA should guarantee these conditions.

For a CA to be reversible the global state transition function, $F : \Sigma^L \to \Sigma^L$ must be reversible. In other words, each global state $C \in \Sigma^L$, where C denotes the state of the entire CA, must have a unique successor and predecessor. Moreover we need to work with a CA definition that allow us to work with the concept of conservation. Our purpose here is to introduce a construction that ensures the reversibility of global states' transition functions, using local properties, and that can also allow for the definition to employ the concept of conservation.

We are aware that from the Toffoli and Margolus CA definitions [76], **block cellular automaton** and **partitioning cellular automaton**, well known classes of CA, that we have alternatives available to work with reversibility and conservation.

The idea of the block formulation is to start from a lattice of cells and then divide it into blocks of non-overlapping cells. Alternatively to the previous CA definition, where in order to update the state from t to $t+1$ we have to apply the transition function only once in a given neighborhood scheme, the evolution here happens in parts. In this definition we have to apply a local transition function f at each block. This operation only concerns with the current block state, which is another contrast compared with the Wolfram's classification, where the transition function need to consider all neighborhood states. Subsequently the block is shifted, in such a

way that in the new block there are different cells, and again we apply f. Only after these two operations we say that the entire CA state was updated. The cellular automaton is guaranteed to be reversible if the local transformation f is itself a bijection.

The partitioning CA employs a different formulation in order to achieve the same goals. The mechanism here consists in partitioning each cell into a finite number of parts, where each part is devoted to interact with another cell part. Likewise the block formulation the update occurs in two parts. We have a local operator that interacts all parts from a given cell and another one that interacts parts from the different cells. Once again, the reversibility can be achieved employing bijective operations.

Within these formulations we have a convenient way to employ conservation laws, as the momentum conservation or particle number. For instance, imagine that we relate a cell state with a particle and we choose local operators that maintain the same number of states at each block. Then, in the end the number of particles is conserved.

Here we propose another CA definition that employs the main characteristics of these previous models, that we called it **partitioned cellular automata**. We called by partitioned since we kept with the same concept of cell partitions, but here we call each cell part a subcell. Similarly to the previous definitions the full evolution occurs in steps, that does not have a fix number, by local operators. Differently from the previous definition, we can have the same subcell interacting with different subcells during its update. For example, we can have a transition function that takes the CA state from t to $t+1$ that was divided in three parts. We can have in this case the same subcell interaction with different subcells in each part of the evolution.

The closeness with the block definition will be clear now. As we just mentioned the evolution happens in parts. At each part we associate an operator that acts into a set of subcells, that does not have overlap with another set. In this sense we can think these set of subcells as blocks that can be shifted during the updated. Therefore, from this illustration we can see some characteristics of the block definition.

Unlike the previous definitions here we equipped our definition with the concept of tiling. A tilling is a uniform partition of the set of subcells and we call *tile* each element of this partition. With this definition in hand we can work with the partition concept or block construction more precisely.

Formally, we define partitioned CA in the following way:

Definition 2.2. [PCA] A Partitioned Cellular Automaton is a 5-tuple $(L, \mathscr{N}, \Sigma, \{\mathscr{T}_i\}, \{\sigma_i\})$ consisting of:

1. a d-dimensional lattice of cells indexed by integers $L \subseteq \mathbb{Z}^d$;
2. a finite neighborhood scheme $\mathscr{N} \subseteq L$;
3. each cell is divided in n subcells, and to the i-th subcell we assign a copy Σ_i of finite alphabet of Σ. The total alphabet associated to each cell is then $\Xi = \Sigma_0 \times \ldots \times \Sigma_{n-1}$;

4. a finite set of tilings $\{\mathcal{T}_i\}_{i=0}^{N-1}$. Each tiling is the union of identical non-overlapping tiles, $\mathcal{T}_i = \bigcup_j T_j^{(i)}$, with each tile $T_j^{(i)}$ containing only subcells of neighboring cells;

5. a set of local functions $\{\sigma_i\}_{i=0}^{N-1}$. The same operator σ_i is applied to each tile $T_j^{(i)}$ of the tiling $\mathcal{T}_i$.

With this definition, the transition function $\mathscr{E} : \Xi^L \to \Xi^L$, which updates the automaton state from the time t to $t+1$, is given by

$$\mathscr{E} = \prod_{i=0}^{N-1} \left(\bigtimes_{T_j^{(i)} \in \mathcal{T}_i} \sigma_i \right). \tag{2.2}$$

To work with these tilings more precisely, it is convenient put labels in each subcell. Given the cell at position $x \in L$, its subcells are denoted by x_i, with $i \in \{0, \ldots, n-1\}$. For instance, suppose we have a one-dimensional lattice where each cell has two subcells, and the neighbor scheme is $\mathcal{N}_x = \{x-1, x, x+1\}$. In this case two tilings are sufficient to evolve the automaton: The first tiling is given by $\mathcal{T}_0 = \bigcup_{x\in\mathbb{Z}} T_x^{(0)}$ with each tile defined as $T_x^{(0)} = \{x_0, x_1\}$. For the second tiling, $\mathcal{T}_1 = \bigcup_{x\in\mathbb{Z}} T_x^{(1)}$, each tile is given by $T_x^{(1)} = \{x_1, (x+1)_0\}$. It is then clear that the first tiling is responsible for "reading" the state of each cell, while the second one is responsible for the interaction between the neighboring cells. Now that the tilings' structure is established, the action of the operator is clear:

$$\begin{aligned} \sigma_0 : (\Sigma_0)_x \times (\Sigma_1)_x &\to (\Sigma_0)_x \times (\Sigma_1)_x ; \\ \sigma_1 : (\Sigma_1)_x \times (\Sigma_0)_{x+1} &\to (\Sigma_1)_x \times (\Sigma_0)_{x+1} \end{aligned}$$

for all $x \in L$. Therefore, in this example we can explicit our transition function as

$$\mathscr{E} = \left(\bigtimes_{T_x^{(1)} \in \mathcal{T}_1} \sigma_1 \right) \left(\bigtimes_{T_x^{(0)} \in \mathcal{T}_0} \sigma_0 \right). \tag{2.3}$$

If we choose σ_i for $i = 0, 1$ as a permutation function, which is clearly reversible, our CA is reversible. The sequence of steps given by $\mathscr{E}$ is illustrated in Fig.(2.5).

2.3 Cellular automata modeling

The purpose of this section is to show one specific example that illustrates how cellular automata may be useful for simulating physics.

2.3.1 The HPP rule

We will now show a very important rule for this work, which can also show us the power of cellular automata to simulate physical phenomena. The HPP rule [37] is a model of a two-dimensional gas which the particles can collide, introduced by Hardy, Pomeau and Pazzis (HPP). For this rule

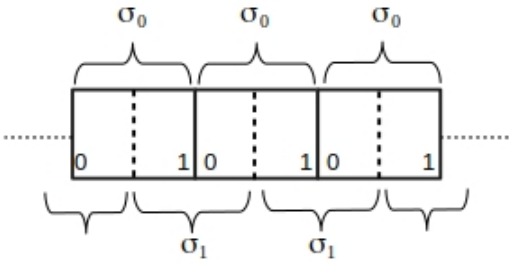


Figure 2.5: This figure presents how each cell is split in two subcells, and how the operators σ_i are applied in accordance with two tilings.

to establish such a phenomenon it must reproduce the basic laws of conservation of microscopic interaction, such as local conservation of momentum, the number of particles and time reversal.

As we have discussed in the section (2.2) we can establish CA which are reversible, in a more simple way, by the formalism of PCA.

Here we are dealing with 2-dimensional lattice of cells indexed by integers, $L = \mathbb{Z}^2$. In this case $|\Xi| = 16$ because we are working with four subcells per cell, 4 bits. In this case we have two subcells in the x direction which we label by 0 and 1, left and right respectively and the two others in the y direction whose labels are 3 and 2, up and down respectively, writing in another way $\Xi = \Sigma_0 \times \Sigma_2 \times \Sigma_1 \times \Sigma_3$. The neighbor scheme is $\mathcal{N}_{xy} = \mathcal{N}_x \cup \mathcal{N}_y$, where $\mathcal{N}_x = \{x-1, x, x+1\}$ and $\mathcal{N}_y = \{y-1, y, y+1\}$. The complete updating step consists of three parts, whose functions will be explained along this section, therefore we need three tilings in order to update this model in a proper way. The tiles from the first tiling are $T^{(0)}_{xy} = \{x_0, x_1, y_2, y_3\}$. Unlike the first tiling the tiles from the second tiling have subcells from different cells, however there are only interactions between subcells related with the same axis, $T^{(1)}_x = \{x_1, (x+1)_0\}$ and $T^{(1)}_y = \{y_3, (y+1)_2\}$ where in the end the second tiling is the union of these two sets regards to the x and y directions, $\mathcal{T}_1 = \left(\bigcup_{x\in\mathbb{Z}} T^{(1)}_x\right)\left(\bigcup_{y\in\mathbb{Z}} T^{(1)}_y\right)$. Finally the third tiling is equal to the first one $\mathcal{T}_2 = \mathcal{T}_0$.

First we describe the rule of evolution without any concern about the physical representation and only after we will show its physical meaning. Let us start analyzing the first operator that we apply simultaneously on all cells into each cell at same time. The permutation operator applied in this rule is given by

$$\begin{aligned} \sigma_0(1_0, 0_2, 1_1, 0_3) &= (0_0, 1_2, 0_1, 1_3), \\ \sigma_0(0_0, 1_2, 0_1, 1_3) &= (1_0, 0_2, 1_1, 0_3), \end{aligned} \tag{2.4}$$

The other fourteen states remain in the same state after the action of σ_0, as an example: $(1,0,1,1) \rightarrow (1,0,1,1)$.

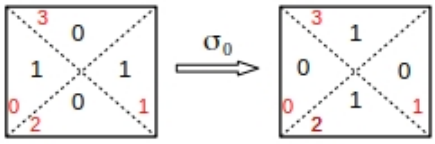


Figure 2.6: The square represents just one cell but partitioned in four subcells.

Now we can move to the next operator. In Fig.(2.7) from the elements of the second tiling we can see where the operators σ_1 are being applied i.e. on subcells in tiles for the second tiling $\mathscr{T}^{-1}$. Concerning those neighbors in the figure, the cell located at xy, whose state is $\Sigma_{xy} = (1_0, 1_2, 0_1, 0_3)$ will have the following final form after the action of σ_1, $\Sigma_{xy} = (1_0, 0_2, 1_1, 0_3)$. In other words, σ_1 is just a swap between its neighbor. This process involves interaction between neighbors.

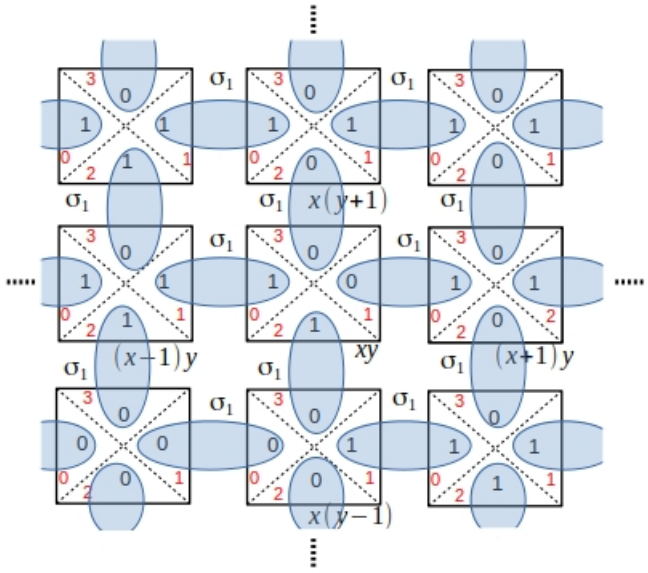


Figure 2.7: Here we are illustrating how the action of σ_1 proceeds.

To finish one time step, we need to apply σ_2 which is just a swap operator between the states 3, 2 and 0, 1 from the same cell for instance, $(1,0,0,1) \rightarrow (0,1,1,0)$.

Now we discuss the physical properties of of this rule. Each subcell, see figure (2.6), can only allocate one particle, where 1 represents the existence of one particle and 0 the absence. Each position in the four entries now will indicate the direction of movement of the particle, as illustrated in figure (2.8).

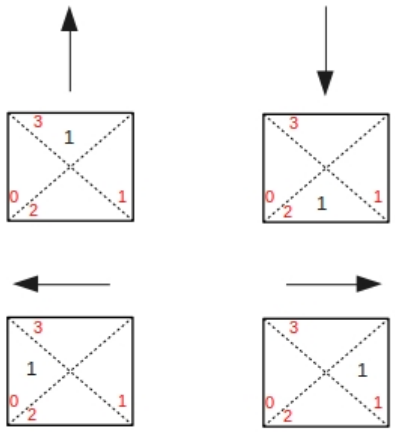


Figure 2.8: The four possibilities of motion of one particle.

Then with these properties we can easily see that σ_0 represents the rule of collision between particles. Clearly the momentum is preserved and give some non-trivial dynamics. Another option, that we could have chosen, is a rule where the particles flip their direction afterwards the collision but keeping their motion direction along the same line,

$$\rightarrow\leftarrow \overset{\sigma_0}{\Longrightarrow} \leftarrow\rightarrow$$

However, it would be equivalent to particles that cross each other without interaction. Moreover it is also clear that σ_1 with σ_2 represent the propagation of the particles. Being more precise in the role of σ_2, it is the part that preserves the motion direction of the particle. Suppose, for instance, that we have a free particle at position xy going to the right direction and the transition function has only σ_0 and σ_1. After we apply $\mathscr{E}$ we want to see this particle at position $(x+1)y$ going to right as well, however from the definition of σ_1 aside with the direction meaning Fig.(2.8), in the end we get a particle at position $(x+1)y$, but going to the left direction, that is an unwanted result. In order to fix it we have to add the operator σ_2.

Often the transition function of this model is written as

$$\mathscr{E} = \mathscr{P} \circ \mathscr{C},$$

where the first part $\mathscr{C}$ is related with the collision and then

$$\mathscr{C} = \prod_{T_{xy}^{(0)} \in \mathscr{T}_0} \sigma_0,$$

and the second with their propagation, thus

$$\mathscr{P} = \left(\bigtimes_{T_{xy}^{(2)} \in \mathscr{T}_2} \sigma_2 \right) \left(\bigtimes_{T_{xy}^{(1)} \in \mathscr{T}_1} \sigma_1 \right).$$

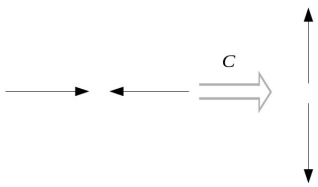


Figure 2.9: We have here the same rule of permutation, as we showed in figure (2.6), but now we have introduced the representation of particle motion.

In case we want to impose boundary conditions we can do it imposing different rules at specific points. For instance, in Fig.(2.10) we are simulating a gas collision process in a closed box. We can do it just removing the action of the operator σ_2 at the border cells.

Although for the HPP problem each particle can be described by Newton's laws and then we could in principle determine the position of each particle at each time, when we work with many particles the dynamics begins to get very complex, Fig.(2.10). Thus a good approach to deal with this problem is to employ the thermodynamics laws that gives laws of macroscopic entities, for example, heat, energy and entropy.

Therefore, for this simple example we can see that the bridges that connect classical mechanics to thermodynamics is statistical mechanics. Since the last theory proposes that large-scale properties emerge from microscopic properties and in our example these particles, that play the role of the microscopic entities, are the source of these macroscopic quantities.

2.4 PCA vs Wolfram classification

Before we finish this chapter we will point out some differences between these two classes os CA presented.

Whereas in the Wolfram classification to move the entire CA state from t to $t+1$ we have to read each individual state before we update it, in the PCA we only have to employ local operators.

Let us understand better how these differences make these two classes quite distinct. In practice when we have to update, for instance, the ECAs state to $t+1$ we need to do a copy of the state t. With this copy in hands we keep with it until we read all states $a(t)$ along its neighbors in

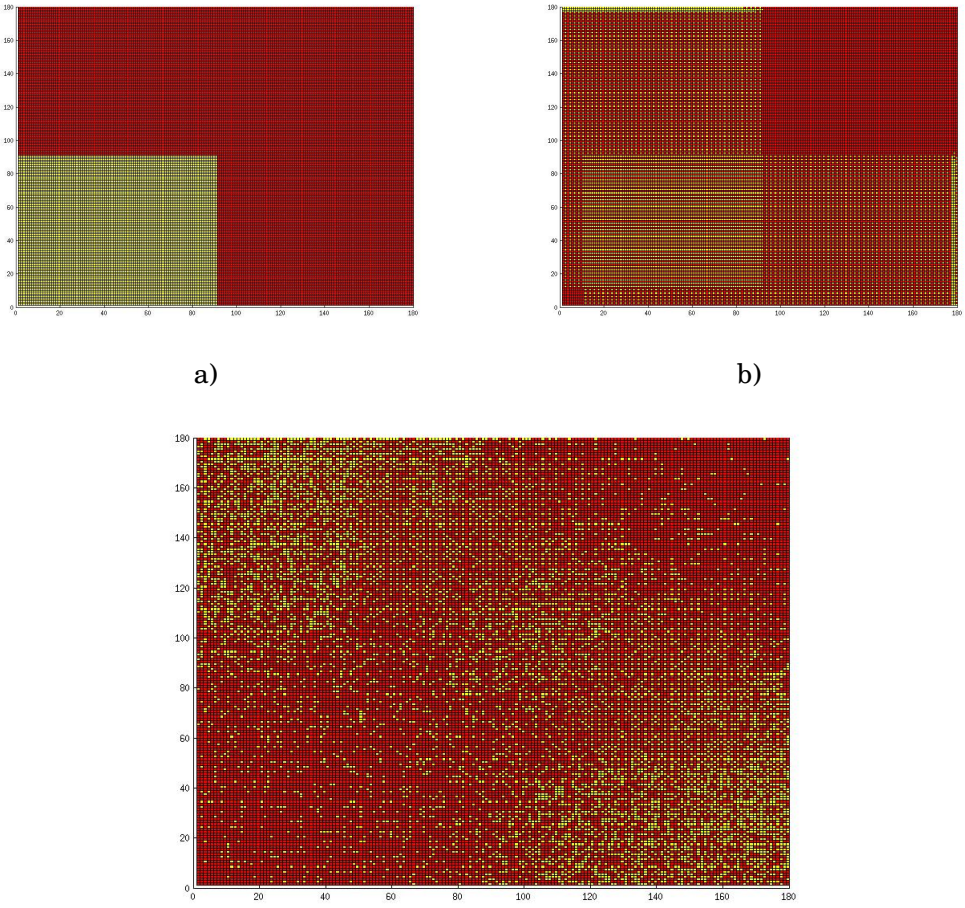


Figure 2.10: In these figures we can see the particles inside a closed box in three different time instants. In Fig.(a) the particles are confined on the left corner. In Fig.(b) after 50 time steps we can see the particles spreading around the box in very ordered behavior. Finally in Fig.(c) after 450 time steps we can see a disorderly behavior, where the particle are spread along all the box.

order to achieve $a(t+1)$. On the other hand, in the PCA class we do not need to work with copies. Since there are no overlaps between different subcells, in each part of $\mathscr{E}$, the updating happens without necessity of read its neighborhood first.

These differences are the reasons for us to adopt a different procedure in chapter 5 in order to get the CG for the PCA. Moreover, these differences make the PCA, and not the ones that belongs to Wolfram classification, the classical counterpart of the QCA, as we will see in chapter 6.

CHAPTER 3

LATTICE GAS AUTOMATA

This chapter has two main goals. The first one is to describe two well-know problems: Brownian motion-(BM) and the random walk-(RW) problem, both in CA formalism. The second one is to analyze the continuum limit of the cellular automata describing both problems in one dimension.

3.1 A Brownian motion automaton

It is almost certain that if we ask people about the main Einstein results of 1905 they will immediately say the special relativity theory and the photoelectric effect. However there is another one as important as these two others, the **Brownian motion** result, [5]. This result was revolutionary in science playing an important role not only in the traditional topics in physics, but also in subjects like life science, studies in the stock market [56], and much more.

Before 1905 the atom and molecules existences were just theoretical ideas not well accepted in the science community, for instance the physicist-philosopher Ernst Mach who believed that atoms do not have practical utility. We can say that these investigations were initiated with **Robert Brown** in 1827. During his studies with pollen he made interesting observations about their microscopical behavior immersed in water. Brown noticed an incessant random motion of these particles that leads to thinking it as the result of live pollen grains. As the next steps he continued with the same experiment but employing different materials. In the end he could conclude that these random motions were not the signatures of life. All these results brought new possible ideas for the incessant particle motion, for example, the temperature gradients. Moreover, in the end, they could prove the existence of atom and molecules.

Back to the Einstein results about Brownian motion, we can point out one intermediate step

of his calculation, extremely important to us. He obtained the diffusion equation

$$\frac{\partial P}{\partial t} - D\frac{\partial^2 P}{\partial x^2} = 0,$$

for $P(x,t)$ the probability distribution of the position x of the Brownian particle at time t, where D is the diffusion constant. Despite the fact this equation was very well knowing in that time Einstein's derivation established a connection between the random walk of a single particle and the diffusion of many particles, as we will see better in this chapter. All these investigations by Robert Brown and Einstein, (and others as the Smoluchowski result [68]) culminated with the Nobel prize of Jean Perrin, who tested Einstein's theoretical predictions 1926. Therefore, after the experimental evidence all the scientific community, including the anti-atomists, could convince themselves about the atom's reality.

As we mentioned in the beginning of this chapter, the Brownian motion is widely explored until today. In particular it is the basis for a numerical algorithm for simulation of molecular process [17, 47], that is our main focus here. To be more specific we are not interested in simulating any specific process. Our interest is to see how we can employ the PCA to do this task.

3.1.1 The problem statement

The basic ideas for this problem are very simple. If we have some particle moving in some fluid, as for instance a grain of pollen in a glass of water, this particle will collide with some water molecules. When the collision happens the particle changes its movement direction [53]. In the real pollen experiment they could observe an average of 10^{12} collisions per second. Thus we can say that the pollen can only do a free displacement, without any collision, an average during 10^{-12} seconds. Let us try to explain this process in one dimension case, where the particle can only move to right or to left. Then if we start with a particle going to right we say that, on average, it can only move freely, without collision, for a short period of time, then, after this short displacement the collision happens with some probability. Then, we say that with probability q the particle will collide with some water molecule changing its movement direction. Then, in case the collision happens, it starts to move again, now to the left, for the same average period. Therefore we can write the following recurrence relation for this problem

$$\begin{aligned} P_l(x,t+1) &= qP_r(x+1,t) + pP_l(x+1,t), \\ P_r(x,t+1) &= pP_r(x-1,t) + qP_l(x-1,t), \end{aligned} \tag{3.1}$$

where $P_i(x,t+1)$ give us the probability to find a particle at point x in a time t going to right, when $i = r$, or going to left, when $i = l$, and $p + q = 1$. We add both equations above and take $P(x,t) = P_l(x,t) + P_r(x,t)$ to find

$$P(x,t+1) = p\left[P_l(x+1,t) + P_r(x-1,t)\right] + q\left[P_r(x+1,t) + P_l(x-1,t)\right]. \tag{3.2}$$

To understand better Eq.(3.1), let us analyze the equation for $P_l(x,t+1)$. The expression for this term says that we will find a particle going to the left at time $t+1$ in the position x if either, at the time t, we had a particle at $x+1$ going to right that flipped after its collision or a particle at the same point going to left that did not collide.

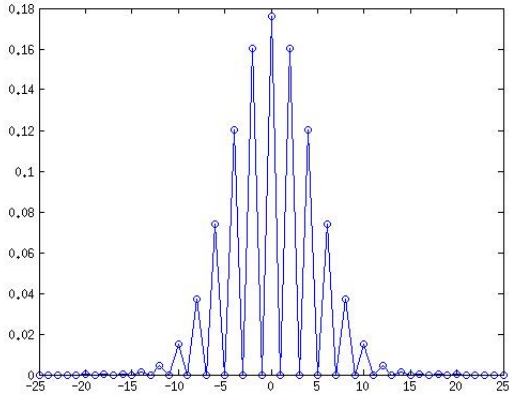


Figure 3.1: As example if we choose $p = q = 1/2$ in Eq.(3.2) we can construct this chart of probability distribution after 20 time steps. Then we can see that even after this period of time the highest probability corresponds to finding the particle at the same starting point.

Now let us see how to simulate this problem with cellular automata. For this problem we need to work with conservation of the number of particles, in this case only one particle. Furthermore we would like to see, in the simulation, the movement direction interpretation, therefore the best option for us is to work with the partition scheme, despite the fact we are not dealing with a non-deterministic dynamics.

We will describe the propagation of only one particle with two options of movement , right and left. Thus, in this scheme we only need to work with two subcells, $\Xi = \Sigma_l \times \Sigma_r$, where instead of calling by 0 and 1, we will call by left, (l), and right, (r), subcells, because of the movement interpretations. If we have a state given by $(1,0)$, this means that we have one particle going to left, and in the case of $(0,1)$ one particle going to right, as depicted in the Fig.(3.2).

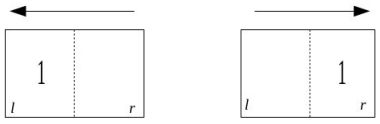


Figure 3.2: Representation of internal states in terms of particle motion.

The neighborhood scheme is $\mathcal{N} = \{x-1, x, x+1\}$, and, likewise the HPP model, we have to employ three local operators, the first that reads to interaction between the subcells that belong to the same cell, the second that reads to interaction between the cells and the third that establishes

the right motion interpretation of the particles. Therefore, we need to work with three tilings where again $\mathcal{T}_0 = \mathcal{T}_2$ and their elements are $T_x^{(0)} = \{x_l, x_r\}$ for $x \in \mathbb{Z}$ and $T_x^{(1)} = \{x_r, (x+1)_l\}$.

The main difference from the previous examples seen until now is in the first operator, σ_0, which is a convex sum of permutations,

$$\sigma_0 := p\,\mathbb{1}_2 + q\,\pi, \tag{3.3}$$

where $\mathbb{1}_2$ is a 2×2 identity operator which acts in every cell preserving its original state, while π swaps the information within the subcells, as example $\pi(0,1) = (1,0)$. Therefore, in summary we have the action of σ_0 given by the following equations

$$\begin{aligned}
\sigma_0 : (1,0) &\rightarrow (0,1) \quad \text{with probability q,} \\
\sigma_0 : (0,1) &\rightarrow (1,0) \quad \text{with probability q,} \\
\sigma_0 : (0,1) &\rightarrow (0,1) \quad \text{with probability p,} \\
\sigma_0 : (1,0) &\rightarrow (1,0) \quad \text{with probability p,}
\end{aligned} \tag{3.4}$$

and if we have no particles, $(0,0)$, nothing happens. The remaining operators σ_1 and σ_2 are just swap operators, σ_1 between neighbors cells and σ_2 again in the subcells from the same cell in order to keep the particles following the same direction, like we did in the HPP problem. We illustrated the action of these two operators in Fig.(3.3).

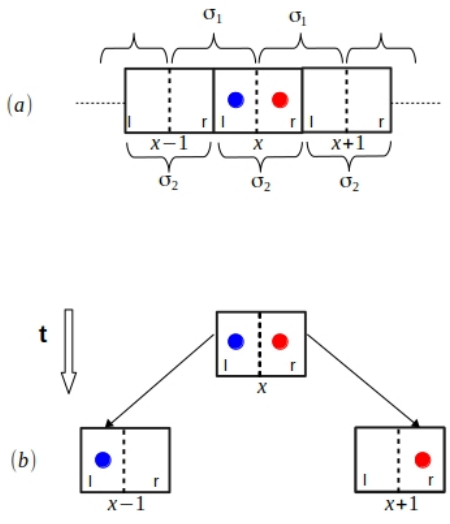


Figure 3.3: We can see in the figure above that if we have a particle in a left (right) subcell in a time t after we apply σ_1 and σ_2 Fig.-(a), it will keep at the same subcell in his left (right) neighbor at time $t+1$, Fig.-(b). It means that if we do not apply σ_0 in the next step the particle will continue in the same direction.

Now we will analyze this problem carefully with the introduction of Boolean functions and given the motion representation for each subcells, see Fig.(3.2).

In terms of Boolean functions what we have now is,

$$\begin{aligned}\sigma_0 : (n_l(x), n_r(x)) &\rightarrow (n_l(x), n_r(x)) \quad \text{with probability } p,\\ \sigma_0 : (n_l(x), n_r(x)) &\rightarrow (n_r(x), n_l(x)) \quad \text{with probability } q,\end{aligned}$$

where $q + p = 1$, and $n_l(x), n_r(x) \in \{0,1\}$. We have started with σ_0 and then we have to apply σ_1 and σ_2

$$\left(\prod_{T_x^{(2)} \in \mathcal{T}_2} \sigma_2\right)\left(\prod_{T_x^{(1)} \in \mathcal{T}_1} \sigma_1\right)(\ldots(n_l(x-1), n_r(x-1)), (n_l(x), n_r(x)), (n_l(x+1), n_r(x-1))\ldots)$$

$$= (\ldots(n_l(x), n_r(x-2)), (n_l(x+1), n_r(x-1)), (n_l(x+2), n_r(x))\ldots). \tag{3.5}$$

This means that we exchanged the values of the left and right subcells with the left register of the right neighbor, and the right register of the left neighbor, respectively. Subsequently, with the action of σ_2 at each cell, we were able to put the particles in the right subcell in order to preserve its movement direction.

Now let us focus in two properties of this evolution. Although $\mathscr{E}$ has three operators in its composition, these just give us one time step. The other important point is to note that after the action of σ_1 and σ_2 each new element, in each cell, needs to be reindexed. If we introduce t as a new parameter, where $t \in \mathbb{N}$, these two aspects can be realized in a simple way. For instance we can rewrite Eq.(3.5) as

$$\mathscr{E} : (n_l(x,t), n_r(x,t)) \rightarrow (n_l(x,t+1), n_r(x,t+1)).$$

As we do not know what happened in the action of σ_0 at time t, the value one of $n_l(x,t+1)$ could come from either the excitation localized in the point $x+1$ in the left subcell, that give to us a value one for $n_l(x+1,t)$ or the excitation localized in the point $x+1$ in the right subcell, that implies the value one for $n_r(x+1,t)$. We do not have this information, but we know if the particle came from the left subcell, the collision did not occur, that happens with probability p. Otherwise the particle was in the right subcell and the collision happened changing its direction, an event that happens with probability q. Therefore, we can codify these information available in the following expression

$$\begin{aligned} n_l(x-1,t+1) &= \big(1-\mu(x,t)\big)\, n_l(x,t) + \mu(x,t)\, n_r(x,t),\\ n_r(x+1,t+1) &= \mu(x,t)\, n_l(x,t) + \big(1-\mu(x,t)\big)\, n_r(x,t),\end{aligned} \tag{3.6}$$

where we introduced a Boolean variable statistically independent of $n_i(x,t)$, $\mu(x,t)$, in order to codify these probabilities. This variable assumes the value one with probability q, which leads us

to the right simulation. Although Eq.(3.6) and Eq.(3.1) are rather similar they have a different meaning. While Eq.(3.1) gives the evolution for the probability distribution, Eq.(3.6) yields a single realization of the simulation, Fig(3.1).

All realizations illustrated in Fig.(3.4) provide to us different trajectories. If we call by $\vec{r}_i(t)$ the i-th particle trajectory after a time t we have $\vec{r}_1(20)$, $\vec{r}_2(20)$, $\vec{r}_3(20)$ and $\vec{r}_4(20)$ distinct trajectories in Fig.(3.4). Thus the motion of the Brownian particle, in general, is not reproducible since we are not working with a deterministic dynamics. However, from these realizations, if we employ large number N,

$$\vec{r}_1(t), \vec{r}_2(t), \ldots, \vec{r}_N(t),$$

we can do probabilistic predictions. Indeed, from these single realizations we can recovery the same prediction showed in figure (3.1). In other words, we can calculate the probability $P(\vec{r},t)$ which is the probability of finding the particle at position $\vec{r}$ at time t, as we will see now.

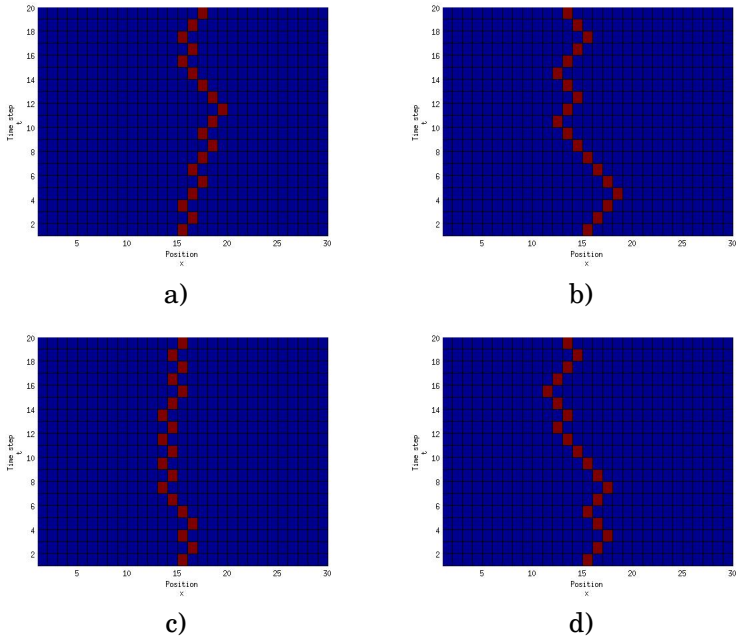
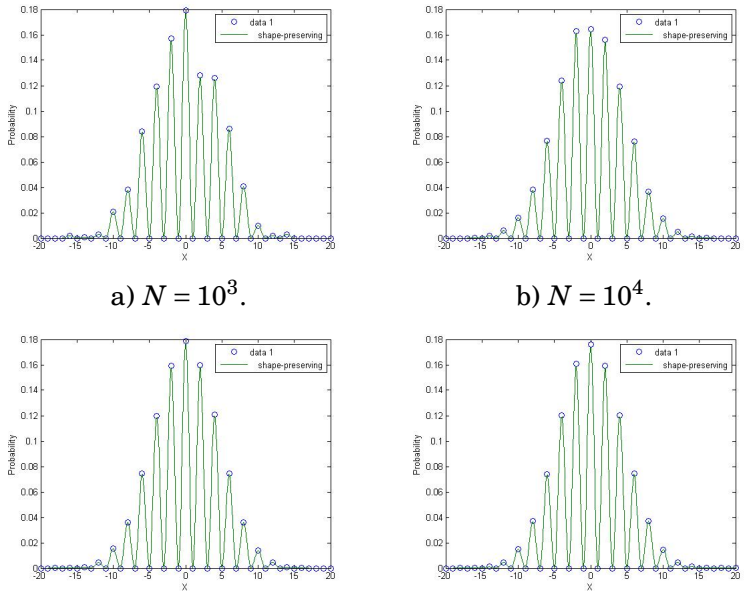


Figure 3.4: Despite the fact that equations (3.1) and Eq.(3.6) have very similar forms they have different meanings. The first gives the probability distribution, while the second provides us a simulation, which we used to do these four figures. In all figures we have taken 20 time steps for the excitation starting from $x = 15$ and $q = p = 1/2$.

The predictions are becoming closer and closer to (3.1) as the value of N increase. We can check these probabilities distribution for different N values in Fig.(3.5). The maximum correspondence with (3.1) happens for $N = 10^6$, which confirms that these realizations reproduce well the Brownian dynamics for the one dimensional case.



a) $N = 10^3$. b) $N = 10^4$.

c) $N = 10^5$. d) $N = 10^6$.

Figure 3.5: In these figures we can see four different probability distribution established from four distinct values for N. The predictions are becoming better as the value for N increase.

3.1.2 The continuous limit

Our goal now is to determine the continuous behavior of our cellular automaton, which describes the BM problem, expressed by Eq.(3.1).

From now on instead of working with discrete time and space, we will consider both continuous, $x, t \in \mathbb{R}$. Our focus now is to analyze the behavior of the automaton in the limit of infinitely short time step τ and a infinitesimal lattice spacing λ, and see which are the differential equations that describe the Brownian movement after we take the limit for λ and τ going to zero. The main steps that we will present here are contained in [17].

Imagine that we have already done several simulations, then the first step is to take the mean values

$$N_l(x,t) = \langle n_l(x,t) \rangle ; \quad N_r(x,t) = \langle n_r(x,t) \rangle , \tag{3.7}$$

thereby allowing this quantities to vary continuously between 0 and 1. These values are equivalent to the ones given by (3.1), but we decided to apply another notation since these quantities now were achieved by the average of the single realizations. Since $\mu(x,t)$ is statistically independent of either $n_l(x,t)$ or $n_r(x,t)$ then

$$\left\langle \mu(x,t)\, n_i(x,t) \right\rangle = \left\langle \mu(x,t) \right\rangle \langle n_i(x,t) \rangle = q N_i(x,t),$$

where $i = l, r$ and q the probability of not changing direction, Eq.(3.4). Therefore the relations (3.6) can be averaged and yield

$$\begin{aligned} N_l(x, t+\tau) &= qN_r(x+\lambda, t) + pN_l(x+\lambda, t), \\ N_r(x, t+\tau) &= qN_l(x-\lambda, t) + pN_r(x-\lambda, t), \end{aligned} \tag{3.8}$$

where $p = 1 - q$. As we can see these quantities are the continuous version of the ones showed in (3.1).

3.1.2.1 The Chapman-Enskog expansion

To proceed with the continuous limit analyses for the BM problem we will use the technique known as the Chapman-Enskog expansion [17] which is commonly used in statistical mechanics to derive the macroscopic laws. The right movement equation can only be achieved if we employ the right dispersion relation of the problem. For the Brownian motion the relation between the frequency and the wave number is given by $\omega(k) = ak^2$, where a is just a constant, see [17, 53], that implies λ^2/τ=constant. This relation plays an important role during the continuum limit computation as we will see.

The quantity N_i, $(i = r, l)$, defined in Eq.(3.7), plays the role of the Boltzmann density function [1] f, except that the velocities are labeled with a discrete index i instead of a continuous variable v. If we only integrate f by velocity variable we obtain a local density of particles $\rho(x, t)$. As we are not dealing with continuous variable v and we know the two possible velocities, right or left, the integration can be substituted by just one sum of these two terms

$$\rho(x, t) = N_r(x, t) + N_l(x, t). \tag{3.9}$$

The key of the Chapman-Enskog expansion is to assume that in the zero order perturbation we have $\rho(x, t) = N_r^{(0)}(x, t) + N_l^{(0)}(x, t)$. Then the Chapman-Enskog expansion makes use of the theory of perturbation for functions

$$N_i = N_i^{(0)} + \varepsilon N_i^{(1)} + \varepsilon^2 N_i^{(2)} + \ldots \quad \text{for } i = l, r; \tag{3.10}$$

where ε is a small parameter and $N_i^{(l)}$ are functions of x and t to be determined.

Firstly let us write again Eq.(3.8) but writing $1 - p$ instead q, then

$$\begin{aligned} N_r(x+\lambda, t+\tau) - N_r(x, t) &= (p-1)(N_r(x, t) - N_l(x, t)), \\ N_l(x-\lambda, t+\tau) - N_l(x, t) &= (p-1)(N_l(x, t) - N_r(x, t)). \end{aligned} \tag{3.11}$$

[1] The quantity $f(\vec{r}, \vec{v}, t)$ is a probability density function defined so that $f(\vec{r}, \vec{v}, t) d^3\vec{r} d^3\vec{v}$ is which all have positions lying within a volume element $d^3\vec{r}$ about $\vec{r}$ and velocity lying within a velocity element $d^3\vec{v}$ about $\vec{v}$, at time t. Integrating over a region of position space and velocity space gives the total number of particles which have positions and velocities in that region

We add both equations above and use Eq.(3.9) to establish

$$N_r(x+\lambda,t+\tau)+N_l(x-\lambda,t+\tau)-\rho(x,t)=0. \tag{3.12}$$

The result given by Eq.(3.12) reflects that the number of particles is locally conserved. For this reason this equation is called the **continuity equation**.

The next step is to consider a Taylor expansion of $N_r(x+\lambda,t+\tau)$ and $N_l(x-\lambda,t+\tau)$, where both λ and τ are infinitesimal parameters. If we neglect third-order terms we have

$$\begin{aligned} N_r(x+\lambda,t+\tau) &= N_r(x,t)+\tau\partial_t N_r(x,t)+\lambda\partial_x N_r(x,t)+\lambda\tau\partial_\tau\partial_\lambda N_r(x,t) \\ &+ \frac{1}{2}\tau^2\partial_t^2 N_r(x,t)+\frac{1}{2}\lambda^2\partial_x^2 N_r(x,t)+\mathscr{O}\left(\lambda^3,\tau^3\right). \end{aligned}$$

Rewriting this expansion to get the same left part in the equality (3.11), we obtain

$$N_r(x+\lambda,t+\tau)-N_r(x,t)=\left[\tau\partial_t+\lambda\partial_x+\lambda\tau\partial_t\partial_x+\frac{1}{2}\tau^2\partial_t^2+\frac{1}{2}\lambda^2\partial_x^2\right]N_r(x,t).$$

For $N_l(x-\lambda,t+\tau)$ a similar equation holds, but we should introduce a minus sign on odd orders terms for λ. Let us introduce a coefficient c_i, where $c_r=1$ and $c_l=-1$, which allow us write only one equation for both expansions

$$N_i(x+\lambda c_i,t+\tau)-N_i(x,t)=\left[\tau\partial_t+c_i\lambda\partial_x+\lambda\tau c_i\partial_t\partial_x+\frac{1}{2}\tau^2\partial_t^2+\frac{1}{2}\lambda^2 c_i{}^2\partial_x^2\right]N_i(x,t). \tag{3.13}$$

As we discussed in the beginning of this section we want to analyze this equation when both, time step and the lattice spacing are infinitely short. To put it in evidence we will continue our calculations employing the infinitesimal parameter ε for λ and τ, however to work with correct dimensionality we will continue write both λ and τ. As we briefly discussed in the beginning of this part $\lambda^2\sim\tau$, which is consequence of the BM dispersion relation. Therefore λ and τ are not the same order of magnitude in terms of ε. The previous dispersion relation can be establish if we write

$$\lambda\sim\varepsilon\lambda;\quad \tau\sim\varepsilon^2\tau. \tag{3.14}$$

Now we can use Eqs.(3.10-3.13) and compare, order by order the two sides of Eq.(3.11). As we are neglecting third order-terms we have

$$\begin{aligned} \left[\varepsilon^2\tau\partial_t+\varepsilon\lambda\partial_x+\frac{1}{2}\varepsilon^2\lambda^2\partial_x^2\right]\left[N_r^{(0)}+\varepsilon N_r^{(1)}+\varepsilon^2 N_r^{(2)}\right] &= (p-1)\Big(N_r^{(0)}+\varepsilon N_r^{(1)}+\varepsilon^2 N_r^{(2)} \\ &- N_l^{(0)}-\varepsilon N_l^{(1)}-\varepsilon^2 N_l^{(2)}\Big), \end{aligned}$$

in the same way

$$\begin{aligned} \left[\varepsilon^2\tau\partial_t-\varepsilon\lambda\partial_x+\frac{1}{2}\varepsilon^2\lambda^2\partial_x^2\right]\left[N_l^{(0)}+\varepsilon N_l^{(1)}+\varepsilon N_l^{(2)}\right] &= (p-1)\Big(N_l^{(0)}+\varepsilon N_l^{(1)}+\varepsilon^2 N_l^{(2)} \\ &- N_r^{(0)}-\varepsilon N_r^{(1)}-\varepsilon^2 N_r^{(2)}\Big). \end{aligned}$$

For the order $\mathcal{O}\left(\varepsilon^0\right)$,

$$N_r^{(0)} = N_l^{(0)} \tag{3.15}$$

consequently,

$$N_r^{(0)} = N_l^{(0)} = \frac{1}{2}\rho; \quad N_r^{(\mu)} + N_l^{(\mu)} = 0, \text{if} \mu \geq 1 \tag{3.16}$$

For the next order $\mathcal{O}(\varepsilon)$,

$$\lambda \partial_x N_r^{(0)} = (p-1)\left(N_r^{(1)} - N_l^{(1)}\right),$$

now from the results that we have established in Eqs.(3.16) $N_r^{(0)} = \rho/2$ and $N_l^{(1)} = -N_r^{(1)}$ yield

$$N_r^{(1)} = \frac{\lambda}{4(p-1)}\partial_x \rho, \tag{3.17}$$

$$N_l^{(1)} = -\frac{\lambda}{4(p-1)}\partial_x \rho. \tag{3.18}$$

Finally for the second-order $\mathcal{O}(\epsilon^2)$,

$$\begin{aligned}
\frac{1}{2}\tau\partial_t\rho + \frac{\lambda^2}{4(p-1)}\partial_x^2\rho + \frac{1}{4}\lambda^2\partial_x^2\rho &= (p-1)\left(N_r^{(2)} - N_l^{(2)}\right), \\
\frac{1}{2}\tau\partial_t\rho + \frac{\lambda^2}{4(p-1)}\partial_x^2\rho + \frac{1}{4}\lambda^2\partial_x^2\rho &= (p-1)\left(N_l^{(2)} - N_r^{(2)}\right),
\end{aligned}$$

where we have already used the previous results. Then, if we add these equations above we obtain

$$\partial_t\rho + \frac{\lambda^2}{\tau}\left(\frac{1}{2(p-1)} + \frac{1}{2}\right)\partial_x^2\rho = 0,$$

Therefore

$$\partial_t\rho + D\partial_x^2\rho = 0 \tag{3.19}$$

where D is the diffusion constant,

$$D = \frac{\lambda^2}{\tau}\left(\frac{p}{2(1-p)}\right).$$

For a better understanding of Eq.(3.19) we will solve it with a specif boundary condition. By taking the Fourier transform (FT) ρ,

$$\rho(x,t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty}\tilde{\rho}(k,t)e^{-ikx}dk,$$

from Eq.(3.19) we easily achieve in the differential equation in terms of $\tilde{\rho}$ below

$$\frac{\partial\tilde{\rho}}{\partial t} = -Dk^2\tilde{\rho}(k,t).$$

The solution is

$$\tilde{\rho}(k,t) = Ce^{-Dk^2t}, \tag{3.20}$$

where the integration constant C may still depend on k and, in general, is determined by initial conditions. We can see in the Eq.(3.20) that $C = \tilde{\rho}(k,0)$ is the FT of the initial spatial distribution of ρ, therefore it is given by $\rho(x,0)$.

A physical motivated choice for our problem is to consider that at time $t = 0$, all particles are located at the same point, $x = 0$, which means $\rho(x,0) = \delta(x)$. It is a quite well known that one representation of the Fourier transform of $\delta(x)$ is given by

$$\delta(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{-ikx} dk,$$

and then we conclude that $C = 1$. Therefore the local density of particle has the following format

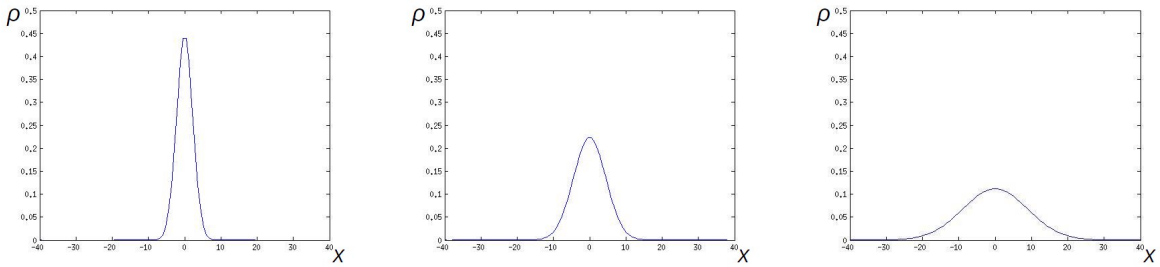
$$\rho(x,t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{-ikx} e^{-Dk^2 t} dk,$$

and finally we can write,

$$\rho(x,t) = \frac{1}{\sqrt{2tD}} e^{-\frac{x^2}{4Dt}}$$

From this analytical solution we can analyze ρ, which is the local particle density, for different values of t and p. If we work with small values of p this is means that we have higher probability of the particle flipping it movement direction at each collision. When we plot this case, as in the Fig.(3.6-a) for $t = 20$ the density of the particles is concentrated around the origin. However for a larger value of p we can see, like in the Fig.(3.6-c), the density of particles spreads faster, this because we have small probability of movement inversion.

It is also important to notice that in any case the peak of the density distribution is in the origin, independently of the value of p we choose.



a) $p = 0.2$ b) $p = 0.5$ c) $p = 0.8$

Figure 3.6: These figures give us ρ for different values of p.

3.2 A random walk automaton

Un contrast to the Brownian motion that was totally motivated by observations of random processes in nature, a random walk (RW) is a mathematical object, known as a stochastic dynamics, that describes a path that consists of a succession of random steps on some mathematical space

like the integers. The person who first introduced the name *random walk* was Karl Pearson, a mathematician and biostatistician, in the same year of the Einstein paper about the Brownian motion 1905. In this year Karl Pearson in a letter to Nature asked the solution to the problem stated as follows: "A man starts from a point O and walks l yards in a straight line; he then turns through any angle whatever and walks another l yards in a second straight line. He repeated this process n times. I require the probability that after these n stretches he is at a distance between r and $r+\delta r$ from his starting point, O." This letter was answered by Lord Rayleigh, who have already solved a similar problem to a more general context for *sounds waves in heterogeneous materials*.

The random walk model, which is an example of Markov process, plays an important role in several areas of physics, chemistry, biology and much more [81]. In particular, the phenomenon of diffusion or Brownian motion can be observed in this model. This equivalence between random walk and Brownian motion will be stressed during this section.

Now, as we have done in the previous section, we will start describing this problem more formally and only after we will encode it in the CA formalism.

Let us consider a walker who can only move in a straight line. She can move either to the right or left with probability p and $q = 1-p$ respectively. The step size in either case is l. The problem is to find the probability $P_N(m)$ of the walker to be at position $x = ml$, after N steps. The probability of some sequence of N steps, with N_1 to the right and N_2 to the left is given by $p^{N_1}q^{N_2}$. But there are different paths that can be taken for fixed N_1 and N_2. For example, in the case of $N = 3$ with $N_1 = 2$, we have p^2q. But she could start giving the first step to the left and then two to the right, or two steps to the right and one to the left, and also one step to the right, one to the left and again one to the right. This gives us three possible sequences with the same probability, or in other words, there are three distinct ways that give the same position to the walker. The number of different paths with fixed N_1 and N_2 is $N!/(N_1!N_2!)$. Therefore, the probability of she gives N_1 steps to the right and N_2 to the left is given by binomial distribution

$$W_N(N_1) = \frac{N!}{N_1!N_2!}p^{N_1}q^{N_2}, \tag{3.21}$$

where $N_1 + N_2 = N$ and $p + q = 1$[2]. The mean values for RW are

$$\langle N_1\rangle = pN, \quad \langle N_2\rangle = qN,$$

while the standard deviation is

$$\sigma_1 = (pq)^{1/2}\sqrt{N},$$

where $\sigma_1 = \sqrt{\langle N_1^2\rangle - \langle N_1\rangle^2}$. If the walker start at $x = 0$ then $m = N_1 - N_2$ and we find that the probability $P_N(m)$ is

$$P_N(m) = \frac{N!}{\left(\frac{N+m}{2}\right)!\left(\frac{N-m}{2}\right)!}p^{\frac{N+m}{2}}q^{\frac{N-m}{2}}, \tag{3.22}$$

[2]We can easily check that this probability is normalized, $\sum_{N_1=0}^{N} W_N(N_1) = (p+q)^N = 1$ and also that $0 \le W_N(N_1) \le 1$ for $0 \le N_1 \le N$

where we have also employed $N_1 + N_2 = N$.

If we consider the limit $N \to \infty$ in Eq.(3.21) we have $W_N(0) = q^N \to 0$ and $W_N(N) = p^N \to 0$. Then we can see that $W_N(N_1)$ must have a maximum for N_1. For large N, even for $N_1 \in \mathbb{Z}$, we can suppose that the function $W_N(N_1)$ is almost continuous near its maximum.

Let us go back to equation (3.22). If we carefully analyze $P_N(m)$ we can see the following recurrence relation

$$P_{N+1}(m) = pP_N(m-1) + qP_N(m+1).$$

If the walker's step is chosen to be $l = 1$, and if we associate to each walker's step a time step t, with $t \in \mathbb{N}$, we can rewrite this recurrence relation as

$$P(x,t+1) = pP(x-1,t) + qP(x+1,t). \tag{3.23}$$

From the equation above we easily conclude that the walker probability for each point $x \in \mathbb{Z}$ depends on the its neighbors probabilities, $P(x-1,t)$ and $P(x+1,t)$, each one multiplied by a constant, in the previous time. These constants are the walker probabilities to move to right p and to move to left q.

Now that we have already faced the RW problem we are ready to give its prescription by CA. As we have done for the Brownian motion we shall work with a partition scheme, where again the complete evolution is given by three tilings in $\mathscr{E}$. We also continue with the interpretation of movement, left or right, and then we will work with exactly the same operators σ_1 and σ_2 for BM. However we have to be prudent with the operator σ_0.

Now we are dealing with a particle (we will not call it by walker anymore) that have a probability p to go to right, which is independent of its internal state, right or left, and q to go to left. Then following this idea a good format for the operator σ_0 is

$$\begin{aligned}
\sigma_0 : (1,0) &\to (0,1) \quad \text{with probability p,} \\
\sigma_0 : (0,1) &\to (0,1) \quad \text{with probability p,} \\
\sigma_0 : (0,1) &\to (1,0) \quad \text{with probability q,} \\
\sigma_0 : (1,0) &\to (1,0) \quad \text{with probability q.}
\end{aligned} \tag{3.24}$$

Differently from the BM the operator σ_0 here is not a permutation, then we can not write (3.24) in terms of convex sum of permutations. As a next step we will see how employing these three operators σ_2, σ_1 and σ_0 we can simulate the RW by CA.

As we have done for the BM, we can write the recurrence relation for this problem. Like before we will call $n_l(x,t)$ being a Boolean variable which represents the left, l, internal state, and $n_r(x,t)$ being a Boolean variable which represents the right, r, internal state, where both are at site x in time t. Thus with the operators just presented for the RW problem it is a straightforward computation to get the following equations

$$\begin{aligned}
n_l(x-1,t+1) &= \mu(x,t)(n_l(x,t) + n_r(x,t)), \\
n_r(x+1,t+1) &= \big(1-\mu(x,t)\big)(n_l(x,t) + n_r(x,t)).
\end{aligned} \tag{3.25}$$

where $\mu(x,t)$ is statistically independent of $n_i(x,t)$ and returns one with probability q.

Before we look for the continuous limit for the RW, let us do some observations about these two problems that we have faced here, the Brownian motion and the random walk. If we chose $p = q = 1/2$ in both problems, RW and BM, we have the same results either in terms of probability ((3.2) and (3.23)) or simulations (Eq.(3.6) and (3.25)). This conclusion will be important for the continuous limit and we will come back to this discussion after we have established it.

3.2.1 The multiscale and Chapman-Enskog expansion

As we have done before the quantity $N_i, (i = r,l)$, which is the mean value of n_i, plays the role of the Boltzmann density function f, and again

$$\rho(x,t) = N_r(x,t) + N_l(x,t),$$

where ρ is the local density of particles. Again we work with theory of perturbations for functions Eq.(3.10), which is one of the main aspects of this expansion. From Eq.(3.25) we easily write its version in terms of N_i,

$$\begin{aligned} N_l(x-\lambda, t+\tau) &= q(N_l(x,t) + N_r(x,t)), \\ N_r(x+\lambda, t+\tau) &= p(N_l(x,t) + N_r(x,t)), \end{aligned}$$

where we have already assumed the fact that we are working with $x, t \in \mathbb{R}$, instead of discrete numbers, and $p + q = 1$. As we want to apply the Taylor expansion in both equations above, it is more convenient we write them as

$$\begin{aligned} N_l(x-\lambda, t+\tau) - N_l(x,t) &= -pN_l(x,t) + (1-p)N_r(x,t), \\ N_r(x+\lambda, t+\tau) - N_r(x,t) &= (p-1)N_r(x,t) + pN_l(x,t). \end{aligned}$$

If we add the both equations above we can see that the continuity equation is satisfied,

$$N_l(x-\lambda, t+\tau) + N_r(x+\lambda, t+\tau) - \rho(x,t) = 0.$$

As we have discussed before in the case that we have $p = q$, the RW reproduces the same behavior of the BM. From this observation we can expect that both problems to have the same dispersion relation for this specific case. However, if we only consider the previous dispersion relation for the RW, λ^2/τ, in advanced, we will get the same differential equation, (3.19), but with $p = 1/2$. Since this result does not give all possibilities of the behavior of the RW problem, for instance when $p \neq q$, as we see from (3.22), we will assume that there is another dispersion relation which is linear $\lambda/\tau = \text{constant} \neq 0$. From this prior knowledge about the problem we can work with the multiscale technique in a simple way.

As we have discussed we are working with two dispersion relations. This means that we should consider two macroscopic scales T_1 and T_2 satisfying

$$\frac{\tau}{T_1} = O(\varepsilon), \quad \frac{\tau}{T_2} = O\left(\varepsilon^2\right),$$

and one macroscopic length scale

$$\frac{\lambda}{L_1} = O(\varepsilon),$$

Therefore we introduce two time variables t_1 and t_2 such that

$$\partial_t = \partial_{t_1} + \varepsilon\partial_{t_2}, \tag{3.26}$$

which is the key of the multiscale expansion. To solve this problem we will adopt the following strategy: first we will solve the problem for the trivial dispersion relation, λ/τ = constant, applying t_1, following the same steps which we have done for the Brownian motion, afterwards we will solve for λ^2/τ = constant, applying t_2. Finally we put both results together by making use of the Eq.(3.26).

1. λ/τ = constant. In this first moment we will only be concerned about t_1;

$$\begin{aligned} N_i(x+\varepsilon\lambda c_i, t+\varepsilon\tau) - N_i(x,t) &= \Big[\varepsilon\tau\partial_{t_1} + c_i\varepsilon\lambda\partial_x + \varepsilon^2\lambda\tau c_i\partial_{t_1}\partial_x \\ &+ \frac{\varepsilon^2}{2}\tau^2\partial_{t_1}^2 + \varepsilon^2\tau\partial_{t_2} + \frac{\varepsilon^2}{2}\lambda^2 c_i{}^2\partial_x^2\Big] N_i(x,t). \end{aligned}$$

   Since ε is an infinitesimal parameter, and we are considering a first order, $\mathcal{O}(\varepsilon)$, approximation for this part, we only have to include the first two terms in the right part of the equation above:

$$\begin{aligned} \left[\varepsilon\tau\partial_{t_1} + \varepsilon\lambda\partial_x\right]\left[N_r^{(0)} + \varepsilon N_r^{(1)} + \varepsilon^2 N_r^{(2)}\right] &= (p-1)\left(N_r^{(0)} + \varepsilon N_r^{(1)} + \varepsilon^2 N_r^{(2)}\right) \\ &+ p\left(N_l^{(0)} + \varepsilon N_l^{(1)} + \varepsilon^2 N_l^{(2)}\right), \end{aligned}$$

   with the equation for N_l assuming the same form.

   For the order $\mathcal{O}\left(\varepsilon^0\right)$,

$$N_r^{(0)} = \frac{p}{1-p} N_l^{(0)},$$

   and again we assume that the $\rho(x,t) = N_r^{(0)}(x,t) + N_l^{(0)}(x,t)$ remains true, which implies

$$N_r^{(0)} = p\rho, \quad N_l^{(0)} = (1-p)\rho. \tag{3.27}$$

   For the next order $\mathcal{O}(\varepsilon)$, and last order that we will consider here

$$\begin{aligned} \tau\partial_{t_1} N_r^{(0)} + \lambda\partial_x N_r^{(0)} &= (p-1)N_r^{(1)} + pN_l^{(1)}, \\ \tau\partial_{t_1} N_l^{(0)} - \lambda\partial_x N_l^{(0)} &= -pN_l^{(1)} + (1-p)N_r^{(1)}. \end{aligned}$$

   Now making use of the relation

$$N_r^{(\mu)} + N_l^{(\mu)} = 0, \text{ if } \mu \geq 1, \tag{3.28}$$

and of the Eq.(3.27) we achieve

$$\begin{aligned} p\left(\tau\partial_{t_1}\rho+\lambda\partial_x\rho\right) &= -N_r^{(1)}, \\ (1-p)\left(\tau\partial_{t_1}\rho-\lambda\partial_x\rho\right) &= -N_l^{(1)}, \end{aligned}$$

and then adding these two equations we find

$$\partial_{t_1}\rho+v\partial_x\rho=0, \tag{3.29}$$

where

$$v=(2p-1)\frac{\lambda}{\tau}. \tag{3.30}$$

2. λ^2/τ = constant. Now we have to deal with t_2, which describes the dispersion of the Gaussian peak. Here we are in the case quite similar of the previous calculation Eq.(3.14), then it is not difficult to get

$$\begin{aligned} \left[\varepsilon^2\tau\partial_{t_2}+\varepsilon\lambda\partial_x+\frac{1}{2}\varepsilon^2\lambda^2\partial_x^2\right]\left[N_r^{(0)}+\varepsilon N_r^{(1)}+\varepsilon^2N_r^{(2)}\right] &= (p-1)\left(N_r^{(0)}+\varepsilon N_r^{(1)}+\varepsilon^2N_r^{(2)}\right) \\ &+ p\left(N_l^{(0)}+\varepsilon N_l^{(1)}+\varepsilon^2N_l^{(2)}\right), \end{aligned}$$

and

$$\begin{aligned} \left[\varepsilon^2\tau\partial_{t_2}-\varepsilon\lambda\partial_x+\frac{1}{2}\varepsilon^2\lambda^2\partial_x^2\right]\left[N_l^{(0)}+\varepsilon N_l^{(1)}+\varepsilon N_l^{(2)}\right] &= -p\left(N_l^{(0)}+\varepsilon N_l^{(1)}+\varepsilon^2N_l^{(2)}\right) \\ &+ (1-p)\left(N_r^{(0)}+\varepsilon N_r^{(1)}+\varepsilon^2N_r^{(2)}\right). \end{aligned}$$

For the order $\mathcal{O}\left(\varepsilon^0\right)$ we have the same result of the Eq.(3.27), however for the order $\mathcal{O}(\varepsilon)$ we can conclude something different

$$\begin{aligned} p\lambda\partial_x\rho &= -N_r^{(1)}, \\ (p-1)\lambda\partial_x\rho &= -N_l^{(1)}, \end{aligned} \tag{3.31}$$

and making the use of Eq.(3.28), after we added both equations above, we find

$$(2p-1)\lambda\partial_x\rho=0\Longrightarrow p=\frac{1}{2}, \tag{3.32}$$

otherwise ρ will be constant in space. The result above implies that in the t_2 time-scale the local density of particles going to right and left is the same, as in the case of BM.

Finally for the second-order $\mathcal{O}(\epsilon^2)$,

$$\begin{aligned} p\tau\partial_{t_2}\rho-\frac{p}{2}\lambda^2\partial_x^2\rho &= -N_r^{(1)}, \\ (1-p)\tau\partial_{t_2}\rho-\frac{(1-p)}{2}\lambda^2\partial_x^2\rho &= -N_l^{(1)}, \end{aligned}$$

where we have already used Eq.(3.31) in the equations above. Now, as before, we add these equations, divide the result by τ and take its continuum limit, which give us

$$\partial_{t_2}\rho - \bar{D}\partial_x^2\rho = 0, \tag{3.33}$$

where

$$\bar{D} = \frac{\lambda^2}{2\tau},$$

is the RW diffusion constant, which is independent of p.

Since these two results (3.29) and (3.33) are established independently of each other, from different dispersion relations, the probability dependence in the first equation does not infer in the second one. Now we can add both equations and use the Eq.(3.26) to get

$$\partial_t\rho + v\partial_x\rho - \bar{D}\partial_x^2\rho = 0. \tag{3.34}$$

We can now solve this equation for the same boundary conditions that we have done for Eq.(3.19) and compare their behavior. In the same way that we have for the BM we can use the well known Fourier transform technique to solve the differential equation above, using the same initial condition. After a few calculations we easily find,

$$\rho(x,t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{-ik(x-vt)}e^{-\bar{D}k^2t}dk,$$

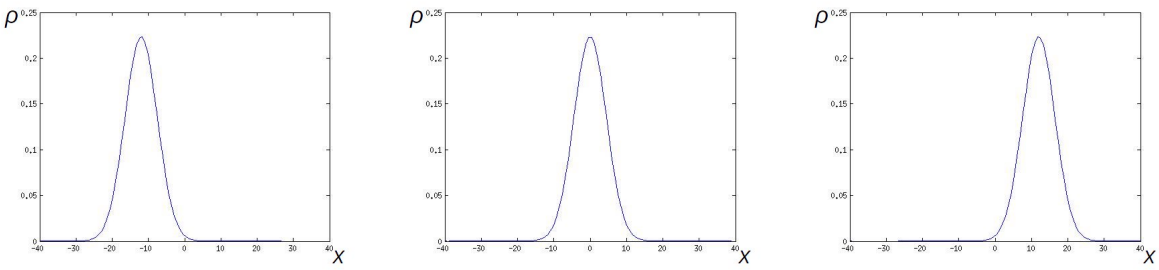
and therefore,

$$\rho(x,t) = \frac{1}{\sqrt{2t\bar{D}}}e^{-\frac{(x-vt)^2}{4\bar{D}t}}.$$

Now we can analyze this solution for the same values which we have analyzed for BM, $p = 0.2$, $p = 0.5$ and $p = 0.8$ for $t = 20$. We can see in Fig.(3.7-b) for $p = 1/2$ the BM and the RW have the same behavior, as we have already commented before. It is very important to notice that only for this value p this coincidence happens,

$$\mathrm{BM} \cap \mathrm{RW} = \left\{\frac{1}{2}\right\}.$$

Liwhich characterizes the difference between these models, except for one case. From plots a) and b) in Fig.(3.7-b), we can see a drift when we have $p \neq 1/2$. This behavior comes from the term $v\partial_x\rho$ in Eq.(3.34). Furthermore we can see from Eq.(3.30) that when $p = 1/2$ this term vanishes and then we are in the same case of the Brownian Motion Eq.(3.19).



a)$p = 0.2$ b)$p = 0.5$ c)$p = 0.8$

Figure 3.7: These figures give us ρ for different values of p.

CHAPTER 4

DIFFERENTIAL EQUATIONS VIA FINITE DIFFERENCE METHOD

Until here we focused in the cellular automata as the computational method. We saw that it can be employed for different problems in several distinct areas. In particular in chapter 3 we saw how the partitioned cellular automata (PCA) can be useful to simulate both the random walk and the Brownian motion problems. We could confirm by taking the continuous limit of these problems that in the end our discrete equation of motion are the discrete version of the stochastic linear differential equation that define these problems, first order in time and second order in space. In this sense we could see that CA is also an useful numerical tool, since it inherits all the characteristics of the simulated models, as the use of the local operators, homogeneity and so on. Although there are interesting connections between partial differential equation (PDE) and CAs, which is an object of study [75, 87], not always there is a straightforward translation from PDEs to CAs. While the PDEs are continuum-based models with the advantages of mathematical methods, which usually deal with systems with small numbers of degree of freedom, CA are rule-based methods with the advantages of local interactions, homogeneity, discrete states, and parallelism. Thus they are suitable for simulating systems with large degrees of freedom, as complex systems. But after we discretize some PDEs, using, for instance, the finite difference method (FDM) the discrete dynamics of the system achieved might be expressed by the same update rule employed by the CA. However, within CA the variables at each point of the grid are only allowed to range over a very small set, different from the real variables of the PDEs that were discretized. Although this drawback can be bypassed, it should be analyzed case-by-case, [75]. Then, we can say that if we have some partial differential equation, where we are not concerned with any particular physical aspects of the problem, for instance, the local interaction between the microscopic particles, and the system has only a few degrees of freedom, the best choice is to go to the ordinary numerical methods to solve differential equations, that is

our current topic.

The proposal of this chapter is to introduce the general concepts and ideas of the simplest and one of the oldest methods to solve differential equations, the finite difference method. The idea here is to make the reader familiar with this subject before they read chapter 9, where we propose a quantum algorithm to solve the wave equation. Then, for this reason we concentrate our examples and analyses on the wave equation problem.

This method was already known since 1768 by L. Euler [29] in the one dimensional space and was extended to a two dimensional space only in 1908 by C. Runge [65]. However their use for numerical applications and theoretical results only initiated in 1950s after the appearance of computers that stimulated their use for science and technology.

4.1 Methods of discretization

4.1.1 General principle

The idea of this method is to approximate the differential operator by replacing the derivatives using differential quotients. The domain is discretized either in space and time or only in space (time) and the numerical solution as well as the exact solution is computed at the space and time points.

There are three pillars that all numerical methods should be tested in order to certify if the method chosen is indeed good, which are **convergence**, **consistence** and **stability** as we will explain later. Through these pillars we have access to the errors associated with the employed method.

Basically, when we talk about errors we are interested in comparing the exact solution with the numerical one. Since the idea here is to apply the FDM for problems whose solutions we do not know, this comparison, at first sight, seems to be impossible. However, it can be accomplished by contrasting the differential operator with a difference operator, which has a special name, *truncation error*, that can be analyzed via Taylor series, as we will see.

Before we move to the next section let us introduce the notation used, and define the approximations for FDM. We discretize the domain into a lattice structure with the points

$$(x_j, t_n)$$

with

$$\begin{aligned} x_j &= jh, \quad j = 0, 1, \dots J, \\ t_n &= n\Delta t, \quad n = 0, 1, \dots N, \end{aligned}$$

and

$$\Delta t = \frac{T}{N}; \quad h = \frac{1}{J}.$$

Now given a continuous function, $\phi(x,t)$, $x \in (0,J)$ and $t \in (0,T)$, we define the following approximation for the derivatives of $\phi(x,t)$ at the point (x_j,t_n) on the lattice

$$\partial_x^+ \phi_j^n = \frac{\phi_{j+1}^n - \phi_j^n}{h}, \tag{4.1}$$

$$\partial_x^- \phi_j^n = \frac{\phi_j^n - \phi_{j-1}^n}{h}, \tag{4.2}$$

$$\partial_x^0 \phi_j^n = \frac{\partial_x^+ - \partial_x^-}{2}\phi_j^n = \frac{\phi_{j+1}^n - \phi_{j-1}^n}{2h}, \tag{4.3}$$

where ϕ_j^n is the approximate value of the function $\phi(x,t)$ in the point (x_j,t_n). We call Eq.(4.1) as the forward difference, Eq.(4.2) as the backward difference, and Eq.(4.3) as the central difference.

4.1.2 Taylor expansion

Back to the error truncation subject, we will see that this error comes from the fact that in the approximations defined above, forward (back) difference and central difference reflect the finite part of a Taylor series. Let us see this in more detail.

Assume that $\phi(x)$ is a C^1 twice differentiable function in the open segment $(a,a+h)$. Then for any $h>0$ we have

$$\phi(a+h) = \phi(a) + h\partial_x\phi(a) + \frac{h^2}{2}\partial_x^2\phi(a+\theta h), \tag{4.4}$$

where $\theta \in (0,1)$. The last derivative in the approximation above is known as Lagrange remainder. We can rewrite the equation above as

$$\phi(a+h) = \phi(a) + h\partial_x\phi(a) + \mathcal{O}\left(h^2\right),$$

which is more convenient to us since we are more interested in knowing the error order in the approximation. From the equation (4.4) we see that

$$\left|\frac{\phi(a+h)-\phi(a)}{h} - \partial_x\phi(a)\right| \leq Ch,$$

and from the forward approximation, we can see that

$$\left|\partial_x^+\phi(a) - \partial_x\phi(a)\right| \leq Ch, \tag{4.5}$$

where from Eq.(4.4) we see that,

$$C = \sup_{y\in[a,a+\theta a]} \frac{\left|\partial_x^2\phi(y)\right|}{2}.$$

Therefore, doing this simple analysis we could conclude that the truncation error for the forward difference is of order h. A similar analysis can be done for the backward difference to conclude that the truncation error is the same.

Now let us suppose that $\phi(x)$ is C^2, 3 times differentiable in the vicinity Ω of a, $\Omega = (a-h, a+h)$, then

$$\begin{aligned}\phi(a+h) &= \phi(a)+h\partial_x\phi(a)+\frac{h^2}{2}\partial_x^2\phi(a)+\frac{h^3}{6}\partial_x^3\phi(a+\theta h),\\ \phi(a-h) &= \phi(x)-h\partial_x\phi(a)+\frac{h^2}{2}\partial_x^2\phi(a)-\frac{h^3}{6}\partial_x^3\phi(a-\theta h),\end{aligned}$$

Subtracting these two expressions and by an analogous manipulation done for the forward difference, we obtain

$$\left|\frac{\phi(a+h)-\phi(a-h)}{2h}-\partial_x\phi(a)\right| \leq Ch^2.$$

Now from the central difference

$$\left|\partial_x^0\phi(a)-\partial_x\phi(a)\right| \leq Ch^2, \tag{4.6}$$

where

$$C = \sup_{y\in[a-\theta h, a+\theta h]}\frac{\left|\partial_x^3\phi(y)\right|}{2}.$$

Thus, now we established an approximation for the first derivative of order h^2, which is a better approximation compared with the previous ones.

Now we will combine the operator (4.1) with (4.2) in order to approximate the second derivative. After a straightforward computation we can write the follow result,

$$\partial_x^-\partial_x^+\phi_j^n = \frac{\phi_{j+1}^n-2\phi_j^n+\phi_{j-1}^n}{h^2}. \tag{4.7}$$

Like we did for the first derivative we can work with the Taylor expansion to see the error related with the approximation given by Eq.(4.7). In order to do it we suppose that $\phi(x)$ is a C^3 function 4 times differentiable on an interval $(a-\theta h, a-\theta h)$. Now from the Taylor expansion of $\phi(a)$

$$\begin{aligned}\phi(a+h) &= \phi(a)+h\partial_x\phi(a)+\frac{h^2}{2}\partial_x^2\phi(a)+\frac{h^3}{6}\partial_x^3\phi(a)+\frac{h^4}{24}\partial_x^4\phi(a+\theta h),\\ \phi(a-h) &= \phi(a)-h\partial_x\phi(a)+\frac{h^2}{2}\partial_x^2\phi(a)-\frac{h^3}{6}\partial_x^3\phi(a)+\frac{h^4}{24}\partial_x^4\phi(a-\theta h),\end{aligned}$$

Then, similarly to the previous computation, after we add these two expressions, we get

$$\left|\partial_x^-\partial_x^+\phi(x)-\partial_x^2\phi(a)\right| \leq Ch^2, \tag{4.8}$$

with

$$C = \sup_{y\in[a-\theta h, a+\theta h]}\frac{\left|\partial_x^4\phi(y)\right|}{12}.$$

Then we conclude that this approximation has the same error order that the central difference, h^2.

The same analysis can be done for the time part. Although the extension for higher dimension can be done without difficulty, in this chapter we will remain restricted to the one dimension examples.

4.2 The wave equation problem

The wave phenomenon can be seen everywhere on nature, from water waves and light waves to gravitational waves. Therefore this phenomenon is extremely important in physics. Mathematically all these phenomena can be described by a linear second-order partial differential equation

$$\frac{\partial^2\phi(x,t)}{\partial t^2} = v^2\frac{\partial^2\phi(x,t)}{\partial x^2}, \tag{4.9}$$

which describes the propagation of oscillations at a fixed speed v in some spatial degree of freedom x. This equation is known as the **wave equation**. This equation can be employed to widely different phenomena since we can use it to model small oscillations about an equilibrium, which is why systems can often be well approximated by Hooke's law. Indeed the one-dimensional wave equation can be derived from Hooke's law.

Eq.(4.9) itself does not determine a physical solution. We need to work with further conditions, such as the **initial conditions**. For the cases that we want to describe we demand the use of the **boundary conditions** where the solutions represent, for instance, *standing waves* and *harmonics*.

Although working with Eq.(4.9) we do not have access to its solution, all solutions to the wave equation are superpositions of "left-traveling" $f(x+vt)$ and "right-traveling" $g(x-vt)$ waves, which is a straightforward computation to check. Moreover, since this equation is linear, any superposition of solutions to the wave equation are also solutions.

Before we move to the numerical analysis of the problem, let us apply the approximate derivatives (4.7) for time and space to the wave equation and see how it works in practice.

We start employing (4.7) into (4.9)

$$\partial_t^-\partial_t^+\phi_j^n = v^2\partial_x^-\partial_x^+\phi_j^n, \tag{4.10}$$

or equivalently,

$$\frac{\phi_j^{n+1}-2\phi_j^n+\phi_j^{n-1}}{\Delta t^2} = v^2\frac{\phi_{j+1}^n-2\phi_j^n+\phi_{j-1}^n}{h^2}.$$

Doing a quick manipulation in the equation above we get

$$\phi_j^{n+1} = 2(1-s)\phi_j^n + s\left(\phi_{j+1}^n+\phi_{j-1}^n\right) - \phi_j^{n-1}, \tag{4.11}$$

with

$$s = \frac{v^2\Delta t^2}{h^2}. \tag{4.12}$$

However, as we said before, in order to determine the solution we need to add extra conditions, that are the initial condition and the boundary condition. The idea here is to provide an approximate solution to the standing wave, where the domain is $\Omega = [0,1]$ for all $t \in [0,T]$. Physically we can

think of this problem as the strings of a harp that are fixed on both ends to the harp's frame, whose oscillation are described by the solution of the wave equation. The way to deal with the fixed-endpoint is to work with the Dirichlet boundary condition,

$$\phi(0,t) = \phi(1,t) = 0. \tag{4.13}$$

This condition in the discrete notation is given by

$$\phi_0^n = \phi_J^n = 0. \tag{4.14}$$

As the initial condition we choose

$$\phi(x,0) = \sin(\pi x),\ x \in \Omega, \tag{4.15}$$

and from the notation adopted here,

$$\begin{aligned} \phi_j^0 &= \sin(\pi x), \\ &= \sin(\pi j h), \\ &= \sin\left(\frac{\pi j}{J}\right). \end{aligned}$$

For the wave equation problem, which is a second order differential equation, the initial condition has to include the first oder derivative in time. For our example we select the following initial condition

$$\frac{\partial \phi(x,0)}{\partial t} = 0\, x \in \Omega. \tag{4.16}$$

Since we are working with approximate derivatives we need to rewrite the condition above. In order to make the numerical analysis easier we pick the central difference Eq.(4.3), that gives

$$\partial_t^0 \phi_j^0 = \frac{\phi_j^1 - \phi_j^{-1}}{2\Delta t} = 0.$$

This approximation allows us solve the "ghost" point ϕ_j^{-1}, the point which does not belongs to the time domain

$$\phi_j^{-1} = \phi_j^1. \tag{4.17}$$

Now let us return to Eq.(4.11). The idea is to solve this equation for ϕ_j^{n+1} employing the initial and boundary conditions. The best way to deal with this problem is to rewrite the equation in a matrix format, as follows

$$\begin{bmatrix} \phi_0^{n+1} \\ \phi_1^{n+1} \\ \vdots \\ \phi_k^{n+1} \\ \vdots \\ \phi_{J-1}^{n+1} \\ \phi_J^{n+1} \end{bmatrix} = \begin{bmatrix} 2(1-s) & s & 0 & \cdots & & 0 \\ s & 2(1-s) & s & 0 & \cdots & 0 \\ \cdots & & \vdots & & \cdots & \\ \vdots & s & 2(1-s) & s & & \vdots \\ \cdots & & \vdots & & \cdots & \\ \vdots & 0 & 0 & s & 2(1-s) & s \\ 0 & \cdots & & 0 & s & 2(1-s) \end{bmatrix} \begin{bmatrix} \phi_0^n \\ \phi_1^n \\ \vdots \\ \phi_k^n \\ \vdots \\ \phi_{J-1}^n \\ \phi_J^n \end{bmatrix} - \begin{bmatrix} \phi_0^{n-1} \\ \phi_1^{n-1} \\ \vdots \\ \phi_k^{n-1} \\ \vdots \\ \phi_{J-1}^{n-1} \\ \phi_J^{n-1} \end{bmatrix} \tag{4.18}$$

because the Dirichlet boundary conditions we have ϕ_0^{n+1} and ϕ_J^{n+1} always zero, then the best option for us is writing

$$\begin{bmatrix} \phi_1^{n+1} \\ \vdots \\ \phi_k^{n+1} \\ \vdots \\ \phi_{J-1}^{n+1} \end{bmatrix} = \begin{bmatrix} 2(1-s) & s & 0 & \cdots & 0 \\ \cdots & & \vdots & & \cdots \\ \vdots & s & 2(1-s) & s & \vdots \\ \cdots & & \vdots & & \cdots \\ 0 & \cdots & & & 2(1-s)s \end{bmatrix} \begin{bmatrix} \phi_1^{n} \\ \vdots \\ \phi_k^{n} \\ \vdots \\ \phi_{J-1}^{n} \end{bmatrix} - \begin{bmatrix} \phi_1^{n-1} \\ \vdots \\ \phi_k^{n-1} \\ \vdots \\ \phi_{J-1}^{n-1} \end{bmatrix}. \tag{4.19}$$

When we start with $n = 0$ we have to take the values of ϕ_j^0 and $\phi_j^{-1} = \phi_j^1$, and then we start from

$$\begin{bmatrix} \phi_1^{1} \\ \vdots \\ \phi_k^{1} \\ \vdots \\ \phi_{J-1}^{1} \end{bmatrix} = \begin{bmatrix} 2(1-s) & s & 0 & \cdots & 0 \\ \cdots & & \vdots & & \cdots \\ \vdots & s & 2(1-s) & s & \vdots \\ \cdots & & \vdots & & \cdots \\ 0 & \cdots & & s & 2(1-s) \end{bmatrix} \begin{bmatrix} \phi_1^{0} \\ \vdots \\ \phi_k^{0} \\ \vdots \\ \phi_{J-1}^{0} \end{bmatrix} - \begin{bmatrix} \phi_1^{1} \\ \vdots \\ \phi_k^{1} \\ \vdots \\ \phi_{J-1}^{1} \end{bmatrix}$$

or

$$\begin{bmatrix} \phi_1^{1} \\ \vdots \\ \phi_k^{1} \\ \vdots \\ \phi_{J-1}^{1} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 2(1-s) & s & 0 & \cdots & 0 \\ \cdots & & \vdots & & \cdots \\ \vdots & s & 2(1-s) & s & \vdots \\ \cdots & & \vdots & & \cdots \\ 0 & \cdots & & s & 2(1-s) \end{bmatrix} \begin{bmatrix} \phi_1^{0} \\ \vdots \\ \phi_k^{0} \\ \vdots \\ \phi_{J-1}^{0} \end{bmatrix}. \tag{4.20}$$

Eq.(4.20) is our starting point for our numerical analysis as long as we fix some value for J and s. Applying this equation in order to get the vector points related with $t = 1$, ϕ_k^1 with $k = 1, \ldots, J-1$ and returning to Eq.(4.19) to get our next points, $t = 2$, which depends on vectors at time $t = 1$ and $t = 0$, we can approximate the solution for successive times, which is our next step.

4.2.1 Numerical analysis

Before we start with the numerical tests let us explain better the three pillars, mentioned previously, that our method needs to satisfy.

In order to understand these concepts in an easy way, we will analyze a time-independent differential equation. Suppose that we have the following second order differential equation to solve

$$\frac{d^2u(x)}{dx^2} = f(x) \quad \text{for } 0 < x < 1, \tag{4.21}$$

with some given boundary conditions $u(0) = \alpha$ and $u(1) = \beta$. The function $f(x)$ is specified and we wish to compute $u(x)$ numerically. In other words, we attempt to compute a grid function

consisting of values u_0, u_1, ..., u_m, u_{m+1}, where u_j is our approximation to the solution $u(x_j)$, where from the boundary conditions we have $u_0 = \alpha$ and $u_{m+1} = \beta$. Now, let us replace d^2u/dx^2 by the centered difference approximation Eq.(4.7), then we obtain a set of algebraic equations

$$\frac{1}{h^2}\left(u_{j-1} - 2u_j + u_{j+1}\right) = f\left(x_j\right) \quad \text{for } j = 1, 2, \ldots, m. \tag{4.22}$$

From the equation above we should notice that the equation for $j = 1$ involves the values $u_0 = \alpha$ and the last equation $j = m$ involves $u_{m+1} = \beta$. Then we have a linear system to solve, which can be written in the form

$$AU = F, \tag{4.23}$$

where U is the vector of unknowns $U = [u_1, \ldots, u_m]^T$ and

$$\frac{1}{h^2}\begin{bmatrix} -2 & 1 & 0 & \cdots & 0 \\ \cdots & & \vdots & & \cdots \\ \vdots & 1 & -2 & 1 & \vdots \\ \cdots & & \vdots & & \cdots \\ 0 & \cdots & & 1 & -2 \end{bmatrix}, \quad F = \begin{bmatrix} u_1 \\ \vdots \\ u_j \\ \vdots \\ u_{m-1} \end{bmatrix}.$$

We also can define a vector of the true values $\hat{U} = [u(x_1), \ldots, u(x_m)]$. From Eq.(4.8) we know that the true solution $u(x_j)$ will not satisfy the Eq.(4.21) exactly. Then we write this discrepancy by τ_j, which is called truncation error

$$\tau_j = \frac{1}{h^2}\left(u\left(x_{j-1}\right) - 2u\left(x_j\right) + u\left(x_{j+1}\right)\right) - f\left(x_j\right),$$

where again from Eq.(4.8) we know that the order of the truncation error is $\mathcal{O}(h^2)$. Now if we define τ to be the vector with components τ_j, we have

$$\tau = A\hat{U} - F,$$

or

$$A\hat{U} = F + \tau. \tag{4.24}$$

The last vector that we need to define before we introduce the concepts of stability, consistency, and convergence is the error vector

$$E = U - \hat{U}, \tag{4.25}$$

which contains the errors at each grid point, $E_j = u(x_j) - u_j$. Then, from Eqs.(4.23) and (4.24) we have

$$AE = -\tau. \tag{4.26}$$

- **Stability:** in order to understand the concept of stability we will rewrite the system (4.26) in the form

$$A^h E^h = -\tau^h, \tag{4.27}$$

where the superscript h indicates that we are on a grid with mesh spacing h. This serves as a reminder that these quantities changes as we refine the grid.

Let $(A^h)^{-1}$ be the inverse of this matrix. Then solving the system (4.27) gives

$$E^h = -(A^h)^{-1}\tau^h,$$

and taking its norms, see App.B, gives

$$\begin{aligned} \left\|E^h\right\| &= \left\|\left(A^h\right)^{-1}\tau^h\right\| \\ &\leq \left\|\left(A^h\right)^{-1}\right\|\left\|\tau^h\right\|. \end{aligned}$$

We know that $\left\|\tau^h\right\| = \mathcal{O}\left(h^2\right)$ and we expect the same to $\left\|E^h\right\|$. Because of it we need that $\left\|\left(A^h\right)^{-1}\right\|$ to be bounded by some constant that does not depend of h as $h \to 0$

$$\left\|\left(A^h\right)^{-1}\right\| \leq C \text{ for all } h \text{ sufficiently small.}$$

Then we will have

$$\left\|E^h\right\| \leq C\left\|\tau^h\right\|, \tag{4.28}$$

and so $\left\|E^h\right\|$ goes to zero as fast as $\left\|\tau^h\right\|$. Now we are in conditions to introduce the definition for *stability*.

Definition 4.1. Suppose a finite difference method for a linear boundary value problem gives a sequence of matrix equations of the form $A^h U^h = F^h$, where h is the mesh width. We say that the method is stable if $(A^h)^{-1}$ exists for all h sufficiently small and if there is a constant C, independent of h, such that

$$\left\|\left(A^h\right)^{-1}\right\| \leq C \text{ for all } h \leq h_0. \tag{4.29}$$

- **Consistency:** We say that a method is *consistent* with the differential equation and boundary conditions if

$$\left\|\tau^h\right\| \to 0 \text{ as } h \to 0. \tag{4.30}$$

This simply says that we have a sensible discretization of the problem. Typically $\left\|\tau^h\right\| = \mathcal{O}(h^p)$ for some integer $p > 0$, and then the method is certainly consistent.

- **Convergence:** A method is said to be *convergent* if $\left\|E^h\right\| \to 0$ as $h \to 0$. Combining the ideas introduced above we arrive at the conclusion that

  $$\text{consistency+stability} \Rightarrow \text{convergence.} \tag{4.31}$$

  It is straightforward to prove it using Eqs. (4.29) and (4.30)

  $$\left\|E^h\right\| \leq C \left\|\tau^h\right\| \to 0, \text{ as } h \to 0. \tag{4.32}$$

  for a given norm $\|\cdot\|$.

Moreover it is known that from the Lax-Richtmyer Equivalence Theorem [43], which is often called the Fundamental Theorem of Numerical Analysis, that always holds for linear differential equation.

$$\text{consistency+stability} \Longleftrightarrow \text{convergence} \tag{4.33}$$

Despite the fact that the problem that we decided to analyze has analytical solution,

$$\phi(x,t) = \cos(\omega t)\sin(\pi x), \tag{4.34}$$

we will not take advantage from this fact in the certification of this method. The strategy that we will adopt in this section is to start with the analytical analyses for the consistency and stability for the wave equation. Subsequently we work with the numerical example of this problem. Finally, in the end we will see numerically that indeed this method is convergent.

We begin with the consistency. From what we have seen in, (4.8) and from the approximation (4.10), it is easy to see that

$$\frac{\phi\left(x_j, t_n + \Delta t\right) - 2\phi\left(x_j, t_n\right) + \phi\left(x_j, t_n - \Delta t\right)}{\Delta t^2} = \frac{\partial^2 \phi(x,t)}{\partial t^2} + C_1 \Delta t^2$$

and

$$\frac{\phi\left(x_j + h, t_n\right) - 2\phi\left(x_j, t_n\right) + \phi\left(x_j - h, t_n\right)}{h^2} = \frac{\partial^2 \phi(x,t)}{\partial x^2} + C_2 h^2$$

with

$$C_1 = \sup_{y \in [x-\theta h, x+\theta h]} \frac{\left|\partial_x^4 \phi(y,t)\right|}{12}; \; C_2 = \sup_{\Delta t' \in [t-\Delta\xi, t+\Delta\xi]} \frac{\left|\partial_t^4 \phi\left(x,t'\right)\right|}{12},$$

where $\theta \in (0,1)$. $\phi(x,t)$ is C^2, 3 times differentiable in space and time in the vicinity of x, $\Omega = (x-h, x+h)$ and in the vicinity of t, $T = (t-\Delta t, t+\Delta t)$, respectively. Then,

$$\frac{\phi\left(x_j, t_n + \Delta t\right) - 2\phi\left(x_j, t_n\right) + \phi\left(x_j, t_n - \Delta t\right)}{\Delta t^2} - v^2 \frac{\phi\left(x_j + h, t_n\right) - 2\phi\left(x_j, t_n\right) + \phi\left(x_j - h, t_n\right)}{h^2}$$

$$= \frac{\partial^2 \phi(x,t)}{\partial t^2} - v^2 \frac{\partial^2 \phi(x,t)}{\partial x^2} + \tau_h^n,$$

where

$$\tau_h^n = C_1 h^2 + C_2 \Delta t^2, \tag{4.35}$$

We should keep in mind that while $\phi(x_j, t_n)$ represents the precise value of the function $\phi(x,t)$ in the point (x_j, t_n) the quantity ϕ_j^n represents the approximate value in this same point. Thus, it is clear that this method is consistent Eq.(4.30), since $\tau_h^n \to 0$ when h and Δt go to zero.

Now let us check if we can find a bound for the error

$$e_j^n = \phi(x_j, t_n) - \phi_j^n. \tag{4.36}$$

and if $\left\| e^h \right\| \to 0$ when $h \to 0$ in order to see if the method is convergent.

It is clear now that we are working with a second order approximation method, then if we rewrite our wave equation in terms of the values that $\phi(x,t)$ assumes at each point (x_j, t_n) we have

$$\partial_t^- \partial_t^+ \phi\left(x_j, t_n\right) - v^2 \partial_x^- \partial_x^+ \phi\left(x_j, t_n\right) = \tau,$$

where

$$\tau \approx \mathcal{O}\left(h^2, \Delta t^2\right).$$

As a next step we use the approximation expression in the equation above in order to get the recurrence relation for $\phi(x_j, t_n + \Delta t)$ thus,

$$\phi\left(x_j, t_n + \Delta t\right) = 2(1-s)\phi\left(x_j, t_n\right) + s\left(\phi\left(x_j + h, t_n\right) + \phi\left(x_j - h, t_n\right)\right) - \phi\left(x_j, t_n - \Delta t\right) + \Delta t^2 \tau, \tag{4.37}$$

Now we can take the difference between (4.37) and (4.11) with the error definition (4.36) to establish the following expression

$$e_j^{n+1} = 2(1-s)e_j^n + s\left(e_{j+1}^n + e_{j-1}^n\right) - e_j^{n-1} + \Delta t^2 \tau. \tag{4.38}$$

From this equation above we can recursively start to analyse how the errors increase with time, however we can not estimate any bound for the error from Eq.(4.38) since there are extra conditions that should be considered. However, we can find a rigorous analysis of the stability condition, and then a convergent method once we have already seen that the method is consistent, of the wave equation in [43]. There they could conclude, for instance, that stability reads as long as

$$1 - s > 0.$$

From the s definition this implies

$$\frac{v^2 \Delta t^2}{h^2} < 1.$$

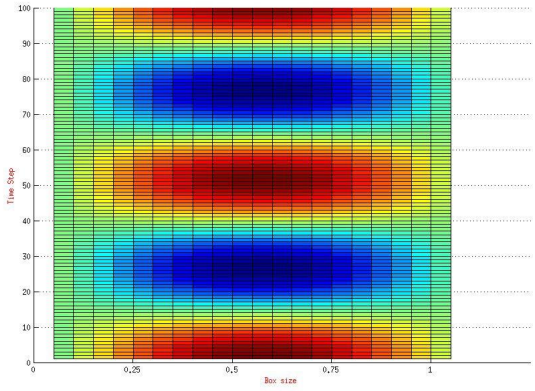
Then, from this result above we can conclude that the method needs to obey

$$\Delta t < \frac{h}{v}. \tag{4.39}$$

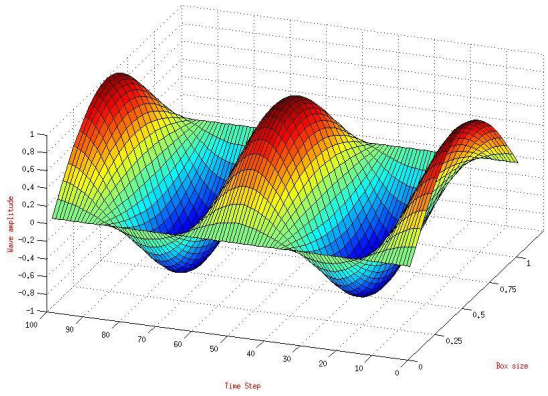
We will go back to this condition later.

Then, from the Lax-Richtmyer theorem, we can conclude that our method is indeed convergent.

Now we are ready to work with a concrete example. Let us get back to equation (4.19), where we only need to choose the values for $h, \Delta t$ and v. We fix $v = 1$ in the examples presented here. In the first example fig.(4.1) we worked with $\Delta t = 0.04s$ and $J = 20$ that implies $h = 0.05$. Employing these parameters in this numerical analysis we can see the stability criterion satisfied, since $\Delta t/h = 0.8 < 1$.



a)



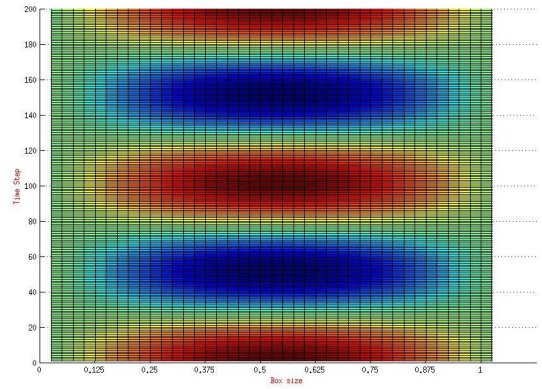
b)

Figure 4.1: In these figures we can see two different views from the standing wave simulation.

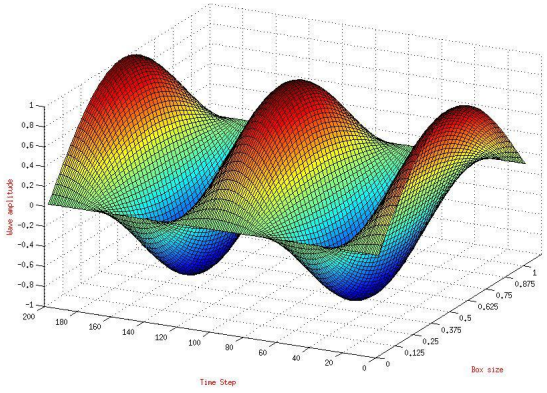
We can confirm from the results showed in fig.(4.1) that our approximation method could reproduced the standing wave behavior. Since we took $\Delta t = 0.04s$ these 100 time steps means that these oscillations happened during four seconds, $t = 4s$.

For the next example we kept the same problem, but we refined the lattice, that means we employed a smaller value for h. In fig.(4.2) we can see the same behavior, but now with $h' = 0.025$. In order to maintain the same value 0.8, that gives us a stable simulation as before, we had to use

$\Delta t' = 0.02s$. Thus, for the same oscillation pattern that happens in four seconds in the previous example example we had to work with 200 time steps when we employ a smaller time interval and we want to keep the simulation as stable as the previous one, $\Delta t'/h' = 0.8$.



a)



b)

Figure 4.2: As in fig.(4.1) we have two different views from the standing wave simulation, but with different values for Δt and h.

One of the reasons to show these two numerical examples is to see how the computational costs can increase if we need to work with smaller and smaller values for h and we want to maintain the simulation with the same stability order.

Before we finish this chapter we will confirm that our numerical examples are indeed convergent. Previously we showed that our method is convergent by analytical analyses, but now we will see it via numerical computation.

4.2.1.1 Q-factor

There are few different ways to analyze the convergence of some numerical method employed. We can, for instance, take the maximum norm of equation (4.36) and start to analyze the behavior of

the expression $\|e^n\|_\infty$ for different values of h, respecting the condition established for the stability criteria (4.39). Thus, we will see that our method is convergence as long as the convergence rates go to zero when $h \to 0$. However, we adopted a different method to analyze the convergence of our simulation, the same used in chapter (9). The numerical method employed here, in order to see the convergence, is called by Q *factor*. The main reason for this choice is that with this method we can predict the values for Q analytically for each approximate derivative used in the numerical simulations. Thus, since we also employed higher order Laplacians in [21], that give us different values for Q for each Laplacian order employed, we thought that this method would be more convenient for our results.

The Q factor is a method where its construction is based by the Richardson expansion, which is a general procedure for improving the accuracy of approximations when the structure of the error is known [45]. This method is used to quantify discretization errors in numerical simulation, [18].

To compute this factor we used the discretized solutions at three different lattice spacings ϕ^h, ϕ^{2h} and ϕ^{4h}. The Q factor is then defined by

$$Q(t) = \frac{\left\|\phi^{4h} - \phi^{2h}\right\|_2}{\left\|\phi^{2h} - \phi^{h}\right\|_2}. \tag{4.40}$$

Now we can use the Richardson expansion

$$\begin{aligned} \phi^h &= \phi(x,t) + h^2 E_2(x,t) + h^4 E_4(x,t) + \dots \\ \phi^{2h} &= \phi(x,t) + (2h)^2 E_2(x,t) + (2h)^4 E_4(x,t) + \dots \\ \phi^{4h} &= \phi(x,t) + (4h)^2 E_2(x,t) + (4h)^4 E_4(x,t) + \dots \end{aligned} \tag{4.41}$$

where E_i are the errors functions that do not depend on h, to predict the values that Q can have. In fact, these errors functions are related with the derivative order associated with the truncated error of the approximation used, the values C showed in section (4.1.2). Since we are aware that we used a second order method for the wave equation simulation (4.35) we expect

$$\lim_{h\to 0} Q(t) = 4, \tag{4.42}$$

if our method is convergent, check this is a straightforward computation, as we will see now. For the second order method we have E_2 as a dominant error function, thus computing the numerator in Eq.(4.40), we get

$$\phi^{4h} - \phi^{2h} \approx 12h^2 E_2(x,t).$$

Doing the analogous computation for the denominator part of Eq.(4.40), we get

$$\phi^{2h} - \phi^{h} \approx 3h^2 E_2(x,t).$$

Thus, putting these results into Eq.(4.40) we get $Q = 4$, as we said.

Although the computation above was quite simple to do, in practice we have to be careful in this calculation. The quantities ϕ^{4h} and ϕ^{2h} as well as ϕ^{2h} and ϕ^{h} are defined on different lattices, and thus they are vectors of a different dimension. Then, we have to choose the lattices such that the vertices present in the lattice of spacing $4h$ are a subset of the vertices present in the lattice of spacing $2h$ and $2h$ is a subset of h. In Fig.(4.3) we can see an example of how we have to proceed with this computation in practice.

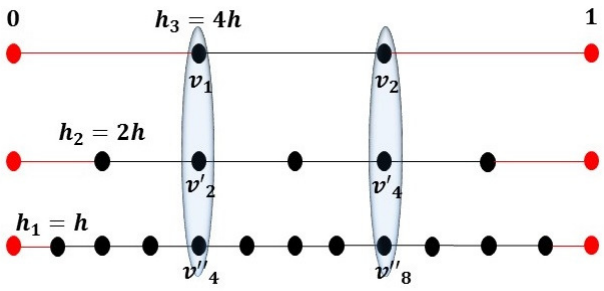


Figure 4.3: This figure illustrate how we have to deal with these vectors that belongs to a different dimension. From up to down we represented the lattice space where the vectors ϕ^{4h}, ϕ^{2h} and ϕ^{h} belongs, respectively. We called by h_3, h_2 and h_1 the three mesh points for each lattice and we showed how these points are related after we calling $h_1 = h$. We marked the points of the one-dimensional lattice that should be compared during the computation of Q. In the end, both are in the same subspace. The red points represent the boundary points for the standing wave simulation.

From our numerical exploration fig.(4.4) we could see that our method is convergent and agrees with the expected value for Q when $h \to 0$. In this test we choose $h = 1/800$ and $\Delta t = 10^{-7}$ and we run our simulation for 10^4 time steps. In (4.4) we can see that Q converges to the predicted value.

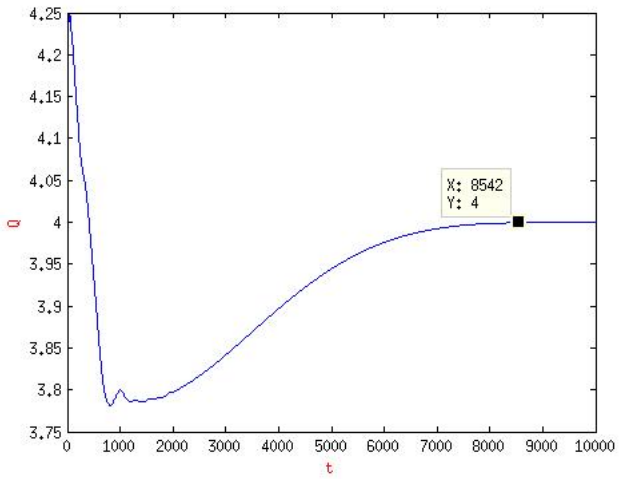


Figure 4.4: Q factor analyses for the standing wave simulation.

4.2.2 The incidence matrix

We reserved this last section to introduce a powerful discretization method that was extremely important in the results for the quantum algorithm for the wave equation, chapter 9.

This discretization method employs some results of spectral graph theory [19] in order to approximate the Laplacian operator. By a simple graph formulation, this technique allows us to establish two important boundary conditions widely explored in numerical methods for differential equation: Dirichlet and Neumann boundary conditions.

Let us return to the one dimensional case, where the Laplacian is just a second derivative. We can now rewrite a second order approximation, (4.7) as follows

$$\partial_x^- \partial_x^+ \phi(x) = -\begin{bmatrix} & & & & & & \\ \ddots & & & & & & \\ & -1 & 2 & -1 & & & \\ & & -1 & 2 & -1 & & \\ & & & -1 & 2 & -1 & \\ & & & & & \ddots & \\ & & & & & & \end{bmatrix} \begin{bmatrix} \vdots \\ \\ \phi(x_{j-1}) \\ \phi(x_j) \\ \phi(x_{j+1}) \\ \\ \vdots \end{bmatrix}. \tag{4.43}$$

We will call this matrix the graph Laplacian $L(G_h)$. This choice of name comes from the fact that it will be constructed from a graph G_h as we will see here. More precisely, we say that the operator $-\frac{1}{h^2}L(G_h)$ approximates ∇^2 in the limit $h \to 0$.

With this new formulation Eq.(4.7) can be written as

$$-\frac{1}{h^2}L(G_h)_j = \frac{\phi(x_{j+1}) - 2\phi(x_j) + \phi(x_{j-1})}{h^2},$$

for the one dimensional case. As the next step we will see how we can apply graph theory in order to construct the Laplacian operator, which is called graph Laplacian. We start with the graph Laplacian definition:

Definition 4.2. Let G be a graph. The Laplacian matrix of G, denoted $L(G)$, is defined by $L(G) = \Delta(G) - A(G)$, where $A(G)$ is the adjacency matrix of G and $\Delta(G)$ is the diagonal matrix whose (i,i) entry is equal to the degree of vertex i of G.

The adjacency matrix is a well known matrix widely used in graph theory and its definition general for a graph with a vertex set $\{v_1 \dots v_n\}$ is quite simple

$$A(G)_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E(G) \\ 0 & \text{otherwise,} \end{cases} \tag{4.44}$$

where $E(G)$ is the edge set of the graph G. Given that, it is clear that the graph Laplacian in (4.7) is achieved from the path graph. For a path graph, Fig.(4.5), the degree of all vertices is 2, we get

a diagonal matrix whose all entries are 2, and an adjacency matrix given by

$$\begin{bmatrix} & & & & & & \\ \ddots & & & & & & \\ & 1 & 0 & 1 & & & \\ & & 1 & 0 & 1 & & \\ & & & 1 & 0 & 1 & \\ & & & & & \ddots & \\ & & & & & & \end{bmatrix}$$

The core property of the graph Laplacian for us, that plays the key role of the discretization

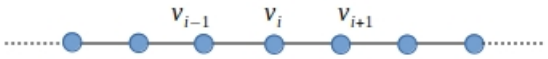


Figure 4.5: Path graph, the graph the yields the graph Laplacian from Eq.(4.43)

method, employed in chapter 9, is the fact that the graph Laplacian is a positive semidefinite matrix. This property implies that this matrix can be decomposed as follows

$$L = BB^{\dagger}. \tag{4.45}$$

The $|V| \times |E|$ matrix B, square root of L, is called as **incidence matrix** and it has many application in graph and spectral graph theory [19]. The general definition of this matrix for a graph where edge j has weight W_j is

$$B_{ij} = \begin{cases} \sqrt{W_j} & \text{if } j \text{ is a self-loop of } i, \\ \sqrt{W_j} & \text{if } j \text{ is an edge with } i \text{ as source,} \\ -\sqrt{W_j} & \text{if } j \text{ is an edge with } i \text{ as sink,} \\ 0 & \text{otherwise.} \end{cases} \tag{4.46}$$

The interesting thing about this decomposition is that we can use the incidence matrix B instead of the matrix L, in order to get the operator $-\frac{1}{h^2}L(G_h)$ that approximates ∇^2 in the limit $h \to 0$. Making use of some properties of spectral graph theory we can from the matrix B easily get the Laplacian operator either under Dirichlet or Neumann boundary conditions. These boundary conditions are widely used either when we have to determine the solution for some differential equation or as an initial condition employed in some numerical method.

Although we will explore only one dimensional examples for unweighted graphs, $W_j = 1$ $\forall j \in E(G)$, we can easily manage this formulation for higher dimensional lattices and for weighted graphs.

4.2.2.1 Dirichlet Boundary condition

Starting with the domain Ω, the Dirichlet case is the one where

$$(\phi)_{\partial\Omega} = 0. \tag{4.47}$$

Let us return to the one dimensional case, with $\Omega = [0,1]$. The Dirichlet condition for this case implies

$$\phi(0) = \phi(1) = 0.$$

Our goal is to establish get the first order Laplacian approximation (a first order Laplacian means a second order approximation method $\mathcal{O}(h^2)$) L for the one-dimensional case under Dirichlet boundary condition. Thus, the challenge here is to see the Laplacian structure in the boundary points. There are two points that should be carefully analyzed, the two extremes points, that we call the leftmost and rightmost points. Let us see this analyze explicitly for the leftmost point. From our approximation (4.7) at point h

$$\frac{d^2\phi(h)}{dt^2} = \frac{\phi(2h) - 2\phi(h) + \phi(0)}{h^2} = \frac{\phi(2h) - 2\phi(h)}{h^2}.$$

This result was established using the fact that $\phi(0) = 0$. Now let us suppose that we fix $h = 1/5$. Thus, doing a similar calculation to the another extreme point, rightmost point, we get

$$-\frac{1}{h^2} L_{\text{Dirichlet}}\phi = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -2 \end{bmatrix}\begin{bmatrix} \phi(h) \\ \phi(2h) \\ \phi(3h) \\ \phi(4h) \end{bmatrix}.$$

From this result above we can see how the Dirichlet boundary conditions change the first and last row of the Laplacian matrix, a result we obtain by analyzing the approximation (4.7) in the extreme points. Furthermore, we can see that choosing $h = 1/5$ we have only to analyze four points in the lattice.

We can now achieve the same result working with the incidence matrix. The $L_{\text{Dirichlet}}$ can be constructed from the graph in fig.(4.6). It is the self loops put at vertices 1 and 4 that provide the Dirichlet condition. From definition (4.46), we can easily build the matrix B,

$$B = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array}\begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} \end{array}.$$

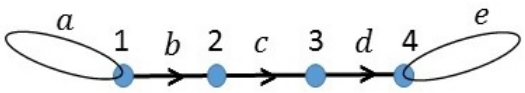


Figure 4.6: Incidence matrix for a line segment, Dirichlet.

Thus, with a straightforward matrix multiplication, we can see that $BB^T = L_{\text{Dirichlet}}$. It is important to notice that the direction chosen in the graph of fig.(4.6) is arbitrary since it will not affect the final structure of L.

4.2.2.2 Neumann Boundary condition

In this case we are restrict to the follow boundary condition

$$\left(\nabla\phi\cdot\hat{n}\right)_{\partial\Omega} = 0, \tag{4.48}$$

where $\hat{n}$ is the unit vector normal to the boundary.

This condition to the one dimensional case $\Omega = [0,1]$ implies

$$\frac{d\phi}{dx} = 0,$$

at the boundaries. Like in the previous case, our goal is figure out how this constraint modifies the second derivatives approximation. In order to see it, let us discretize the second derivatives as

$$\begin{aligned}
\frac{d^2\phi}{dx^2} &= \lim_{h\to 0}\frac{\phi(x+h)-2\phi(x)+\phi(x-h)}{h^2}, \\
&= \lim_{h\to 0}\frac{\left(\frac{\phi(x+h)-\phi(x)}{h}\right)-\left(\frac{\phi(x)-\phi(x-h)}{h}\right)}{h}, \\
&= \lim_{h\to 0}\frac{\frac{d\phi}{dx}(x+h/2)-\frac{d\phi}{dx}(x-h/2)}{h}.
\end{aligned}$$

Now from the Neumann condition at the leftmost point yields

$$\frac{d^2\phi(0)}{dx^2} = \lim_{h\to 0}\frac{\frac{d\phi(h/2)}{dx}-\frac{d\phi(-h/2)}{dx}}{h} = \lim_{h\to 0}\frac{\phi(h)-\phi(0)}{h^2}.$$

Similarly to the rightmost point,

$$\frac{d^2\phi(1)}{dx^2} = \lim_{h\to 0}\frac{\frac{d\phi(1+h/2)}{dx}-\frac{d\phi(1-h/2)}{dx}}{h} = \lim_{h\to 0}\frac{\phi(1)-\phi(1-h)}{h^2},$$

where $\phi(x+h/2)$ vanishes. Thus we can see that $d\phi/dx = 0$ at $x = -a/2$ and $x = 1+a/2$. Then, if we choose $h = 1/4$, that implies $1 = 4h$, we get

$$-\frac{1}{h^2}L_{\text{Neumann}}\phi = \frac{1}{h^2}\begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}\begin{bmatrix} \phi(0) \\ \phi(h) \\ \phi(2h) \\ \phi(3h) \end{bmatrix},$$

where again we have to work with four points of the lattice, but with $h = 1/4$ instead of $h = 1/5$ under Dirichlet boundary condition. In terms of the incidence matrix, we can recover this Laplacian from the graph in Fig.(4.7) with four vertices.

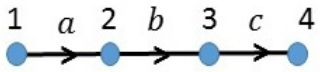


Figure 4.7: Incidence matrix for a line segment, Neumann.

In this last section, we just saw two basic examples for one-dimensional cases. Even with these basic examples, we could see that in order to achieve the correct Laplacian, we need to be careful. It is necessary we do some algebra calculations to get the correct matrix form under the two different boundary conditions. This task can become much more complicated if we have to move to higher dimensions with different boundary conditions. For instance, if we have to simulate a wave propagation in a two-dimensional lattice, surrounded by holes of different shapes and so on. On the other hand we saw that employing some results of spectral graph theory we only have to draw the correct graph, employing self-loops or not, depending on the desired boundary conditions, and subsequently build the incidence matrix B. Thus, doing just a matrix multiplication we get L, the reason that makes this technique so powerful. In addition, this decomposition in terms of incident matrices was extremely important in the quantum algorithm for simulating the wave equation as we will see in chapter 9.

4.2.3 Overview of the FDM complexity

Now we will see a general view about the complexity of the finite difference method to solve differential equations. The complexity here is related with the number of computations that our computer has to do in order to give the approximate value of the function $\phi(\vec{x}, t_n,)$ at time t_n. In order to understand better the complexity of this problem let us see some characteristics of the matrices that we saw during this chapter, in particular, the matrix of Eq.(4.19) and the Laplacian of Eq.(4.43). We can see that in both cases the matrices are quite sparse (the number of nonzeros elements per row), to be more precise the sparsity s of these matrices is $s = 3$. In general when we solve differential equations via numerical methods, either from finite difference method [44]

or finite element method [91] the matrices that we get are sparse. The low sparsity is good for several reasons, for instance we can take advantage of the small numbers of nonzeros to do fewer computations to get the approximated solution and in general, we get well-conditioned systems (see App(B)). There are different algorithms, classical and quantum, that take advantage of the low sparsity to decrease the complexity of the system.

Given the importance of the knowledge of the sparsity, we can estimate the sparsity of the wave equation problem precisely for a D-dimensional lattice from its Laplacian structure

$$L(G) = \Delta(G) - A(G).$$

The time discretization it is not important in this analysis since it only changes the diagonal value of Eq.(4.19). Looking at the expression for L it is clear that the sparsity changes according to the adjacency matrix, since $\Delta(G)$ is a diagonal one. From the definition for $A(G)$, which is a square matrix $|V| \times |V|$, we saw that for the one-dimensional lattice, path graph, $s = 2$. This result is because each vertex is linked with other two. If we move to the two-dimensional lattice the degree of the vertice is four, so each vertex has now four links, then $s = 4$. Thus, it is clear that as we increase the dimension two new links appear. Therefore, we get the following expression for the sparsity of L,

$$s = 2D + 1, \tag{4.49}$$

where the value one in the expression above comes from the matrix $\Delta(G)$.

Now we can look to the condition number of this matrix. As we take small values for h this matrix becomes more and more ill-conditioned, which means in the end that small perturbations in $\phi(x,t_n) \rightarrow \phi(x+\Delta x,t_n)$ will lead to larger errors for $\phi(x,t_{n+1})$). The first thing that we can observe is that if have a line segment of size one and L a $N \times N$ matrix, the lattice spacing is giving by $h = 1/N$, since as we increase N we get more vertices $N = |V|$ and the segment size still the same. From App.(B) we know that the condition number $\kappa(L) = \|L\| \left\|L^{-1}\right\|$ in the spectral norm, the norm commonly used to study κ in linear systems, is given by

$$\kappa(L) = \frac{\lambda_{\max}(L)}{\lambda_{min}(L)},$$

where $\lambda_{\max}(L)$ is the maximum eigenvalue of L and $\lambda_{min}(L)$ is the minimum. When we look at Eq.(4.43) we see that our Laplacian is a triangular matrix,

$$L = \begin{pmatrix} a & b & 0 & \cdots & & 0 \\ b & a & b & 0 & \cdots & 0 \\ \vdots & & & & & \vdots \\ & & & \ddots & & \\ 0 & & \cdots & b & a & b \\ 0 & & \cdots & & b & a \end{pmatrix},$$

with $a = 2$ and $b = -1$. Then, we can use the results of [89] in order to get the expression for the eigenvalues of L,

$$\begin{aligned} \lambda_l &= a + 2b\cos\left(\frac{\pi l}{N+1}\right), \\ &= 2\left(1 - \cos\left(\frac{\pi l}{N+1}\right)\right), \end{aligned} \tag{4.50}$$

for $l = 1, \ldots, N$. In our case, we can see in Eq.(4.50) that the greatest eigenvalue is when $l = N$ and the smallest when $l = 1$, however, from the fact that $\cos(\pi/(N+1)) = -\cos(\pi N/(N+1))$ we can write the following expression for the condition number for L

$$\kappa(L) = \frac{\left(1 + \cos\left(\frac{\pi l}{N+1}\right)\right)}{\left(1 - \cos\left(\frac{\pi}{N+1}\right)\right)}.$$

Now when we take larger N, we can use the approximation $\cos\left(\frac{\pi}{N+1}\right) \simeq 1 - \left(\frac{\pi}{N+1}\right)^2$ to conclude how $\kappa(L)$ scales,

$$\kappa(L) \simeq O\left(N^2\right), \tag{4.51}$$

see App.(A) to understand the big O notation. Although we will not use this condition number result for our complexity analyses here, the knowledge of κ is widely used in many classical and quantum algorithm complexity analyses, including the results that we will show in chapter 9.

After we have explored some characteristics of L we are more than ready to see the complexity of the wave equation simulation via FDM. When we look to Eq.(4.19) we see that the task involved is a matrix-vector multiplication. We are aware that there are efficient algorithms where the complexity involved in matrix multiplication is $O(N^2)$. Since here we are dealing with a multiplication between a sparse matrix and a vector we can take $\Omega(N)$ or $\Omega(1/h)$ as a good lower bound estimation. Now suppose that we are free to change the line segment size, so instead of 1, we have l. Thus, our lower bound change to $\Omega(l/h)$. Now in case we have a D-dimensional lattice it is easy to see that the matrix dimension of L will change to N^D once the number of vertices will increase as follows $|V| \to |V|^D$, therefore we get $\Omega(l/h)^D$ as the lower bound complexity. This analysis is just to get $\phi(\vec{x}, t_{j+1})$ from $\phi(\vec{x}, t_j)$. Thus, if we have to do T computations in the total, we get

$$\Omega\left(T\left(\frac{l}{h}\right)^D\right). \tag{4.52}$$

Although we have only analyzed the complexity of a particular differential equation via finite difference method, we can establish the same bound for other differential equations and be choosing other numerical methods as the finite element method [51]. Let us do general comments about other differential equations. In particular, let us see the ones widely explored by engineers where in general are time-independent differential equations, like the ones studied in static and

elasticity problems [91]. Suppose we want to solve numerically the electrical field in the problem where there is a wave between the parallel plates Fig.(4.8) described by the following differential equation

$$\frac{1}{\mu_r}\frac{d^2}{dx^2}E_y + k_0^2\epsilon_r E_y = f(\vec{x}), \tag{4.53}$$

where $E_y(\vec{x})$ is the electrical field in the horizontal direction between the plates with the boundary conditions $E_y(0) = E_y(x_a) = 0$, ϵ_r and μ_r are the medium's relativity permitivity and the medium's relativity permeability constants respectively, k_0 the free space wave number and finally $f(\vec{x})$ the source function. It is quite simple to see that solve this differential equation via numerical method is equivalent to solve a linear system

$$Ax = b,$$

with $E_y(\vec{x}) = x$ and $b = f(\vec{x})$. This equivalence become obvious when we obesrve that in Eq.(4.53) the operator is the Laplacian, thus,

$$A = \frac{1}{\mu_r}L + k_0^2\epsilon_r I,$$

where I is the identity matrix. Then, solving this problem is the same as solving $x = A^{-1}b$ and the complexity is at least linear in N, when $A \in N \times N$, that leads to the same complexity get for the wave equation problem.

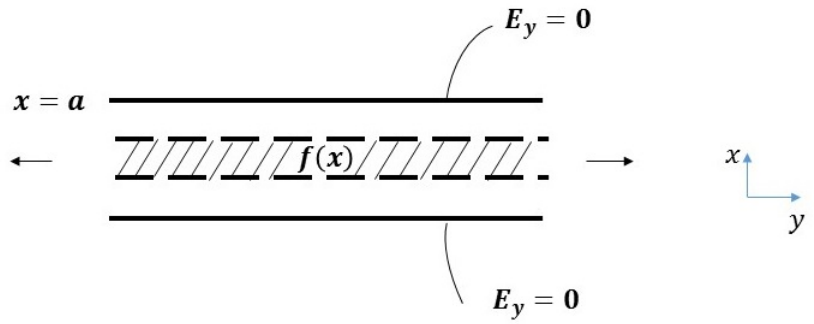


Figure 4.8: Parallel Plate.

To conclude this chapter it is important to notice that in general, using standard techniques any linear dependent equation can be converted to a first-order linear differential equation

$$\dot{x}(t) = Ax(t) + b(t), \tag{4.54}$$

with larger dimension. Then, again we can see that solve this system is at least linear in N. In particular, our wave equation problem translated to the first order differential equation is given

by

$$\frac{d}{dt}\begin{bmatrix}\phi\\ \varphi\end{bmatrix} = \frac{1}{h}\begin{bmatrix}0 & 1\\ -L & 0\end{bmatrix}\begin{bmatrix}\phi\\ \varphi\end{bmatrix}. \tag{4.55}$$

CHAPTER 5

EMERGENT PHENOMENA

This chapter is related with an extremely rich topic in the science, **emergence**, a topic deeply related, as we will see, with another rich and tricky topic, **complex systems**. The general consensus is that emergence is a consequence of collective dynamics of interacting subsystems [49].

We can observe collective dynamics everywhere. For example, we can cite an ant colony, resulted of large amount of individual interacting ants, which is also considered a complex system [24]. The complexity part here comes from the fact that although each individual ant is a simple creature, well understood, their behavior inside the colony is hard to predict. Working together they can build complex structures, like their anthill, thinking collectively, without a leader. Another example of collective dynamics that is also complex is our brain [74]. Here the neurons play the role of the ants, which are relatively simple components. Besides, the interaction between them is limited, in the sense that each neuron can only communicate with a small group of others, compared with the amount existent. Nevertheless, these simple structures, again without a central controller, with this limited interaction, lead us to complex and important large-scale behavior, as the consciousness. Very often, the full understanding of each individual part does not help us to predict the behavior of the whole system.

From these two examples we can see how this topic is rich and important, even nowadays we can not explain the consciousness. Then a lot of effort is put in order to explain the emergence from single parts. Here we can cite Ref.[39], where they used cellular automata to study emergence. But why CA is a good candidate to do this task? Like complex systems in nature, CAs are composed of large numbers of components, cells, with no central controller, each component can only interact with a small group of others, like neurons, obeying its neighborhood scheme. Furthermore CAs can exhibit very complex behavior that is hard to predict from the transition function. As we saw

in (2.1), from a simple rule (rule 90) we got a complex pattern, a fractal, that we could not predict from this simple rule. Then, in [39] they build a tool called coarse-graining to study emergence from cellular automata that played the role of complex system. In their investigations they used this tool to try to predict and understand the emergence as we will briefly comment in the next section.

Now we can give a more physical interpretation to this issue. It seems quite plausible that macroscopic systems are emergent from microscopic. For instance molecules are emergent from atomic interaction. When we move from micro to macro systems, in general, there are fewer degrees of freedom in the last. Let us return to the discussion about gas theory in chapter 2 section 2.3.1 to understand better this point. Although we know that each single gas molecule is fully described by Newton's theory, in order to proceed with some prediction we move to the thermodynamics theory, which demands much fewer resources to work. The idea here is to try to move to another model that can capture the main properties of the microscopic system but that demands fewer degrees of freedom or less resources to work. We also can think the macroscopic behavior as being a consequence of weak detector sensibility, which often happens in physics. For instance, a neutral structure, in general, is established from interactions of positive and negative charges. Often our detector can not access the total reality of the system, it just gives us the information that the system is neutral. The same idea can be done in terms of spin particles. Very often our detectors can not distinguish if there are two neighbors particles with spins pointing to the same direction, and in the end it only processes the information about one. As in [39] our main goal is to develop a tool to study the predictability of systems that can be described in terms of cellular automata. Here, however, guided by physical insights, the ones pointed in the last paragraph. We could conclude, from chapters 2 and 3, that partitioned cellular automata are a class of CAs more relevant in the physical context since we can without difficulty work with conservation laws. Thus, we become interested in doing an equivalent study as was done in [39] for this model. Similarly to the previous results presented by Israeli and Goldenfeld, we propose a coarse graining tool that allows us to connect the microscopic world with the macroscopic one. One interesting aspect about PCA, as we have seen in chap.3, is that we can without difficulty achieve the continuum limit of the model under investigation. Thus, alternatively to the results presented in [39] we explored this possibility, from where we can see stochastic differential equations as emergent from deterministic ones.

This chapter is divided in two sections. In the first one we will briefly present the main ideas and results achieved by Israeli and Goldenfeld in [39], the CG procedure for Wolfram's classification CAs, and subsequently we will move to our new results: Coarse-graining of partitioned cellular automata.

5.1 Coarse Graining of CA

In this section we will be employing the CA definition introduced in section (2.1). We restrict the discussion to the CG of binary ECAs, i.e, $f : \Sigma^3 \to \Sigma$, with $\Sigma = \{0,1\}$.

The main goal of this CG model is that starting from an automaton $A = (a(t), \Sigma_A, f_A)$, see if we can establish a new automaton $B = (b(t), \Sigma_B, f_B)$ after we coarse grain A. We will use the terminology *lower level* to the system A and *upper level* to B. To try to see if exists an emergent automata B, Israeli and Goldenfeld defined a supercell version of A, $A^s = (a^s(t), \Sigma_{A^s}, f_{A^s})$. In A^s each cell represents a composition of s cells from A. The new alphabet Σ_{A^s} simply includes including all possible configurations of s cells in A, thus

$$\Sigma_{A^s} = \underbrace{\Sigma_A \times \ldots \times \Sigma_A}_{s \text{ times}} = \Sigma_A^s.$$

Finally the transition function works following the previous definition, $f_{A^s} : \{\Sigma_{A^s}\}^3 \to \Sigma_{A^s}$, rewrite it in terms of A we have to apply transition function, f_A, s times on all possible initial conditions of length $3s$, $f_{A^s} = f_A^s$. Let us clarify all these concepts with an ECA binary example for $s = 2$.

Let $\Sigma_A = \{0,1\}$ be the binary alphabet for the automaton A. Then the alphabet for the supercell is just $\Sigma_{A^2} = \{00, 01, 10, 11\}$. It is clear that $a^2(t)$ is composed by two cells. Then, in order to update the supercell n we should see all its neighbors $\mathcal{N}_A = \{n-1, n, n+1\}$, but now in terms of supercells,

$$a_n^2(t+1) = f_{A^2}\left[a_{n-1}^2(t), a_n^2(t), a_{n+1}^2(t)\right].$$

Then, likewise for A the state $a_n^2(t)$ updates to $a_n^2(t+1)$ after we read cells at $n-1$, n and $n+1$, but now there are two cells in each location. A problem arises here from the fact that we do not know the rule for f_{A^2}, only for f_A. However, there is a simple procedure to construct f_{A^2} from f_A.

Let us return to the rule for f_A. In chapter 2 we saw that f_A is composed by eight rules, a number that can be easily computed by the follow equation $|\Sigma_A|^{|\mathcal{N}_A|}$, where in this instance $|\Sigma_A| = 2$ and $|\mathcal{N}_A| = 3$. Keeping the same idea, there are $|\Sigma_A|^{s|\mathcal{N}_A|}$ rules for f_{A^s}, that yields 64 for our case, $s = 2$. The task is to build these rules from f_A. Suppose we have a supercell A^2 where the neighborhood is $\{a_{n-1}^2(t), a_n^2(t), a_{n+1}^2(t)\}$ and we want to update the state $a_n^2(t) = (a_l(t), a_r(t))_n$, where we wrote $a_l(t)$ and $a_r(t)$ as the cells state at time t that belongs to the super cell s and the subindex l and r we choose to indicate that they are left and right cells. Thus, f_{A^2} should read all the content inside the supercells, that includes the left and right cells inside $n-1$ and $n+1$. Since we only have access to f_A, one way that we can update the state $a_n^2(t)$ is by applying f_A twice as follows: in the first application we need to apply f_A in the right cell in $n-1$, in both cells in n and in the left cell in $n+1$. In this part we can think this neighborhood as a CA state with 6 cells

$$\{a_l(t)_{n-1}, a_r(t)_{n-1}, a_l(t)_n, a_r(t)_n, a_l(t)_{n+1}, a_r(t)_{n+1}\},$$

where the cells in the boundaries are quiescent states, with means that they remain in the same state after the update. Then, in this first part we applied f_A in the ordinary way updating all

cells in A except the quiescent cells

$$\{a_l(t)_{n-1}, a_r(t+1)_{n-1}, a_l(t+1)_n, a_r(t+1)_n, a_l(t+1)_{n+1}, a_r(t)_{n+1}\}.$$

We can wonder if with just one application it would be enough to update the state $(a_l(t), a_r(t))_n$. However, applying f_A only once the state at the supercell n could not process the state information of the left cell in $n-1$

$$a_l(t+1) = f_A[a_r(t)_{n-1}, a_l(t)_n, a_r(t)_n],$$

and in the right cell in $n+1$

$$a_r(t+1) = f_A[a_l(t)_n, a_r(t)_n, a_l(t)_{n+1}].$$

Thus, in order to catch these informations we need to apply f_A again, since $a_r(t+1)_{n-1}$ contains the information about $a_l(t)_{n-1}$ and $a_l(t+1)_{n+1}$ contains the information about $a_r(t)_{n+1}$. Differently from what we did in the first application we need to apply the transition function only in both cells located in n, $a_n^2(t+1) = (a_l(t+2), a_r(t+2))_n$. We need to stress the fact that we applied f_A only twice in case $s = 2$. For general cases we need to apply, following the same previous idea, the transition function s times, which leads to relation $f_{A^s} = f_A^s$. Therefore, if we want to construct the 64 rules for f_A^2 we need to do the same procedure for all possible initial conditions of length 6, which have the same number of possible rules, 64, whose value is again achieved from the same equation $|\Sigma_A|^{s|\mathcal{N}_A|}$. Although this procedure is quite simple it has a exponential complexity in terms of the supercell size for a fixed neighborhood.

As a next step they defined a projection function $P : \Sigma_A^s \to \Sigma_B$, which is used to map a block with s cells from A into a single cell of B. We can try to understand better the P action with a physical example. Let us get back to a binary case with the following interpretation: if we have the state 1 in cell n it will be equivalent to a particle located in this cell, while state 0 means that the cell is empty. Suppose now we have access to a detector that can not resolve when we have two neighboring particles. This is equivalent to saying that if we have a state 11 our detector gives 1 as the output. We can characterize this process as follows

$$\begin{aligned} P(00) &= 0, \\ P(01) &= 1, \\ P(10) &= 1, \\ P(11) &= 1. \end{aligned} \tag{5.1}$$

Here we presented just one example with a physical meaning. However the number of choices for P increases exponentially in terms of s, $|\Sigma_A|^{2s}$.

Following this simple procedure we can already establish states $b(t)$ from A. However, the core aim is to achieve an emergent dynamics from A, that in the end is to try to get f_B from A. In order to do it, they constructed f_B from f_{A^s} by projecting the inputs and outputs of A^s,

$$f_B\left[P\left(a_{n-1}^s(t)\right), P\left(a_n^s(t)\right), P\left(a_{n+1}^s(t)\right)\right] = P\left(f_{A^s}\left[a_{n-1}^s(t), a_n^s(t), a_{n+1}^s(t)\right]\right). \tag{5.2}$$

Like we did in the previous steps, let us understand better what this expression above says. Starting with left part in the equality (5.2) we see that from A^s we first applied the projection P and then the transition function f_B, since now these states belong to B

$$\left[P\left(a^s_{n-1}(t)\right), P\left(a^s_n(t)\right), P\left(a^s_{n+1}(t)\right)\right] = [b_{n-1}(t), b_n(t), b_{n+1}(t)].$$

We illustrated this idea in Fig.(5.1-a). Differently from this part, the right side of Eq.(5.2) first updates the state $a^2_n(t)$ and just after we have the action of P, that we illustrated in Fig.(5.1-b). Therefore, expression (5.2) demands consistence in both paths at $t+1$, Fig.(5.1-c).

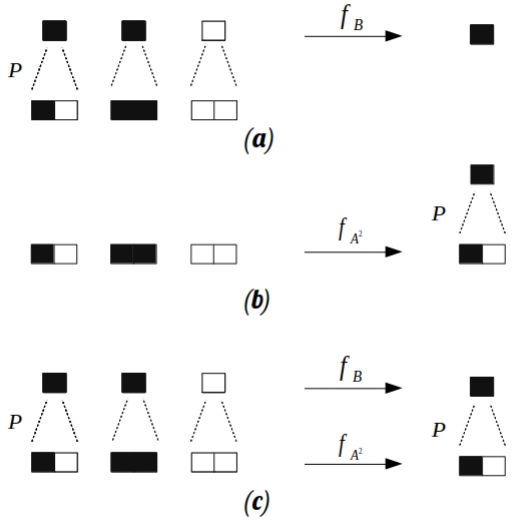


Figure 5.1: These figures illustrate the expression (5.2) for a binary ECA in the lower level with $s = 2$. In this illustration we applied the projector P expressed in Eq.(5.1).

There is an important constraint to this procedure that is given by

$$P(f_{A^s}[x_{n-1}, x_n, x_{n+1}]) = P(f_{A^s}[y_{n-1}, y_n, y_{n+1}]), \tag{5.3}$$

for all x, y such that $P(x_i) = P(y_i)$, where x and y are possible supercell states at the same instant and $i = n-1, n, n+1$. Writing in another way, this constraint means that if we start with two different states in A^s at time t, whose values are equal under the action of P, their states should be the same after we apply the projection function in these states at time $t+1$, obtained by the transition function f_{A^s}.

This CG idea can also be visualized, for general cases, in the scheme below,

$$\begin{array}{ccc} b_t & \xrightarrow{f_B} & b_{t+1} \\ {\scriptstyle P}\uparrow & & \uparrow{\scriptstyle P} \\ a^s_t & \xrightarrow{f^s_A} & a^s_{t+1} \end{array} \tag{5.4}$$

From these two equations equation (5.2) and (5.3) we have constraints enough to establish transition functions for B.

We can now comment briefly about some results established in [39]. Some interesting results were the ones where, from different values for s, they started from binary ECAs and restricted B to be binary ECAs as well. Since they imposed both CAs, in the lower and upper level, to be binary ECAs, which are limited by 256 possible rules, they got different rules being emergent from others.

Even from these simple cases they could learn interesting aspects from these rules and from these CG procedure. For example, there are different structures that we can establish after we evolved some CA rule during a given time. In rule 146, for instance, we can see small triangles appearing everywhere during its evolution. Moreover, this dynamics generates three large triangles, from a given initial state, well distinct from the others. After they coarse grained this rule they got the rule 128. Thus, starting from this rule, where its initial state is achieved from the P action in the initial state of 146, and evolving it during the same time steps we can see the same types of structures emerging. These three triangles also appear in this rule, furthermore in these evolution the small triangles do not appear anymore. Then, from this simple example they could see that the CG eliminated the small scale details of rule 146 and at the same time preserved the most relevant information. This loss of information is expected since P is not a injective function and we are contracting the lattice space.

We summarize the idea of their results in Fig.(5.2).

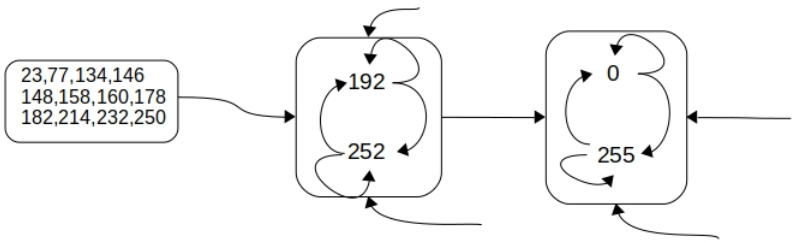


Figure 5.2: Here we illustrated the idea of the results reported in [39] where A and B are binary ECAs. These numbers represent CAs rules and the arrows between them means that they are linked by some coarse-graining or writing in another way, there are projection function and a supercell with size s that allow these rules from A to be connected with different or same rules in B after the coarse graining.

5.2 Coarse Graining of PCA

Like the results of [39], the coarse graining idea for this CA class is to try to build a tool to predict an emergent PCA. As we saw in the last section, here we also have a map that reduces the CA space in the lower level in order to try to get a new CA in the upper one. In the last section we learned that from the transition function f_A we have to apply this function s times in order to update the supercell with size s. Then, putting in other words, all cells inside the same supercell $a_n^s(t)$ that were updated until $t+s$ represents the cell $b_n(t)$ which was updated to $b_n(t+1)$. But here, differently from this previous scheme, with this CA class we can build CG for different time scales. Ultimately it means that we are not restricted to applying the transition function s times in order to update the PCA in the upper level to $t+1$. Furthermore in this result we found CG maps, that play the role of the projectors P in this version, that take deterministic PCA to others deterministic PCA, but also CG maps from deterministic PCA to non-deterministic PCA. These new results are achieved when we discard a constraint analogous to Eq.(5.3) for this current version. Working with less restrictions increase the possibilities for PCAs in the upper level, and allows us to see results with a more physical perspective, as we will see.

The main difference between the work that we will present here and [39] appears because of their structural differences. Although we pointed out some of these differences in the end of chapter 2, we will refresh the ones relevant for us here. As we learned from the previous results, in order to get f_B we need to check all distinct initial conditions in the neighborhood of $a_n^s(t)$, since the transition function f_{A^s} needs to read these states first. But here we need to check all different initial states only inside the supercell. This difference comes from the fact that in each part of $\mathscr{E}$ we only need to read the current state of the subcells that belong to the same tile Fig.(5.3).

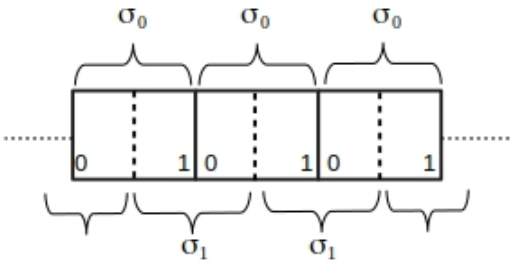


Figure 5.3: Whereas σ_0 is related to the first tiling σ_1 is related to the second one. In this instance, each operator just read the current subcell sates of the two subcells from the same tile.

In these results we only employed subcells with two states $\Sigma_i = \{0,1\}$, then from now on we

will write $(\mathbb{Z}_2)_i$ instead of Σ_i which corresponds to one bit per subcell. Then, $\mathbb{Z}_2^n$ corresponds to our finite set of cell states, which is equivalent to saying that there are n bits per cell. Moreover, we will concentrate our results on one dimensional PCAs, where the excitations do not suffer any kind of interaction. Here no interaction is equivalent to say that the dynamics of each excitation is not influenced by the others. Furthermore, we can easily extend our procedure to higher dimensions with or without interaction, as we will show in the end of this chapter.

Without loss of generality the evolution in this one-dimensional PCA will be restricted to the cases of two tilings in the lower level, since the third tiling from the examples analyzed in chap.(3) does not play an essential role during the PCA evolution. It only gives us a different movement interpretation. Moreover we will always employ the Swap as the second operator, $\sigma_1 = \text{Swap}$. The reason for this restriction, as we will see soon, is that non-trivial dynamics, for the type of interaction between the cells that we choose here, only will happen for this choice of σ_1.

Alternatively to the previous examples seen until the moment, we will explore cases with more than two subcells per cell in the one dimensional example. However, the interactions will remain only between two subcells from different cells, that means the tiles of the second tiling have the following structure; $T_x^{(1)} = \{x_{n-1}, (x+1)_0\}$. That means the interaction between the cells happens only across boundary subcells. Besides, this kind of interaction explains the reason for $\sigma_1 = \text{Swap}$, otherwise we would not allow interaction between the cells.

As we showed previously, for one time step in our CA we need to apply the transition function $\mathscr{E}$. We are aware that $\mathscr{E}$ is composed by the maps σ_0 and σ_1 that we need to apply simultaneously in the first and in the second tiling elements respectively. Then we will often write $\mathscr{E}(\sigma_1, \sigma_0)$ to indicate the transition function format employed.

Before we move to the procedure for CG of PCA, we should add an important characteristic of the operator σ_0 that will be present in this work. In all deterministic cases the internal transition function σ_0 is given by permutation matrix $\pi^{(i)}$, while the extension to the non-deterministic evolution is obtained by employing a convex combination of permutations,

$$\sigma_0 = \sum_{i=1}^{n!} p_i \pi^{(i)},$$

where $p_i \geq 0$, and $\sum_{i=1}^{n!} p_i = 1$.

The restriction to work only with permutation operators naturally appears, since we are interested in reversible PCA in the lower level and to consequence the number of excitations during the evolution. Since these operators are just permutations between the subcells from the same cell, we can easily count the total amount of permutation matrices that we have. The number of matrices is just the number of possible permutations that we have. To be clearer, n subcells gives us $n!$ distinct permutations matrices $\pi^{(i)}$ for $i = \{1, \ldots, n!\}$, which is the reason that we put $n!$ in the convex sum equation above.

5.2.1 Coarse-graning procedure

Like in the previous section, the first thing we should to do in order to get our CG is to construct a supercell. We begin with a PCA state, Φ_t, with N cells, each with n subcells,

$$\Phi_t \in \left(\mathbb{Z}_2^n \times \ldots \times \mathbb{Z}_2^n\right)_N .$$

As the next step we join s cells, where s is an integer number. Thus we get a PCA state in terms of supercells, Φ_t^s,

$$\Phi_t^s \in \left(\mathbb{Z}_2^{sn} \times \ldots \times \mathbb{Z}_2^{sn}\right)_{N/s}, \tag{5.5}$$

where from this construction we get N/s supercells. We need to stress the fact that we always take a choice for N such that $N/s \in \mathbb{N}$. Once we get this N/s supercells we have to construct a CG map as follows:

$$\Lambda_{CG} : \mathbb{Z}_2^{sn} \to \mathbb{Z}_2^{n'}, \tag{5.6}$$

where although we have the possibility of choosing whenever number of subcells n' in the upper level as long as $n' < sn$, we will be restricted to the case where $n' = n$. Afterward, we apply this map in all supercells to achieve a possible CA candidate with N/s cells and with n subcells,

$$\Lambda_{CG}^{N/s} \Phi_t^s = \tilde{\Phi}_T, \tag{5.7}$$

where

$$\Lambda_{CG}^{N/s} = \underbrace{\Lambda_{CG} \times \ldots \times \Lambda_{CG}}_{N/s \text{ times}},$$

and $\tilde{\Phi}_T$ is a PCA state in the upper level. However we do not know yet the transition function, $\tilde{\mathscr{E}}$, for $\tilde{\Phi}_T$. Moreover, like in [39] our interest is to construct $\tilde{\mathscr{E}}$ from the transition function in the lower level.

With this goal in mind, we propose an analogous procedure to [39] for the PCA. Thus, starting with Eq.(5.7) our next step is to apply the transition function in the lower level h times.

$$\mathscr{E}^h \Phi_t^s = \Phi_{t+h}^s, \quad \text{where } h \leq s, \tag{5.8}$$

where here alternatively to the previous result of Israeli and Goldenfeld who always take $h = s$ we relaxed this constraint for the PCA, which are the cases that we called **temporal and spatial coarse-graining**. Subsequently we apply the CG map to get a PCA state in the upper level at time $T+1$,

$$\Lambda_{CG}^{N/s} \Phi_{t+h}^s = \tilde{\Phi}_{T+1}. \tag{5.9}$$

Then, we say that a PCA in the upper level is emergent from the lower one as long as there exists a PCA transition function $\tilde{\mathscr{E}}$, satisfying the PCA definition transition function presented in chapter 2 and thus composed by local operators, that connects these two PCA states,

$$\tilde{\mathscr{E}} : \tilde{\Phi}_T \to \tilde{\Phi}_{T+1}. \tag{5.10}$$

with the following restriction

$$\Lambda_{CG}^{N/s}\left(\mathscr{E}^h\Phi_t^s\right)=\Lambda_{CG}^{N/s}\left(\mathscr{E}^h\Theta_t^s\right), \tag{5.11}$$

for all Φ_t^s and Θ_t^s that are distinct PCA states at time t, such that $\Lambda_{CG}^{N/s}\left(\Phi_t^s\right)=\Lambda_{CG}^{N/s}\left(\Theta_t^s\right)$. Then we can see that Eqs.(5.9) and (5.10) play the role of Eq.(5.2) and Eq.(5.11) plays the role of Eq.(5.3).

Until the moment we have assumed that our CG procedure should be done in the PCA state that includes all supercells Eq.(5.5). However, from the PCA space homogeneity and from its time and space translation invariance, the procedure can be done just analyzing these states inside some neighborhood, $\Phi_t^s\in\mathscr{N}^s$, likewise the CG procedure to CA.

Rather than the previous CG work [39], in this version we have different possibilities for time scale, thus instead of writing t in the upper level we wrote T. We do not allow $h>s$, since in these cases there will be time enough for the information to cross the neighborhood scheme in the upper level, once we only analyzed the cases where we move s cells to one. We can understand it better with a simple example. Imagine that we choose $s=2$ with the following neighbor scheme $\mathscr{N}^s=\{n-1,n,n+1\}$. Each supercell will be reduced to one cell in the upper level, where its neighborhood is the same. As we discussed later our procedure is to build $\tilde{\mathscr{E}}$ from $\mathscr{E}^h$. Now, in case we have $h>2$ the excitation can leave the supercells $n\pm1$. In this case our procedure will fail, since the transition function in the upper level only interact inside $\mathscr{N}=\{n-1,n,n+1\}$. Then, if we allow $h>s$ we can have an emergent structure with non-local operators, out of the PCA definition.

As before, we can summarize our general procedure in the scheme below.

$$\begin{array}{ccc}\tilde{\Phi}_T & \xrightarrow{\tilde{\mathscr{E}}} & \tilde{\Phi}_{T+1}\\ {\scriptstyle\Lambda_{CG}^{N/s}}\uparrow & & \uparrow{\scriptstyle\Lambda_{CG}^{N/s}}\\ \Phi_t^s & \xrightarrow{\mathscr{E}^h} & \Phi_{t+h}^s\end{array} \tag{5.12}$$

where $N/s=|\mathscr{N}^s|$.

At this moment it is important we show some characteristics of the Λ_{CG} employed in this model. Understanding better this map we can also understand better about the physical process. From Eq.(5.6) and from the fact that we worked with the same number of subcells in both levels, $n'=n$ we have that $\Lambda_{CG}\in\mathbb{Z}^{n\times sn}$. What we are saying is that Λ_{CG} belongs to the space of $n\times sn$ matrices, where their elements are either zero or one. From here we can already see that this map is not injective, thus there are different states in the lower level that give to us the same state on the upper level. Physically speaking, there are different microscopic states that corresponds the same macroscopic state. Moreover, there is another important characteristic of this map which is a consequence of the physical interpretation that we are using in our investigations. Since we are interpreting the value one in the subcells as the existence of one particle, or excitation, and zero as an empty cell, and as the number of particles is preserved during the evolution, we only

allow one value different of zero in each column of $\Lambda_{CG}(n,s)$. Otherwise, we will get maps that increase the number of particles after we coarse grain, which also can lead to dynamics in the upper level that does not conserve the number of particles. From here we can already see the number of possible CG maps N_{CG} given the supercell size s and the number of subcells, n. From the fact that we only can have a single one value in each column of Λ_{CG}, there are $n+1$ possible entries for each column. We added one since there are also cases only with zeros. Thus, as we have sn columns we get $(n+1)^{ns}$ possibilities. However, we will exclude the trivial map, which is the case of a map with only zeros, then,

$$N_{CG}(n,s) = (n+1)^{ns} - 1, \tag{5.13}$$

where like the previous results we can see that the map also increase exponentially in terms of s.

In this work, we will only report the results for the CG maps that take two and three cells, $s = 2,3$, to one. In addition, we will explain how to get the CG for a two-dimensional lattice where the particles can collide. Then, this procedure gives the prescription to apply our method to the HPP, chap.(2). The extension for more dimensions as well as for different values for s can be done naturally.

The last point that we should notice when we attempt to get the CG in the deterministic cases is the number of possible connections between the lower and upper level. We are aware that there are $n!$ permutation matrices for n subcells. Moreover the PCAs will be kept with the same structure, in the lower and upper level (the same neighborhood scheme and the same number of subcells). Thus, from each initial dynamics $\mathscr{E}(\text{Swap},\pi^{(i)})$ there are $n!$ possible deterministic dynamics in the upper level $\tilde{\mathscr{E}}(\text{Swap},\pi^{(j)})$. Then, in the end, we get $(n!)^2$ possible **links** between the lower and the upper level. We adopted the word "link" to say that we have connections between the lower and upper level. In case we have more than one CG map connecting the same rules between these two levels we will say that there is just one link. The number of links, where the biggest value is $(n!)^2$, give us the number of different rules connected between the lower and upper level. This fact will be important in our results analyses, either for quantitative or qualitative understanding.

5.2.2 Deterministic CG results for one-dimensional PCA

5.2.2.1 Spatial coarse-graining

In this first part we will show the results for the case where we apply the transition function only once before we coarse grain the lower level $\mathscr{E}^h$, $h = 1$.

Two cells, s = 2, to one cell: Our starting point is $n = 2$, for a case where we map two cells to one. As we previously mentioned, in this case we only have two different permutation matrices,

$$\pi^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$
$$\pi^{(2)} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Then, working only with $\sigma_1 = \text{Swap}$ as the local interaction operator, we have only two deterministic transition function $\mathscr{E}\left(\text{Swap}, \pi^{(1)}\right)$ and $\mathscr{E}\left(\text{Swap}, \pi^{(2)}\right)$.

Despite the fact that we have four possible connections linking the levels, we got only one connection $\mathscr{E}\left(\text{Swap}, \pi^{(1)}\right)$ to $\tilde{\mathscr{E}}\left(\text{Swap}, \pi^{(1)}\right)$ with the CG map given by

$$\Lambda_{CG} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{5.14}$$

First, let us understand these dynamics better, which are the same in the upper and lower level. Thus, we will confirm that they obey the constraints imposed by our CG procedure and finally, we will show the two possible ways to establish the PCA in the upper level.

The dynamics generated by $\mathscr{E}\left(\text{Swap}, \pi^{(1)}\right)$ is a particle that keeps confined in two cells, going back and forward between them,

$$\cdots, (0,1)_1 (0,0)_2, \cdots \overset{\mathscr{E}\left(\text{Swap}, \pi^{(1)}\right)}{\rightleftarrows} \cdots, (0,0)_1 (1,0)_2, \cdots.$$

In case we have started with the particle in the first (second) subcell in the cell one (two), the particle will go to the cell zero (three), but what is important here is that the dynamics will be the same, wherever the point we start.

Now let us compose our supercell putting the cells i and $i+1$ together. Then applying our CG map (5.14) before the transition function

$$\Lambda_{CG}^3 \left[\{(0,0)_{-1} (0,0)_0 \}, \{(1,0)_1 (0,0)_2 \}, \{(0,0)_3 (0,0)_4 \} \right],$$

we get

$$(0,0)_0, (1,0)_1, (0,0)_2. \tag{5.15}$$

Now we apply the transition function in the same initial state

$$\mathscr{E}\left[\{(0,0)_{-1} (0,0)_0 \}, \{(1,0)_1 (0,0)_2 \}, \{(0,0)_3 (0,0)_4 \} \right]$$
$$= \{(0,0)_{-1} (0,1)_0 \}, \{(0,0)_1 (0,0)_2 \}, \{(0,0)_3 (0,0)_4 \}.$$

and afterwards our CG map,

$$(0,1)_0, (0,0)_1, (0,0)_2. \tag{5.16}$$

Now from the results (5.15) and (5.16) we can easily check that they are connected by $\tilde{\mathscr{E}}\left(\text{Swap}, \pi^{(1)}\right)$ in the upper level. Starting with the other three initial conditions in the supercell the upper level is always connected by $\tilde{\mathscr{E}}\left(\text{Swap}, \pi^{(1)}\right)$. These sum up the scheme written in (5.12) which is true for all times.

Now we can move to the computational task to get these results. There are two alternatives to deal with this problem. We can either fix the transition function in the lower and upper level and then we search for a CG map that connects both levels or we can fix the transition function in the lower level and a CG map and then we look for a transition function in the upper level. Let us show the idea using this last procedure. Beginning with the following state $\phi_t^2 \in \mathbb{Z}_2^4 \times \mathbb{Z}_2^4 \times \mathbb{Z}_2^4$ for three supercells for $n = 2$,

$$\phi_t^2 = \begin{pmatrix} 0_4 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0_4 \end{pmatrix},$$

where the subscript 4 means that we have a vector composed only by four zeros

$$0_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

As the next step we apply the CG map showed in Eq.(5.14), which is a 2×4 matrix in this state, thus

$$\begin{pmatrix} \Lambda_{CG} & & \\ & \Lambda_{CG} & \\ & & \Lambda_{CG} \end{pmatrix} \begin{pmatrix} 0_4 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \tilde{\phi}_T,$$

where $\tilde{\phi}_T$ corresponds the state with only three cells after the coarse-grained ϕ_t. All matrices that act into states in the upper and lower level are diagonal and are composed by block matrices. Now we return to ϕ_t^2 and then we apply the transition function,

$$\begin{pmatrix} 1 & & & & \\ & \text{Swap} & & & \\ & & \text{Swap} & & \\ & & & \text{Swap} & \\ & & & & 1 \end{pmatrix} \begin{pmatrix} \pi^{(1)} & & & & & \\ & \pi^{(1)} & & & & \\ & & \pi^{(1)} & & & \\ & & & \pi^{(1)} & & \\ & & & & \pi^{(1)} & \\ & & & & & \pi^{(1)} \end{pmatrix} \begin{pmatrix} 0_4 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0_4 \\ 0_4 \end{pmatrix} = \phi_{t+1}^2,$$

where the ones in the matrix that contains the Swap operator means that we are not applying any operator in the boundaries, since the action in these points will always be trivial. Subsequently, we apply the CG map in the state ϕ^2_{t+1},

$$\begin{pmatrix} \Lambda_{CG} & & \\ & \Lambda_{CG} & \\ & & \Lambda_{CG} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0_4 \\ 0_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \tilde{\phi}_{T+1}.$$

Now from ϕ_T and ϕ_{T+1} we can start to see if there exists some transition function in the upper level. Since we are looking for a permutation operator in the upper level that connects these two states, we can parameterize the first operator as follows,

$$\pi^{(x)} = \begin{pmatrix} p & q \\ q & p \end{pmatrix},$$

where $x = 1(2)$ if $p = 1(0)$ and $q = 0(1)$. Now we can apply the transition function using, $\pi^{(x)}$ to start to build a linear system to solve,

$$\begin{pmatrix} 1 & & & \\ & \text{Swap} & & \\ & & \text{Swap} & \\ & & & 1 \end{pmatrix} \begin{pmatrix} \pi^{(x)} & & & \\ & \pi^{(x)} & & \\ & & \pi^{(x)} & \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

From this simple case we will get $p = 1$ and $q = 0$. However, it does not guarantee that we have a PCA in the upper level. We have to do this same procedure for all different states inside the supercell. Then, if we get $p = 1$ and $q = 0$ for all different initial conditions for all ϕ^s_t and Θ^s_t that are distinct PCA states at time t, such that $\Lambda^{N/s}_{CG}\left(\phi^s_t\right) = \Lambda^{N/s}_{CG}\left(\Theta^s_t\right)$ and say that we have a PCA in the upper level with $\tilde{\mathscr{E}}(\text{Swap}, \pi^{(1)})$ as its transition function.

One of the main characteristics of our results is that we rarely established maps that preserve the number of particles. For instance, applying this map either in $\{(0,0)_1, (1,0)_2\}$ or $\{(0,1)_1, (0,0)_2\}$ we get $(0,0)$. In fact these results are expected for us. Once we are reducing our space we expect some information loss, where in this case the information is the number of particles. This also can be understood from a physical point of view.

Now we move to the case where $n = 3$. In this case there are six different permutation matrices that we will list here,

(5.17)
$$\pi^{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}; \qquad \pi^{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix};$$
$$\pi^{(3)} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}; \qquad \pi^{(4)} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix};$$
$$\pi^{(5)} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}; \qquad \pi^{(6)} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

In this scenario we achieved twelve connections from all the thirty six possible ones, one third of all possible connections. These links were made by only eight distinct CG maps,

$$\Lambda_{CG_1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}; \qquad \Lambda_{CG_2} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix};$$
$$\Lambda_{CG_3} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \Lambda_{CG_4} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix};$$
$$\Lambda_{CG_5} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \Lambda_{CG_6} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix};$$
$$\Lambda_{CG_7} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}; \qquad \Lambda_{CG_8} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

These results are summarized in figure (5.4).

Moving forward we got results for $n = 4$. In this case from $(4!)^2 = 576$ permutation matrices we established 218 connections which represent almost 38 percent of all possible links. Our last result for this case, $s = 2$ and $h = 1$, is with $n = 5$. Once again we could see more relative links showing up. Now from 14400 we got 6628, more than 46 percent of cases. Then we could see, until here, that the relative number of links increases with the number of subcells.

Three cells, s = 3, to one cell: We again started with $n = 2$. Our results for this case, where now we have CG maps from three to one cell, in terms of their dynamics, are exactly the same that we got before, from two to one cell. We established one link between the same transition functions $\mathscr{E}\left(\text{Swap}, \pi^{(1)}\right)$ to $\tilde{\mathscr{E}}\left(\text{Swap}, \pi^{(1)}\right)$, where the CG map is only an extension from the previous case

(5.18)
$$\Lambda_{CG} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

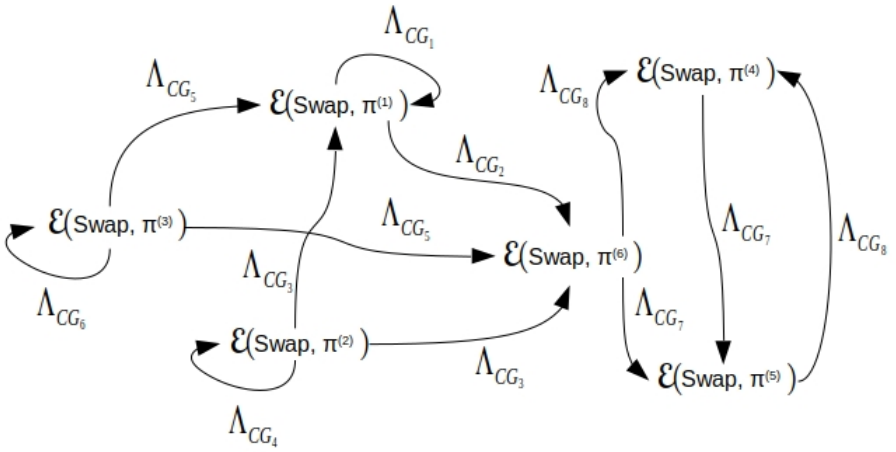


Figure 5.4: CG results with $s = 2$ and $n = 3$. These arrows are connecting the CA dynamics after the CG maps.

Now with $n = 3$ we got eight links from the thirty six possibilities. These connections are given by seven CG maps

$$\Lambda_{CG_1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}; \qquad \Lambda_{CG_2} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix};$$

$$\Lambda_{CG_3} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \Lambda_{CG_4} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix};$$

$$\Lambda_{CG_5} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \Lambda_{CG_6} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix};$$

$$\Lambda_{CG_7} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

The results are illustrated in figure (5.5). With four subcells we got 202 links and with five subcells 6286, which represent approximately 35 percent and 44 percent respectively of all possible connections.

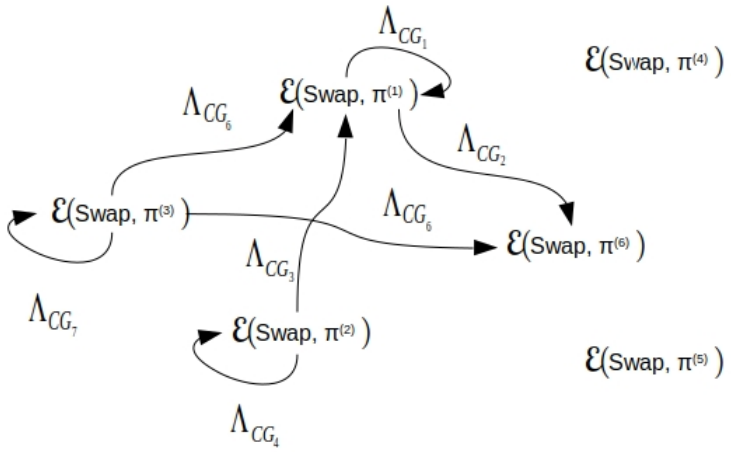


Figure 5.5: CG results with $s = 3$ and $n = 3$.

5.2.2.2 Spatial and temporal coarse-graining results

Two cells, s = 2, to one cell: Now we will allow change the time scale by working with different values of h in the lower level Eq.(5.8). As we are in the case of $s = 2$ the only value allowed for h in this case is $h = 2$.

Starting with $n = 2$ we did not get any link between the lower and upper level. Working with $n = 3$ we got eight links from six different maps

$$\Lambda_{CG_1} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \Lambda_{CG_2} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix};$$

$$\Lambda_{CG_3} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \Lambda_{CG_4} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix};$$

$$\Lambda_{CG_5} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}; \qquad \Lambda_{CG_6} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

These are fewer connections and maps than we established with $h = 1$. These results are summed up in Fig.(5.6). Moving to $n = 4$ we got 172 results which represent almost 30 percent of the possible connections and finally with $n = 5$ we established 5912, that is approximately 46 percent of the total possible links. Although these relative numbers are smaller than the case we achieved

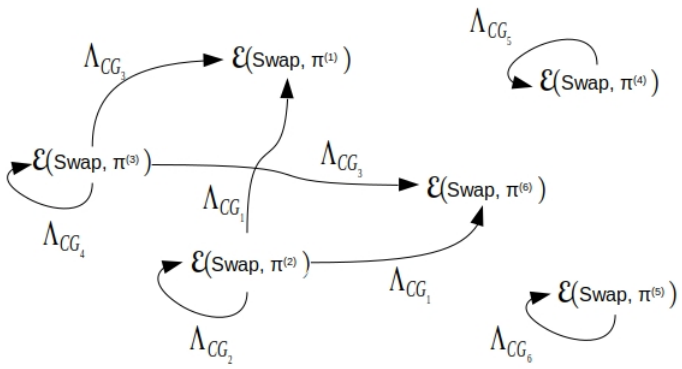


Figure 5.6: CG results with $s = 2$ and $n = 3$.

with $h = 1$, it is important to notice that these results are still increasing with the number of subcells.

Three cells, s = 3, to one cell: Now we can work with the two values for h, $h = 2$ and $h = 3$, since $s = 3$. Let us start showing the results for $h = 2$. With $n = 2$ we did not get any result, but with $n = 3$ the results are summarized in Fig.(5.7). In this case we only got four different maps

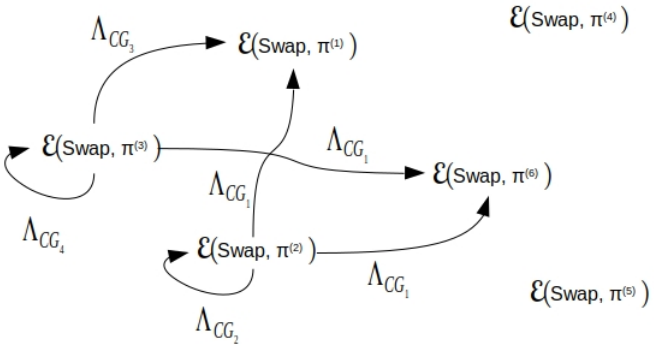


Figure 5.7: CG results for $s = 3$ with $h = 2$ and $n = 3$.

$$\Lambda_{CG_1} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \Lambda_{CG_2} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix};$$

$$\Lambda_{CG_3} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \Lambda_{CG_4} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix};$$

and again comparing with $h = 1$ we got fewer links. With $n = 4$ we have 152 connections and finally $n = 5$ give to us 5516, whose values represents approximately 26 percent and 38 percent respectively from the total possible links.

Now we move to the last value, $h = 3$. First with $n = 2$ we just recover the result for $h = 1$. It means that after we apply three times the transition function is equivalent to $h = 1$, $\mathscr{E}^1 = \mathscr{E}^3$. With $n = 3$ although we found only few a links, they are different from the case we got applying the transition function just once. Different from the previous cases, here we got four CG maps for four links, as we can see in Fig.(5.8). Going to the case of four subcells we established 174

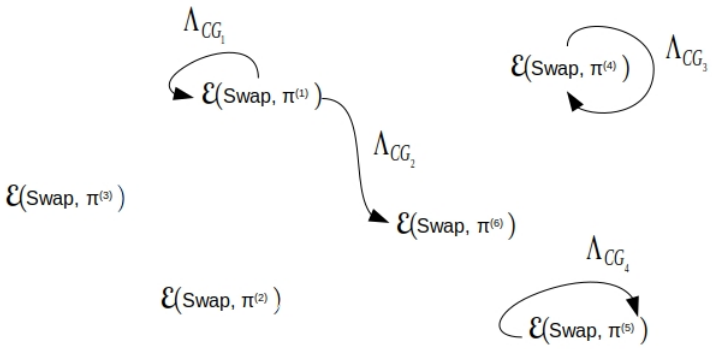


Figure 5.8: CG results for $s = 3$ with $s = 2$ and $n = 3$.

connections, approximately 38 percent and with five subcells, 6108, more than 42 percent. While the number of links for $n = 3$ is smaller than what we got for $h = 2$, with four and five they become larger.

5.2.2.3 Overview of deterministic results

From all these results that we got from deterministic dynamics in the lower level to deterministic in the upper one we could see that we did not establish all possible links for each value n explored. However, we could see these relative links increasing, as we can see in Fig.(5.9) for $s = 2$ and in

Fig.(5.10) for $s = 3$. Should we expected all links showing up for some number of subcells? In fact

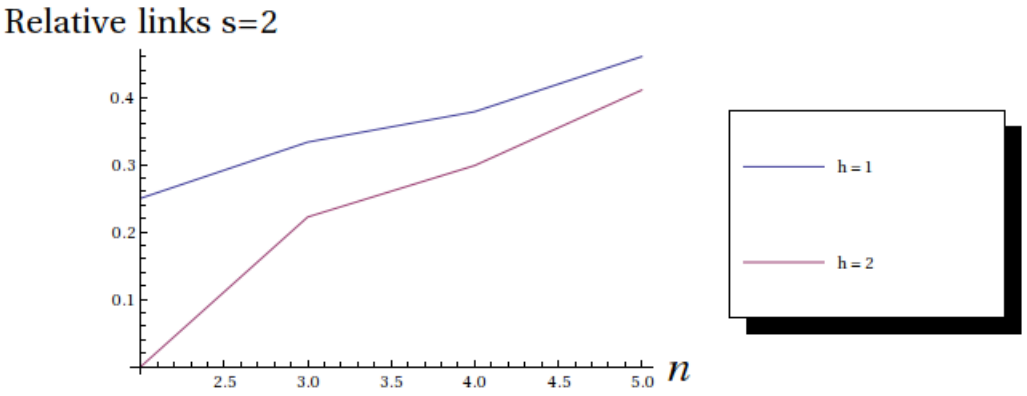


Figure 5.9: Relative links from the case that we are going from two cells to one cell, afterwards we apply our CG map.

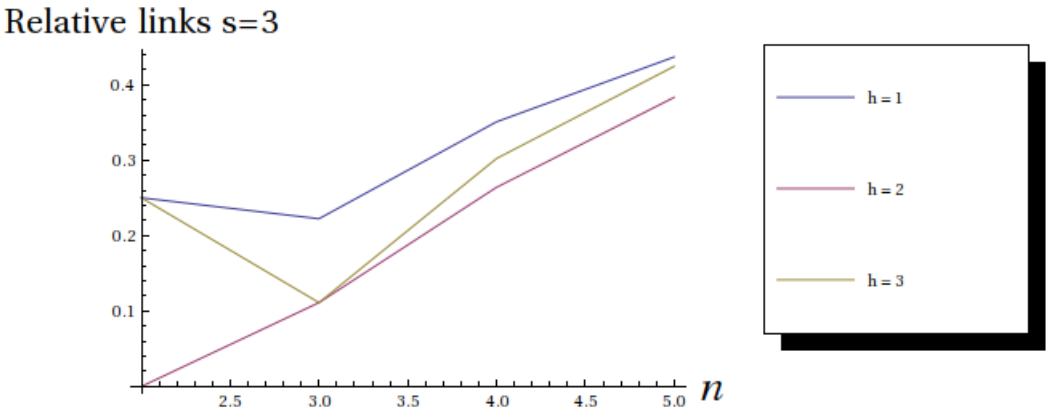


Figure 5.10: Relative links from the case that we are going from three to one cell, afterwards we apply our CG map.

this seems quite probable. Let us understand why.

From Eq.(5.13) we know the total number of possible maps given n and s. Besides we are aware that there are $(n!)^2$ possibly links. Thus, from these quantities we can see that the number of maps increase faster than the total of links,

$$\lim_{n\to\infty} \frac{(n!)^2}{(n+1)^{ns}-1} = 0,$$

for a given s. From this brief analysis, we can expect that at some point all links will appear. Since as we increase the number of subcells the number of maps increase faster than the possible dynamics. In the end, it means that we are contracting more and more the space. Ultimately,

it implies that we will have more microscopic dynamics that can not be distinguished after the coarse-graining.

Another observation that we can do at this point is that the links also depends on the values of h employed. The reason is that there are values for h, the number of times that we have to apply the transition function in the lower level before we coarse grain the state, that might lead the particle to stay inside the same initial supercell. In these cases, we will get trivial dynamics in the upper level, which means particles that do not interact with the neighbors Since we are not including these possibilities we will get fewer links for these cases.

5.2.3 Non-deterministic CG results for one-dimensional PCA

Now we will search for convex combination of permutations in the upper level starting from some fixed CA dynamics in the lower level. This is possible as long as we relax some of the constraints imposed to achieve the transition function $\tilde{\mathscr{E}}$, in this case, we no longer impose Eq.(5.11). Relaxing this constraint implies to us that if we have two or more initial states in the lower level going to the same state after the coarse graining (i.e. different states in the lower level represents the same state in the upper one), these states in the lower level, afterwards we apply the transition function, might go to different states in the upper level. Thus, at the end of this process, we can have different transition functions in upper level.

By comparison with what we saw for the deterministic cases, where there is not more than one CG map doing the same link, the possibility of the non-deterministic evolution in the upper level allows us to get different CG maps linking the same dynamics.

The possibility to get different maps from the same link gave us much more maps in comparison with the previous results. Moreover, we will only show the results for $s = 2$ with three subcells and since the permutation matrices are the same that we worked in the deterministic cases we will keep the same notation presented in (5.17).

In this section, we will present our results in a different way, since the method employed was different. We started saying which dynamics we started in the lower level and thus we present all possible non-deterministic dynamics in the upper level. In this part, we search these dynamics fixing the CG map and looking for emergent dynamics.

5.2.3.1 Spatial coarse-graining results

Likewise deterministic results, spatial CG means $h = 1$.

- $\pi^{(\mathbf{1})}$: working with $\pi^{(1)}$ in the lower level we found seven maps, for example

$$\Lambda_{CG} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

that give in the upper level the follow convex combination for the operator related with the first tiling,

$$\sigma_0 = p_1\pi^{(1)} + p_6\pi^{(6)}, \tag{5.19}$$

where $p_1, p_6 \geq 0$ and $p_1 + p_6 = 1$. We also got a convex combination for the operator related with the second tiling σ_1,

$$\sigma_1 = q_1\mathbb{1}_2 + q_2\text{swap}, \tag{5.20}$$

where $q_1, q_2 \geq 0$ and $q_1 + q_2 = 1$ and $\mathbb{1}_2$ is the identity operator. Putting this last part in words it says that with probability q_1 the particle will stay in the same cell and with probability q_2 the particle will leave the cell.

- $\pi^{(2)}$: for this case we only got one non-deterministic case in the upper level,

$$\Lambda_{CG} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{5.21}$$

which leads to the same evolution that we got in Eq.(5.19), except that now $\pi^{(1)}$ remains the same.

- $\pi^{(3)}$: like the result for $\pi^{(2)}$ we got only one map,

$$\Lambda_{CG} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{5.22}$$

for the same dynamics in the upper level, (5.19) for the first operator and swap for the second.

- $\pi^{(4)}$: alternatively for the previous cases the upper level, whose the CG map is given by

$$\Lambda_{CG} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix},$$

has a deterministic operator for σ_0 that is $\pi^{(5)}$, but the second operator has the follow format,

$$\sigma_1 = \frac{1}{2}\mathbb{1}_2 + \frac{1}{2}\text{swap}.$$

Thus we have 1/2 as the probability of stay or leave the cell.

- $\pi^{(5)}$: once again we got a deterministic evolution for the first operator in the upper level, but now the permutation is $\pi^{(4)}$. Coincidentally with the result achieved for $\pi^{(4)}$ both the CG map and the σ_1 operator are the same. In fact, doing a careful analysis of these permutation operators, $\pi^{(4)}$ and $\pi^{(5)}$, we can see that they are related by a transposition transformation, $(\pi^{(4)})^T = \pi^{(5)}$, then this result, in fact, is expected.

- $\pi^{(6)}$: finally for our last permutation operator we established three different maps, for instance

$$\Lambda_{CG} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

  connecting the same dynamics got for $\pi^{(1)}$, (5.19) and (5.20).

5.2.3.2 Spatial and temporal coarse-graining results

Now we present the results established for $h = 2$.

- $\pi^{(1)}$: again we established the same dynamics for $\pi^{(1)}$ with $h = 1$, (5.19) and (5.20). However, now there are 63 CG maps doing the same, for instance

$$\Lambda_{CG} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

- $\pi^{(2)}$: we keep with the same dynamics achieved in the previous result, but there is only one map (5.22).

- $\pi^{(3)}$: like for the two previous cases, we achieved the same non-deterministic transition function in the upper level with (5.21) as our CG map.

- $\pi^{(4)}$: we could not established any dynamics in the upper level from this deterministic PCA.

- $\pi^{(5)}$: as we have already commented that the dynamics generated by $\pi^{(4)}$ and $\pi^{(5)}$ are quite similar, we replicated the last result, non dynamics were established in the upper level.

- $\pi^{(6)}$: now we established three CG maps for only one dynamics, the same one that we have seen for $h = 1$.

These last results either for $h = 1$ or $h = 2$ here a strong physical meaning, as we will see now. The dynamics generated by Eq.(5.19) with a swap operator related with the operator for the second tiling gives the Brownian motion, as we saw in chapter 3, Eq.(3.3). Thus, this discrete

equation of motion, as we saw in chapter 3, is the one that gives a stochastic partial differential equation in the continuous limit,

$$\partial_t \rho + D\partial_x^2 \rho = 0.$$

We could see from these last results that this dynamics showed up repeatedly. Now let us to try to understand the main reason for that.

In the lower level, we had particles with a deterministic behavior. However as we have seen the dynamics of these particles are strongly related to their initial condition. For example, in case we have $\mathscr{E}\left(\text{Swap}, \pi^{(6)}\right)$ we will see particles propagates either to the right or to the left, depending on its initial condition. In our procedure we have to include all initial conditions and after we take the average of them. Then, despite the fact that these particles have deterministic behavior, in the upper level our detector do not have access to its information and in the end the process looks like random to us.

Moreover the differential equation that we got afterwards we took the continuum limit is described in relation to density of particles. Then, we can also think that we had a bunch of particles in the lower level with a really well defined behavior. However, after we put all together our sensors could not describe the dynamics of each particle individually, where their behavior now look like random for us.

5.2.4 CG in $\mathbb{Z}^d$ for multiparticles with or without interaction

In all examples and results presented until now, we only have considered the dynamics of only one excitation. A natural question is if our prescription works for more than one particle. There are two cases that we need to consider, the case where the particles can interact and the case they can not.

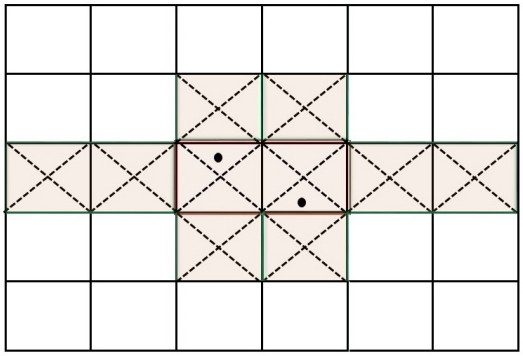
Let us start with case where the particles do not feel any interaction. In this scenario our prescription already works. It is because in this case the particles are acting independently of each other. Then it means that we can get our results for only one particle and apply the same maps for multiparticles CA. This can be done as long as we take only the CG maps that have only one element nonzero per row. Let understand better this constraint. We have already explained the reason that we do not accept more than one value different from zero in each column, however, we did not say anything about the rows. For the cases of only one excitation there is not any restriction, however for the cases that we have more than one particle the restriction gave for the columns have to be put for the rows. Otherwise, we can get more than one particle in each subcell, which is not allowed in our model.

In case we have more particles and they can interact we should take a special attention. To understand the general prescription for this case, let us focus in the example for a two dimensional lattice and $n = 4$, where the neighbor scheme is $\{(x, y-1), (x-1, y), (x, y), (x+1, y), (x, y+1)\}$. Imagine that we have only two particles and we want to get a CG for $s = 2$ and $h = 1$. In this example we only have to include two neighbors in all directions, as we can see in Fig.(5.11), since

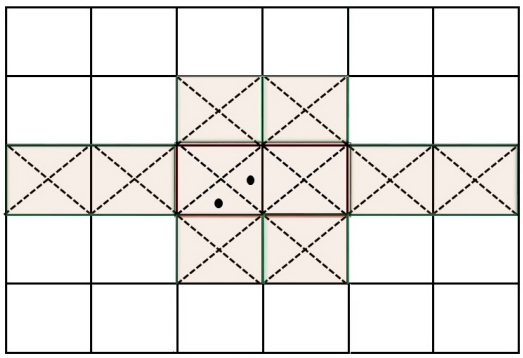
applying the transition function once, $h = 1$, does not allow the particle arrives in any other supercell. In order to get the CG map we need to consider all different initial conditions inside the supercell, which gives

$$\binom{8}{2} = \frac{8!}{(8-2)!2!}, \tag{5.23}$$

different possibilities for the initial condition with two particles. Two distinct cases can be viewed in Fig.(5.11).



($\boldsymbol{a}$)



($\boldsymbol{b}$)

Figure 5.11: CG in a two dimensional lattice with $s = 2$. These figures show two distinct cases for initial condition when we have two particles. As in this example our supercell has size two and we are only working with CG in space we do not have to include more neighbors.

Despite the fact we are in the scenario of two particles, in many situations they can be far a way from each other. Therefore we also need to consider all cases with only one particle in this procedure, adding up more eight cases to this example. After we include all these possibilities for the initial condition we apply the same procedure showed before, for one single excitation. We

do not have to include the cases where we have one particle inside the supercell of center and the other in some neighbor supercell since the particles will not interact in one single step. Thus, these cases are equivalent to more particles without interaction. Besides, after we presented a new way to deal with many particles, in the end of this section, where we increase the lattice dimension, the reason that we do not have to consider the initial condition where there are excitations inside the neighbors supercells will become even more clear.

If we include more particles we need to include more initial conditions, however there is a limit for these inclusions, whose value is given by the supercell size and the number of subcells, sn. The argument that we do not have to include the cases where there are particles in the neighbors is the same said for case where the supercell is completely full.

In summary, if we are working with p particles in total, where $p \leq ns$, and we want to get a CG map where the supercell size is s and each cell has n subcells, the number of initial conditions that we should consider is

$$\sum_{i=1}^{p} \binom{ns}{i}, \tag{5.24}$$

and if we are in the case that $p > ns$ this sum only have to run until $p = ns$.

Rather than thinking in terms of many particles and therefore concern about the interactions between them, we can translate it to one particle scenario which is completely equivalent to multiparticles if we increase the dimension of the subcells. Let us come back to this previous example where we had two particles in a two dimensional lattice with four subcells. The dimension related with the subcells is four and it can be spanned by the canonical bases $\{e_0, \ldots, e_3\}$ where

$$\begin{aligned} e_0 &= (1,0,0,0) \\ &\vdots \\ e_3 &= (0,0,0,1). \end{aligned}$$

The interaction between the particles can be implemented using two distinct permutation operators acting in a conditional way. Let us understand it better. From Fig.(5.11) we see that the four subcells were encoded in terms of left, right, top and the bottom part of each cell. We will represent the left and the right part as e_0 and e_3 and the top and the bottom part as e_1 and e_2, respectively. When the particles are moving freely if we have a particle located at $e_{0,(3)}$ it goes to $e_{3,(0)}$ and in case of a particle inside $e_{1,(2)}$ it goes to $e_{2,(1)}$. As we can see the dynamics inside of each cell can be described in terms of a permutation operator, which we will call by π_1. This operator cover all cases that we have one particle and the cases that we have two particles, except two particular cases $e_0 + e_3$ and $e_1 + e_2$. These two last cases are the ones that represents the particles with opposite direction in the same cell, thus we want to see interaction between them, following the HPP collision rule, chapter 2. In these configurations we have the action of π_2 as

the permutation operator, that acts in the following way

$$\begin{aligned}\pi_2(e_0+e_3) &= e_1+e_2,\\ \pi_2(e_1+e_2) &= e_0+e_3.\end{aligned}$$

Although we have our operator σ_0 being described by two permutation operators, π_1 and π_2, they are acting in a conditional way, which means that in the end σ_0 is not a permutation matrix. However, if we increase the internal dimension from four to ten we can do it. The canonical basis our space now is spanned by $\{e_0,\ldots,e_9\}$. If we work only with this first four basis vector, without any linear combination, is equivalent to the previous case, four subcells with one particle, but with a larger space

$$\begin{aligned}e_0 &= (1,0,0,0,0,0,0,0,0,0)\\ &\vdots\\ e_3 &= (0,0,0,1,0,0,0,0,0,0).\end{aligned}$$

Now we can take advantage of this larger space to encode the states where we have two particles in the four-dimensional space. The idea is to create a single excitation in this new space related with two particles. We can encode it as follows,

$$\begin{aligned}(1,1,0,0) &\rightarrow e_4,\\ (1,0,1,0) &\rightarrow e_5,\\ (1,0,0,1) &\rightarrow e_6,\\ (0,1,1,0) &\rightarrow e_7,\\ (0,1,0,1) &\rightarrow e_8,\\ (0,0,1,1) &\rightarrow e_9.\end{aligned}$$

Now we can construct σ_0 with only one permutation operator that includes the two interaction cases, which is given by $\sigma_0 : e_6 \rightleftarrows e_7$. Thus, working with $n = 10$ in each cell using the correct permutation operator as the operator for the elements in the first tiling, we can achieve our CG map working with only one particle as long as we compute all initial configurations. This number is exactly the same that we have to apply working with $n = 4$ in the scenario of two particles. Therefore, this new point of view suggests to us that both formalisms are equivalent. Furthermore, in this case, it is evident that we do not have to analyze the initial states that represent the excitation outside the center supercell once we have only this excitation in this larger space. Thus, this point of view agrees with our claim about do not include particles in the neighborhood of the supercell.

5.2.5 Final considerations

Similarly to [39] in this work we studied emergent dynamics but in a different scenario of CA. Differently from the previous results, work with PCA allowed us to get CG maps for different time

scales. One advantage in choosing this class of CA is its strong connection with physical processes, for instance the Navier Stokes [32] and Brownian motion chap.(3) can be easily simulated applying this computation model. Moreover we established two distinct results, links connecting deterministic CA to deterministic CA and deterministic CA to non-deterministic CA.

Despite the fact that our results suggest to us that all links between the lower and the upper level will be achieved in the deterministic results, for some high number of subcells, we could see how difficult it is to get these emergent phenomena, since the total number of CG maps increases fast in terms of subcells. While we could not observe different CG maps doing the same connections in the deterministic results it happened very often in the non-deterministic cases, that can pinpoint to us why the non-deterministic process in the macroscopic world naturally emerge from well determined individual particle actions, in agreement with the statistical mechanics, aspect discussed in section 2.3.1.

Taking advantage of this class of automata we saw in the last section that our method can be easily translated to multiparticles case, where we only have to be more careful when we have interaction between them.

Going beyond the classical CA, our CG prescription could be useful to its quantum counterpart which is the core motivation for the results presented here.

Part II

Quantum Models of Computation

CHAPTER 6

QUANTUM CELLULAR AUTOMATA

This chapter initiates our study of quantum models of computation. In part I of this thesis we gave emphasis to cellular automata, showing how powerful this tool is to simulate complex process in physics. In particular, we showed how we can use this model of computation to simulate many particles that can collide, chapter 2, and how the partitioned CA provides a numerical method for differential equations, when we applied it into the random walk problem and Brownian motion, chapter 3. Besides, in chapter 5, we saw that the CA structure, constructed in terms of cells and local operators, is an excellent tool to study emergent processes. These were only a few examples from the vast research activity of the cellular automata theory, [17, 35, 52, 87].

After the brief discussion above and everything that we saw in the previous chapters about CAs, one question naturally emerges to us when we move to quantum models of computation: does the quantum cellular automata (QCA) share the same amount of research activity? Despite the fact that the first model proposed to the QCA was given by [36] in 1988, that we can say that is old since the Feynman's paper was published in 1982 [30], the research activity on QCA is not even closer to its classical counterpart. We can point out two possible reasons why the QCA was not widely employed until today. First, we can say that is because the limitation of our classical computers to deal with quantum particles. We saw that CA is really powerful to deal with many particles, complex systems, problems with many degrees of freedom and so on. Thus, if we try to convert all these problems to their quantum counterparts we might face the limitations that our classical computers have. Moreover, if we try to use the current quantum computers available it will not help us because the number of qubits still limited. Thus, it might not motivate people to try to explore this model. The second possible reason is that the current QCA definitions might be not so clear, avoiding people use this model of computation to their problems. If we look carefully

the majority of publications until today about QCA, we will conclude that there are more results proposing definitions to QCA [36, 48, 57, 80],which in the end were shown to be equivalent [4], than results applying QCA to investigate physics [23].

After all this study we did, after we have seen how CA is useful to address different classes of problems, from biology to physics, we believe that its quantum counterpart deserves a special attention. It is quite possible that if we develop more this model of computation, bringing a new understanding, the QCA will be as useful as its classical version in a near future. We are convinced that with a clear QCA definition and after we see how to use them in practical problems, QCA will be a quantum computational tool widely explored to distinct areas, likewise its classical counterpart. The main proposal of this chapter is to introduce a new QCA definition and apply it to two basic scenarios. This new definition served as the basis for our new result presented in the next chapter, a result that translates the main quantum walk flavors, like the coined model [1], in terms of QCA.

6.1 Previous QCA models

Instead of starting this chapter writing our QCA definition, we will first make a few comments about some previous attempts and definitions of QCA.

Grössing and Zeilinger were the first that have tried to formalize a QCA [36]. Their focus was in the attempt to define an infinite one dimensional QCA. As we expect when we are dealing with closed quantum systems, the update of their automata was given by an unitary operator, which was a band-diagonal operator U. Then, if we have $|\psi_t\rangle$ being the vector state that describes our automata in the time t, the vector state of our automata in the time $t+1$ is $|\psi_{t+1}\rangle$ in such a way that $U|\psi_t\rangle = |\psi_{t+1}\rangle$. To be more precise, their formal definition was

Definition 6.1. [Grössing-Zeilinger QCA]. A Grössing-Zeilinger QCA is a 3-tuple $(L, \mathscr{H}, U)$ consisting of:

1. an infinite one-dimensional lattice $L \subseteq \mathbb{Z}$ representing basis states of $\mathscr{H}$;
2. a Hilbert space $\mathscr{H}$ with basis set $\{|\phi_i\rangle\}$;
3. a band-diagonal unitary operator U;

The band-diagonality of U corresponds to a locality condition. It turns out that there is no QCA with this definition with nearest-neighbor interaction and nontrivial dynamics [36]. Furthermore here the unitarity constraint is relaxed to only approximate unitarity, let us see some of these details with little more attention. They constructed a band-diagonal unitary operator from the

following Hamiltonian

$$H = \begin{pmatrix} \ddots & \ddots & \ddots & & & & 0 \\ & \delta^* & 0 & \delta & & & \\ & & \delta^* & 0 & \delta & & \\ & & & \delta^* & 0 & \delta & \\ 0 & & & & \ddots & \ddots & \ddots \end{pmatrix},$$

where $\delta \in \mathbb{C}$ and $|\delta| \ll 1$. If we also consider small enough times steps we can construct a band-diagonal unitary from the following approximation

$$U = e^{-iHt/\hbar} \simeq 1 - iHt/\hbar. \tag{6.1}$$

Now we can understand better why this QCA definition leads us to some problems. We wish that in a model for QCA even for small or large number of steps the QCA be well defined. But clearly, even for a small time steps, Eq.(6.1) only represents a unitary evolution for small values of δ. For larger values of δ, other matrix elements farther off the diagonal would have to be nonzero in a very specific way to preserve unitarity. This would imply nonlocality, where the nonlocality here means that we can have interactions between distant cells.

Another try was the QCA defined by Watrous in [80], who made a deep research in the one-dimensional case. His first model can be viewed as direct quantization of a CA where at each step, instead of having only one configuration as in the ECA case, he worked with a superposition with several possible configurations. In this model he had problems with his definition of the transition function, in such a way that it would be represented by a nonunitary evolution, therefore problems like norm preservation turned up.

Given this previous problem, Waltrous developed a second model, in such a way that only after we apply two operators in the QCA we have a one time step in his automata [80]. In fact, his second version can be thought as a quantization of the partitioned CA. In his second model, he fixed the number of subcells being equal to three. The first operation that he defined was inside of each cell. In this operation from a initial state, we get the amplitude of probability to get all possible configurations. Then, the second and the last operation, before the automaton updates to the next time, is between the left and right neighbors of each cell, in such a way that it preserves the movement direction of some excitation, as we have done in the classical case. Actually, this last operation is divided in two parts, the first that interacts the boundary cells between neighbors and the last that only does a permutation between the left and right cell from the same cell. In Fig.(6.1), we illustrated this last operation in just one step.

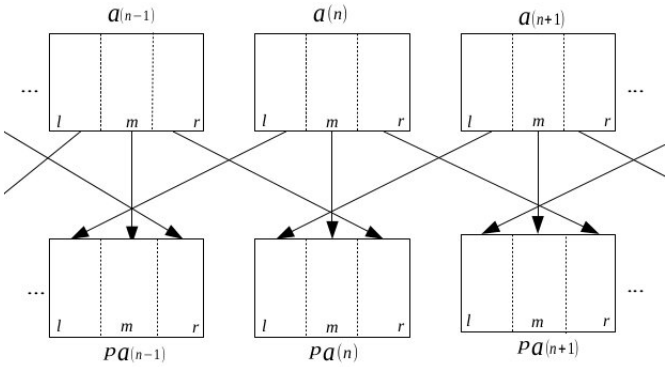


Figure 6.1: In this figure we illustrated the second part of the transition function proposed by Watrous. Each cell localized in position $n \in \mathbb{Z}$ has three subcells. First the excitation goes from the right (left) to the left (right) internal cell in the left (right) neighbor, after we have to apply a permutation operation, p, between the left and the right internal cells, in such a way that we have $pa(n,t) = (a_l(n-1,t), a_m(n,t), a_r(n+1,t))$, where $a(n,t)$ denotes the state of the cell indexed by n at the time t. The arrow in the figure indicates the final position of the excitation afterward we had applied these two parts of the same operation.

Although the QCA proposed by Watrous does not lead to any problem, neither nonlocality nor nonunitarity, its QCA structure is difficult and was not generalized to higher dimensions. Perhaps because of that, we did not see any application of this model so far.

As a final example of previous QCA definition, we will present the QCA proposed by Pérez and Cheung in [57], the main QCA model that inspired us in our new definition to quantum cellular automata. Like the other previous QCA models in [57], they were concerned in try to establish a natural extension of a classical CA, which can recover the classical CA behavior under reasonable assumptions.

Like Watrous the first step in their quantization of CA was to change the state space of a cell to reflect a quantum system. With this focus they could convert the alphabet of cellular automaton, Σ, into orthogonal basis states of a Hilbert space to every cell $x \in L$, with $L = \mathbb{Z}^d$, assigned $\mathscr{H}_\Sigma$ with the span of $\{|x\rangle, x \in \Sigma\}$. They also had to quantize the standard classical CA update rule. Then they replaced the classical cell update rule for a quantum analogue that acts on the Hilbert space.

Until here they got that their quantum cell update rule was given by a unitary operation with two restrictions:

- The operator must act on a finite subset of the lattice. Precisely,

  $$U_x : \mathscr{H}(\mathscr{N}_x) \to \mathscr{H}(\mathscr{N}_x), \tag{6.2}$$

  where $\mathscr{N}_x = \mathscr{N} + x \subseteq L$ is the finite neighbourhood about the cell x.

- The operator must commute with lattice translations of itself. Precisely, they require that

$$[U_x, U_y] = 0, \tag{6.3}$$

for all $x, y \in \mathbb{Z}^n$.

Until this moment in their quantization, they were closer to Wolfram's classification CA scheme. We can see this proximity when we analyze U_x closer. In Eq.(6.2) we can see that U_x is acting on cell x and in its entire neighborhood $\mathcal{N}_x$ in order to update cell x, in the same spirit that the transition function showed in (2.1). Thus, moving to another point y, for instance, a neighbor cell of x there is also an operator U_y that will update this cell, that also operates in its entire neighborhood. Therefore, there will be cases where $\mathcal{N}_x \cap \mathcal{N}_y \neq \varnothing$. These cases are not a problem to the classical CAs, in particular to the ones proposed by Wolfram the CA class that we are considering in this moment, since in classical computer architectures we can use two lattices in memory: one to store the current values of all points of the current CA state, and one to store the computed updated values. However, here we are thinking in terms of quantum architectures, where do a copy of the current CA state is forbidden [86] by the **no-cloning theorem**[1]. The way that they overcome this problem for quantum architectures was imposing the commutation relation (6.3). Then with this condition the operators $U_x, x \in \mathbb{Z}^n$ can be applied in parallel without the need to consider the ordering of the operators, allowing to apply them simultaneously. At this point, they become closer to the partitioned CA, since the operators can be applied simultaneously without the necessity of reading the state first.

In principle the global evolution of their QCA could be described as

$$U = \prod_x U_x, \tag{6.4}$$

whose action on the lattice is well-defined. However, as we know from [36], the authors argue that except for the trivial case, strictly local, unitary evolution of the whole QCA array is impossible (afterwards, in [48] David Meyer proved what they had claimed and called it a No-go lemma). Thus, if we try to use Eq.(6.4) to update the QCA state we can only establish trivial dynamics.

They could solve this problem by introducing another unitary operator which acts independent from the first. Therefore, there are two unitary operators as a update rule in QCA [57]. The first operator, corresponding to the *read* operation, the one defined in (6.2). The second operator, V_x, $x \in L$, corresponds to the *update* operation, and will only act on the single cell x.

Then, they established the following global update rule

$$E = VU = \left(\bigotimes_{x \in L} V_x\right)\left(\prod_{x \in \mathcal{N}} U_x\right),$$

[1]This theorem says that there is no acting of $\mathcal{H} \otimes \mathcal{H}$ unitary operator U such that for all $|\psi\rangle \in \mathcal{H}$,

$$U(|\psi\rangle \otimes |e_0\rangle) = |\psi\rangle \otimes |\psi\rangle,$$

where $|e_0\rangle = [1, 0, 0, \ldots, 0, 0]$.

which is space-homogeneous and has a well-defined action on the lattice.

We can now present a formal definition of the QCA model proposed in [57].

Definition 6.2. [Pérez and Cheung QCA] A Quantum Cellular Automata is a 5-tuple $(L, \Sigma, \mathcal{N}, U_0, V_0)$ consisting of:

1. a d-dimensional lattice of cells indexed by integers, $L = \mathbb{Z}^d$;
2. a finite set Σ of orthogonal basis states with $\mathcal{H}_\Sigma = \text{span}(\{|\sigma\rangle\}_{\sigma\in\Sigma})$. To each cell we assign a copy of $\mathcal{H}_\Sigma$;
3. a finite neighborhood scheme $\mathcal{N} \subseteq \mathbb{Z}^d$;
4. a local unitary read function $V_0 : (\mathcal{H}_\Sigma) \mapsto (\mathcal{H}_\Sigma)$;
5. a local unitary update function $U_0 : (\mathcal{H}_\Sigma)^{\otimes\mathcal{N}} \mapsto (\mathcal{H}_\Sigma)^{\otimes\mathcal{N}}$.

As before the update operation carries the further restriction that any two lattice translations U_x and U_y must commute for all $x, y \in L$.

We only presented three previous models, but there are more and we can find the main different definitions to QCA in [83]. Although there are many QCA models proposed, in the end some of these definitions were proved equivalent [4].

In this section, we gave emphasis to the model proposed by Perez and Cheung. The main reason is that it was this model that guided us to proposed our QCA definition. After we had employed the definition given in [57] into some examples we returned to the classical theory of CAs to see which one corresponded to the quantization obtained in [57]. We concluded that the block CA and the partitioning CA, the ones proposed by Toffoli and Margolus [76], were the closest models that should be quantized to get [57]. We also realized that doing a right quantization of these CA classes we also could lift the restriction given by (6.3), since in their evolution scheme we can avoid overlap between the operators that compose their transition functions, as we explained in chapter 2. The fact that the transition function of these CA classes is divided in two operators also helped us to see that these models would be a good candidate to the quantization since it would allow us to get a quantum version with a non-trivial evolution, thus avoiding the No-go lemma mentioned above. Then, after we have a better idea from the main classical and quantum definitions of cellular automata we tried to join the main characteristics of the partitioning and block CAs to proposed our definition to partitioned cellular automata, chapter 2. We notice that from our PCA definition there is more access to complicated geometries and dynamics since we are not restricted to only two operators at each time step. Moreover, we also realized that the quantization of our PCA definitions would be more natural, without imposing too many restrictions. Given that we move forward and we propose our quantum version for PCA, that we called by **partitioned unitary quantum cellular automata** (PUQCA).

6.2 PUQCA

As we explained in the last section, the QCA definition that we will present here is established directly from the quantization of the partitioned cellular automata presented in chapter 2, definition (2.1). We followed the same spirit of the quantization given in [57], which are the conversion of the alphabet Σ of each cell into a Hilbert space $\mathscr{H}_\Sigma$ and the replacement of the classical transition function $\mathscr{E}$ for a quantum analogue that acts on the Hilbert space. Therefore the main structures of the PCA definition showed in chapter 2 remain the same here, like the tiling concept and the partition of each cell into n subcells generating, as before, a finer description for the space where the automaton is defined – be it a lattice or the more general case of a graph. After all this review and discussion we are ready to define our partitioned unitary quantum cellular automata.

Definition 6.3 (PUQCA)**.** A Partitioned Unitary Quantum Cellular Automata is a 5-tuple $(L, \mathscr{N}, \Sigma, \{\mathscr{T}_i\}, \{W_i\})$ consisting of:

1. a d-dimensional lattice of cells indexed by integers $L = \mathbb{Z}^d$;
2. a finite neighborhood scheme $\mathscr{N} \subseteq L$;
3. a finite set Σ of orthogonal basis states with $\mathscr{H}_\Sigma = \text{span}\{|\sigma\rangle_{\sigma\in\Sigma}\}$. Each cell is divided in n subcells, and to the i-th subcell we assign a copy $\mathscr{H}_{\Sigma_i}$ of $\mathscr{H}_\Sigma$. The total space associated to each cell is then $\mathscr{H}_\Xi = \bigotimes_{i\in\{0,\ldots,n-1\}} \mathscr{H}_{\Sigma_i}$;
4. a finite set of tilings $\{\mathscr{T}_i\}_{i=0}^{N-1}$. Each tiling is the union of identical non-overlapping tiles, $\mathscr{T}_i = \bigcup_j T_j^{(i)}$, with each tile $T_j^{(i)}$ containing only subcells of neighboring cells.
5. a set of local unitary functions $\{W_i\}_{i=0}^{N-1}$. The same unitary W_i is applied to each tile $T_j^{(i)}$ of the tiling $\mathscr{T}_i$;

With this definition, the transition function $\mathscr{E} : (\mathscr{H}_\Xi)^{\otimes L} \mapsto (\mathscr{H}_\Xi)^{\otimes L}$, which updates the automaton state from the time t to $t+1$, is the given by

$$\mathscr{E} = \prod_{i=0}^{N-1} \left(\bigotimes_{T_j^{(i)}\in\mathscr{T}_i} W_i \right). \tag{6.5}$$

As we said the main structures of the PCA are still valid here. Thus, if we have an one-dimensional lattice where each cell has two subcells and the neighborhood scheme is $\mathscr{N}_i = \{i-1, i, i+1\}$, where $i \in L$ and its subcells are denoted by i_j, with $j \in \{0, n-1\}$ and we want to evolve our automata by employing two tilings, we can use exactly the same tilings applied in chapter 2: the first given by $\mathscr{T}_0 = \bigcup_{i\in\mathbb{Z}} T_i^{(0)}$ with each tile defined as $T_i^{(0)} = \{i_0, i_1\}$ and the second tiling given by $\mathscr{T}_1 = \bigcup_{i\in\mathbb{Z}} T_i^{(1)}$, where each tile is given by $T_i^{(1)} = \{i_1, (i+1)_0\}$. But now, instead

of permutation operators, we apply unitary operators in each tile, where the alphabet of each subcell was converted to a Hilbert space, thus

$$\begin{aligned} W_0: &\quad \left(\mathscr{H}_{\Sigma_0}\right)_i \otimes \left(\mathscr{H}_{\Sigma_1}\right)_i &\to \left(\mathscr{H}_{\Sigma_0}\right)_i \otimes \left(\mathscr{H}_{\Sigma_1}\right)_i \\ W_1: &\quad \left(\mathscr{H}_{\Sigma_1}\right)_i \otimes \left(\mathscr{H}_{\Sigma_0}\right)_{i+1} &\to \left(\mathscr{H}_{\Sigma_1}\right)_i \otimes \left(\mathscr{H}_{\Sigma_0}\right)_{i+1}, \end{aligned}$$

for all $i \in \mathbb{Z}$. Therefore, in this example we can explicit our transition function as

$$\mathscr{E} = \left(\bigotimes_{T_i^{(1)} \in \mathscr{T}_1} W_1 \right) \left(\bigotimes_{T_i^{(0)} \in \mathscr{T}_0} W_0 \right). \tag{6.6}$$

The cell representation presented in chapter 2 is maintained here Fig.(6.2-a), with the permutation operators replaced by unitary operators. In this example, we can notice that while the operators W_0 plays the role of the local unitary function V_0, the operators W_1 plays the role of the local unitary update function in the Pérez and Cheung QCA definition.

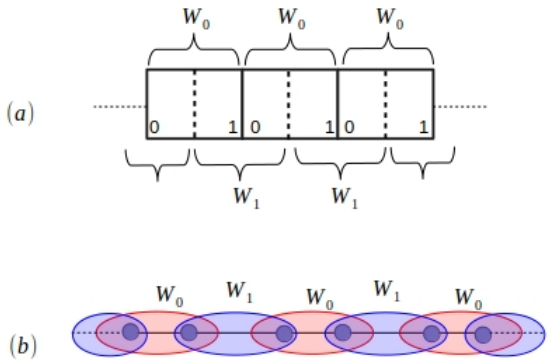


Figure 6.2: **1-dimensional automaton.** In top-panel (a) it is shown how each cell is split in two subcells, likewise we did in Fig.(2.5), but now in terms of unitary operators W_i. In the bottom-panel (b) it is shown the same 1-d automaton, but now in the graph perspective. The tiling $\mathscr{T}_0$ is represented by the red ellipses, while the tiling $\mathscr{T}_1$ is shown in blue. Figure from [22].

It is interesting to notice that, if we want, we can emulate and use the same structure used by Waltrous in [80] into our QCA. This can be done by employing three subcells in each cell and three tilings. In the first tiling $\mathscr{T}_0$ each tile is defined as $T_i^{(0)} = \{i_0, i_1, i_2\}$, where we have to apply the operator W_0 on each tile. In the second tiling $\mathscr{T}_1$ there are two tile structures, the ones that will allow the interaction between the neighbors cells $T_i^{(1)} = \{i_2, (i+1)_0\}$ and the others that do not suffer interaction in this step, the middle cells in Fig.(6.1), $T_i'^{(1)} = \{i_1\}$. We have a Swap operator being applied in the elements $T_i^{(1)}$, and an identity operator being applied in the elements $T_i'^{(1)}$. Finally, in order to preserve the movement direction employed in [80], there is an extra tiling $\mathscr{T}_2$, where the tiles are the same ones used in the first tiling $T^{(2)} = \{i_0, i_1, i_2\}$. In this part the operator W_2 is just a Swap operator between the subcells i_0 and i_2.

Although we did not mention in chapter 2, the above definition of the PUQCA, likewise for the PCA definition, immediately generalizes to QCA over a regular graph $G = G(V,E)$. In this situation, the neighborhood scheme is represented by the edge set E, and the tilings are defined over partitions of the graph G into complete subgraphs – in every tiling all the vertices are included, and the union of the tilings must contain all the edges. Within graph theory, tilings are usually called tessellations (see section 7.2). We can see how the one-dimensional example is represented in the "graph perspective" in Fig.(6.2-b). Another example of a PUQCA in the graph picture is shown in Fig. (6.3).

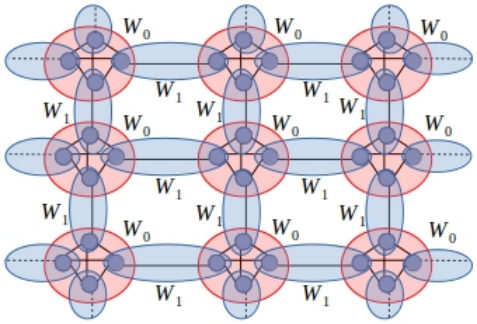


Figure 6.3: **Graph perspective: 2-dimensional automaton.** For the automaton defined over the 2-dimensional lattice, with four neighbors, each cell is transformed into a complete graph with 4 vertices (K4). The first tiling is shown in red, with corresponding unitary operation W_0 acting only on the subcells of each cell. The second tiling is depicted in blue, with the unitary operation W_1 being responsible for the interaction between neighboring cells. Figure from [22].

6.2.1 Quantum lattice gases

The main goal here is to show how we can apply the PUQCA to interesting problems in physics, as we did for the partitioned CA in part I of this thesis. We showed how the Brownian motion and the random walk problem, chap.(3), are described in terms of the partitioned CA. Then our idea here is to do the same for the QCA. Having this goal in mind we will see how the quantum Brownian motion (QBM), usually known as coined quantum walk, can be described in terms of the QCA. While the problems employed to the PCA belong to the classical lattice gases, the QBM belongs to quantum lattice gases. The basic principles are the same in both the classical and quantum cases: one starts with QCA model which describes particles on the lattice. One can then take the continuous limit of such CA and show that in this limit, the behavior of the CA mimics a well-know differential equation.

Our strategy now is to take a similar path that we have done in the classical case. We will start by introducing the quantum version of the Brownian motion and then we will describe the

QBM problem using the PUQCA, without taking its continuous limit. The continuous limit which we will analyze here is the one that yields the Dirac equation in (1+1) dimension.

6.2.1.1 Quantum Brownian motion

Based upon the previous literature about the QW [42], we will see that in fact what people usually call by QW is really the QBM.

Let us define what people usually call by discrete time quantum walk which is our QBM, as we will see. We will just define the model in one dimension, on the line or the circle. For general case see [42, 58].

The QBM is described by the tensor product of two Hilbert spaces,

$$\mathscr{H} = \mathscr{H}_C \otimes \mathscr{H}_L,$$

where $\mathscr{H}_L$ is the Hilbert space spanned by the positions of the particle and $\mathscr{H}_C$ is a *coin*-space. For a line $\mathscr{H}_L$ is spanned by basis states $\{|i\rangle : i \in \mathbb{Z}\}$, if we work on a circle of size N we have $\mathscr{H}_L = \{|i\rangle : i = 0, \ldots N-1\}$. For $\mathscr{H}_C$ we take a two dimension coin, which is spanned by two basis states

$$\{|\uparrow\rangle, |\downarrow\rangle\}, \tag{6.7}$$

that we can imagine as the states of a spin-1/2 particle.

The conditional translation of the system can be described by the following unitary operation $S : \mathscr{H} \to \mathscr{H}$, known as shift operator,

$$S = |\uparrow\rangle\langle\uparrow| \otimes \sum_i |i+1\rangle\langle i| + |\downarrow\rangle\langle\downarrow| \otimes \sum_i |i-1\rangle\langle i|, \tag{6.8}$$

where the index i runs over $\mathbb{Z}$ in the case of a line. In the case of circle we have $0 \leq i \leq N-1$ and the operation related with the particle position in S have to change to sum modulo N. As we can see the S transforms the basis state $|\uparrow\rangle \otimes |i\rangle$ to $|\uparrow\rangle \otimes |i+1\rangle$ and $|\downarrow\rangle \otimes |i\rangle$ to $|\downarrow\rangle \otimes |i-1\rangle$. For this reason the shift operator presented above is known as the **moving**, given that basis state remains with the same coin state. Thus, the walker continues moving into the same direction.

The first step of the QBM is a rotation in the coin-space, by $C : \mathscr{H}_C \to \mathscr{H}_C$. The unitary transformation C is arbitrary and we can define a rich family of QBM with different behaviors by modifying C. Choosing different operators for C corresponds to choosing different probabilities of collision with some molecule in the classical BM.

Then we can define the unitary operator $U : \mathscr{H} \to \mathscr{H}$ which acts on the tensor product of Hilbert space $\mathscr{H}$ by

$$U = S \cdot (C \otimes I), \tag{6.9}$$

where I is the identity matrix that acts in the walker position Hilbert space $I : \mathscr{H}_L \to \mathscr{H}_L$.

First we will analyze a balanced unitary coin, called Hadamard coin

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

With this coin we are able to recover, if we measure our state after each interaction, the classical probability distribution with probability 1/2. The unitary operator Eq.(6.9) takes the following form

$$U = |\uparrow\rangle\langle +| \otimes \sum_i |i+1\rangle\langle i| + |\downarrow\rangle\langle -| \otimes \sum_i |i-1\rangle\langle i|, \tag{6.10}$$

where

$$\begin{aligned} H|\uparrow\rangle &= |+\rangle = \frac{(|\uparrow\rangle + |\downarrow\rangle)}{\sqrt{2}}, \\ H|\downarrow\rangle &= |-\rangle = \frac{(|\uparrow\rangle - |\downarrow\rangle)}{\sqrt{2}}, \end{aligned}$$

where we are identifying

$$|\uparrow\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, |\downarrow\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

In fact U is the time-evolution unitary operator for this model,

$$|\psi_{t+1}\rangle = U|\psi_t\rangle. \tag{6.11}$$

Now let us apply U successively in the initial state

$$|\psi_0\rangle = |\downarrow\rangle \otimes |0\rangle,$$

where $|0\rangle = (\ldots 010\ldots)^T$. We will see the induced probability distribution on the position. From Eq.(6.11) we have $|\psi_{t+l}\rangle = U^l|\psi_t\rangle$ and then

$$\begin{aligned} |\psi_1\rangle &= \frac{1}{\sqrt{2}}(|\uparrow\rangle \otimes |1\rangle - |\downarrow\rangle \otimes |-1\rangle), \\ |\psi_2\rangle &= \frac{1}{2}[|\uparrow\rangle \otimes |2\rangle - (|\uparrow\rangle - |\downarrow\rangle) \otimes |0\rangle + |\downarrow\rangle \otimes |-2\rangle], \\ |\psi_3\rangle &= \frac{1}{2\sqrt{2}}[|\uparrow\rangle \otimes |3\rangle + |\downarrow\rangle \otimes |1\rangle + (|\uparrow\rangle - 2|\downarrow\rangle) \otimes |-1\rangle - |\downarrow\rangle \otimes |-3\rangle], \\ |\psi_4\rangle &= \frac{1}{4}[|\uparrow\rangle \otimes |4\rangle + (|\uparrow\rangle + |\downarrow\rangle) \otimes |2\rangle - (|\uparrow\rangle + |\downarrow\rangle) \otimes |0\rangle + (3|\downarrow\rangle - |\uparrow\rangle) \otimes |-2\rangle - |\downarrow\rangle \otimes |-4\rangle]. \end{aligned}$$

Evaluating the probability of finding the particle at position i at time t, $P_t(i) = |\langle i|\psi_t\rangle|^2$, we can construct, from the states above, a table which gives us the probability distribution evolution, see figure (6.4). We can see in (6.4) that only until $t = 2$ we have the same probabilities from its classical version. At $t = 3$ we can already see a difference due to the quantum nature of the system. Furthermore from this initial condition and coin operator the QBM is asymmetric

$t\backslash i$	-4	-3	-2	-1	0	1	2	3	4
0					1				
1				$\frac{1}{2}$		$\frac{1}{2}$			
2			$\frac{1}{4}$		$\frac{1}{2}$		$\frac{1}{4}$		
3		$\frac{1}{8}$		$\frac{5}{8}$		$\frac{1}{8}$		$\frac{1}{8}$	
4	$\frac{1}{16}$		$\frac{5}{8}$		$\frac{1}{8}$		$\frac{1}{8}$		$\frac{1}{16}$

Figure 6.4: Walker probability distribution with $|\psi_0\rangle = |\downarrow\rangle \otimes |0\rangle$, in the initial state.

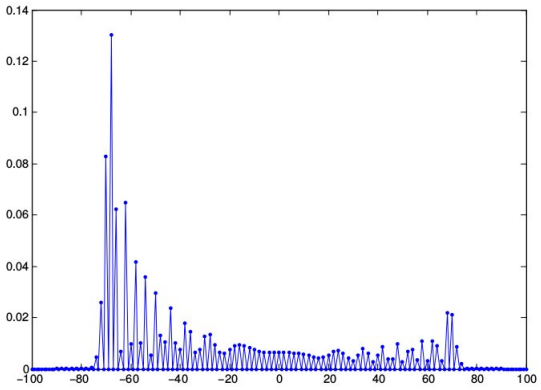


Figure 6.5: Probability distribution of the QBM with Hadamard coin and initial state $|\psi_0\rangle = |\downarrow\rangle \otimes |0\rangle$. The x-axis gives us the positions while y-axis give us the probabilities associated with the position at $t = 100$.

with a drift to left. A rather distinct distribution becomes more evident when we observe the probability distribution after many time steps. In Fig.(6.5) we plotted the case for $t = 100$. This asymmetry in the probability distribution, which is evident in Fig.(6.5), arises from the fact that the matrix H treats the two direction $|\uparrow\rangle$ and $|\downarrow\rangle$ differently, since $H|\uparrow\rangle = 1/\sqrt{2}\,(|\uparrow\rangle + |\downarrow\rangle)$ while $H|\downarrow\rangle = 1/\sqrt{2}\,(|\uparrow\rangle - |\downarrow\rangle)$. Although $H|\uparrow\rangle$ and $H|\downarrow\rangle$ lead us to the same probability distribution at one single walker step, this negative signal in the last equation leave us to more cancellations of one side in relation to another after many applications of the unitary U. There are two ways to obtain a symmetric distribution [59]. We can start with the state vector

$$|\psi_0\rangle = \frac{1}{\sqrt{2}}(|\uparrow\rangle + i\,|\downarrow\rangle) \otimes |0\rangle\,,$$

since H does not introduce any complex amplitudes, and then $|\uparrow\rangle$ will stay real whereas $|\downarrow\rangle$ will be purely imaginary, or we can use a different (balanced) coin, namely

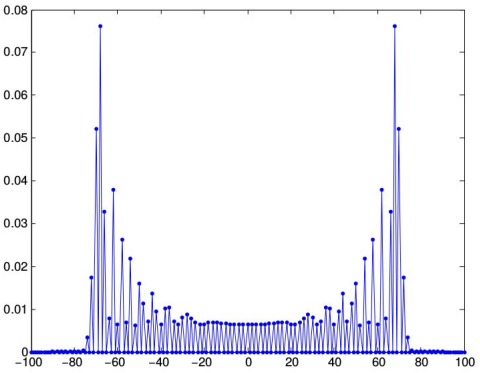
$$Y = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & i \\ i & 1 \end{pmatrix}.$$

This coin treats the right and left direction in the same way.

Working with Y as the coin, in Fig.((6.6)-a) we show the table only for five-time steps and in Fig.((6.6)-a) we plot the probability distribution after 100 steps, obtained from a computer simulation.

$t\backslash i$	-4	-3	-2	-1	0	1	2	3	4
0					1				
1				$\frac{1}{2}$		$\frac{1}{2}$			
2			$\frac{1}{4}$		$\frac{1}{2}$		$\frac{1}{4}$		
3		$\frac{1}{8}$		$\frac{3}{8}$		$\frac{3}{8}$		$\frac{1}{8}$	
4	$\frac{1}{16}$		$\frac{3}{8}$		$\frac{1}{8}$		$\frac{3}{8}$		$\frac{1}{16}$

a)



b)

Figure 6.6: Symmetric probability distribution of the QBM.

As the classical BM we can extract the recurrence relation for QBM. Let us do that by choosing a $SU(2)$ operator

$$C = \begin{pmatrix} p & q \\ q & p \end{pmatrix}. \tag{6.12}$$

with p, $q \in \mathbb{C}$ respecting the unitarity constraint $|p|^2 + |q|^2 = 1$ and $p^* q + q^* p = 0$. Thus, if the state of the system at time t is

$$\big|\psi(t)\big\rangle = \sum_{i \in L} \Big(\psi^{\uparrow}(i,t) |\uparrow\rangle \otimes |i\rangle + \psi^{\downarrow}(i,t) |\downarrow\rangle \otimes |i\rangle \Big), \tag{6.13}$$

where $\psi^s(i,t) = (\langle s| \otimes \langle i|)\big|\psi(t)\big\rangle$ with $s \in \{\uparrow, \downarrow\}$ is the amplitude of the walker being located at the vertex i with $\uparrow$ ($\downarrow$) as its coin state. Then the after we apply $(C \otimes I)$ into the initial state gives,

$$\begin{aligned} (C \otimes I)\big|\psi(t)\big\rangle &= \sum_{i \in L} \Big[\Big(q\psi^{\downarrow}(i,t) + p\psi^{\uparrow}(i,t) \Big) |\uparrow\rangle \otimes |i\rangle \\ &+ \Big(p\psi^{\downarrow}(i,t) + q\psi^{\uparrow}(i,t) \Big) |\downarrow\rangle \otimes |i\rangle \Big]. \end{aligned}$$

As the next and last step, in order to update our state to $t+1$, we apply the shift operator

$$\begin{aligned} S \cdot (C \otimes I)\big|\psi(t)\big\rangle = \big|\psi(t+1)\big\rangle &= \sum_{i \in L} \Big[\Big(q\psi^{\downarrow}(i,t) + p\psi^{\uparrow}(i,t) \Big) |\uparrow\rangle \otimes |i+1\rangle \\ &+ \Big(p\psi^{\downarrow}(i,t) + q\psi^{\uparrow}(i,t) \Big) |\downarrow\rangle \otimes |i-1\rangle \Big]. \end{aligned}$$

From this last expression, we can immediately write the recurrence relations that govern the quantum BM dynamics

$$\begin{aligned} \psi^{\uparrow}(i,t+1) &= q\psi^{\downarrow}(i-1,t)+p\psi^{\uparrow}(i-1,t), \\ \psi^{\downarrow}(i,t+1) &= p\psi^{\downarrow}(i+1,t)+q\psi^{\uparrow}(i+1,t). \end{aligned} \tag{6.14}$$

Now, in order to contrast the QBM with the recurrence relations achieved for its classical counterpart Eq.(3.1), we will get recurrence relations in terms of probabilities. From Quantum mechanics theory, we know that the Born's rule yields the probabilities, which for this case are given by

$$P^{s}_{t+1}(i) = \left|\langle\psi(t+1)|(|s\rangle\otimes|i\rangle)\right|^2 = \left|\psi^{s}(i,t+1)\right|^2, \tag{6.15}$$

where $P^{s}_{t+1}(i)$ with $s\in\{\uparrow,\downarrow\}$ is the probability of the particle being found at position i at time $t+1$ in the coin state $|\uparrow(\downarrow)\rangle$. Then, putting Eq.(6.14) into Eq.(6.15) we get

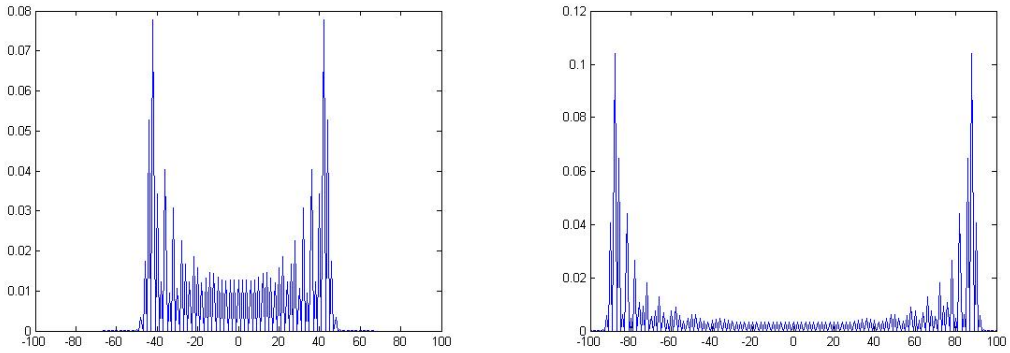
$$\begin{aligned} P^{\uparrow}_{t+1}(i) &= |p|^2 P^{\uparrow}_{t}(i-1)+|q|^2 P^{\downarrow}_{t}(i-1) \\ &+ pq^{*}a^{\uparrow}_{t}(i-1)a^{\downarrow *}_{t}(i-1)+h.c, \\ P^{\downarrow}_{t+1}(i) &= |q|^2 P^{\uparrow}_{t}(i+1)+|p|^2 P^{\downarrow}_{t}(i+1) \\ &+ pq^{*}a^{\uparrow}_{t}(i+1)a^{\downarrow *}_{t}(i+1)+h.c, \end{aligned} \tag{6.16}$$

where $a^{s}_{t}(i)$ are the amplitudes terms of the particle being found at position i at time t in the state $s\in\{\uparrow,\downarrow\}$. From these equations above we can see that the two first terms are exactly the same in Eqs.(3.1), however instead of working with $\uparrow,\downarrow$, we worked with r, l respectively. The difference from the classical BM came from the interference terms, which is the quantum signature of the problem. In addition, we did two plots for the probability distribution employing two different values for p. In case we were describing the quantum version of the random walk problem we would expect both plots to have the same width for different p values concentrated in different points, similarly to, the results showed in Fig.(3.6). But we do not see this in Fig.(6.7). These results agree with what we get in Fig.(3.6) since both results are symmetric around the origin. Besides, the probability distribution is more concentrated around the origin when we employ small values for p, which strongly suggests to us that p is the probability amplitude associated with no collision with the water molecules, chapter 3.

Another distinct characteristic of QBM is its propagation speed. The symmetric BM after T steps has a variance $\sigma^2 = T$. On the other hand it can be shown that QBM has a variance that scales with $\sigma^2 \sim T^2$ which implies that the expected distance from the origin is of order $\sigma \sim T$, see [59]. The QBM propagates quadratically faster.

6.2.1.2 QBM by quantum lattice gases

Like we have done for PCA, we will propose a PUQCA which is capable to simulate the QBM. To start with this translation our first step is to correspond to a cellular automaton cell each



a) $|p|^2 = 0.2$ b) $|p|^2 = 0.8$

Figure 6.7: Different from the classical BM the highest probability to find the particle is far way from the origin. However for both cases, QBM and BM the value of $|p|^2$ determines the speed of the distribution of probability.

walker position $i \in L$. As we can notice the structure here is very similar to what we have done in the classical case (3.1): a partition scheme in a one-dimensional lattice. Then, each cell has two subcells, which means now two qubits per cell, i.e., to each cell is assigned a tensor product of two Hilbert space of two dimensions,

$$\mathscr{H}_\Sigma = \mathscr{H}_{\Sigma_0} \otimes \mathscr{H}_{\Sigma_1} \cong \mathbb{C}^2 \otimes \mathbb{C}^2.$$

These two subcells encode the two orthogonal states of the coin space $\mathscr{H}_C$, Eq.(6.7), of the QBM. Moreover, we will keep with the convention adopted for BM. Thus, instead of labeling these two Hilbert spaces by 0 and 1, we will denote them by left, (l), and right, (r) respectively $\mathscr{H}_\Sigma = \mathscr{H}_{\Sigma_l} \otimes \mathscr{H}_{\Sigma_r}$. This will allow us keep to the movement interpretation.

Like before we will only work with just one excitation or particle, but now we associate an amplitude of probability for each cell

$$\Psi(i,t) = \Psi_l(i,t) + \Psi_r(i,t), \tag{6.17}$$

where $\Psi_l(i,t)$ and $\Psi_r(i,t)$ are the amplitudes for a particle to enter cell i from left and right, respectively. This is the Feynman path sum for one time step of the evolution operator, where the amplitude $\Psi(i,t)$ is the propagator of evolution. We then encode the QBM state $|\downarrow\rangle \otimes |i\rangle$ with the automaton state $\left|\ldots,(0,0)_{i-1},(1_l,0_r)_i,(0,0)_{i+1},\ldots\right\rangle$ while the state $|\uparrow\rangle \otimes |i\rangle$ is written as $\left|\ldots,(0,0)_{i-1},(0_l,1_r)_i,(0,0)_{i+1},\ldots\right\rangle$. In this way, an excited subcell (a qubit in the 1 state) indicates the walker position and its movement direction. All the encoding is done within the single excitation subspace.

To finish this translation we need to show the tilings and the unitary operators required in order to mimic the QBM dynamics. In this step, we face one dilemma. In chapter 3 we described the BM by PCA using three tilings and consequently three operators. On the other hand, we saw

that the quantum BM demands only two unitaries, the coin, and the shift operator. However, the PUQCA structure does not allow us to apply the moving shift operator Eq.(6.8) by a single unitary, let us understand it better. Thinking in terms of QCA, from the translation showed until now, this operator, Eq.(6.8), means that if we have a single excitation located in the right subcell r of cell i, which is related with $\uparrow$, we have to build an operator for the QCA that moves this excitation to the subcell r located at cell $i+1$. If we have such local operator it means that we are interacting the subcells $(i)_r$ and $(i+1)_r$, which at first glance does not seem to be a problem, since we can do this operation applying the SWAP operator between these subcells. However, from the PUQCA definition, which is a computational model invariant by translation, it implies that we have this kind of interaction happening for all $i \in L$. In particular, there is the same kind of interaction between the subcells $(i-1)_r$ and $(i)_r$. Consequently we will get two operators acting simultaneously in the subcell $(i)_r$, in fact, this overlap of operations is happening in all subcells. Therefore, it would lead us to have a tiling, related to this operation, with overlapping tiles, which is not allowed from our PUQCA definition. Furthermore, although interaction with overlaps are allowed in other QCA models this interaction with SWAPs is not well defined. For instance, for the one proposed by Pérez and Cheung [57], these overlaps are possible. These interactions mean that we have a SWAP operator as a local unitary update function between the same subcells showed above. However, we can see that we will get update functions that do not commute,

$$[SWAP_{i_r,(i+1)_r} \otimes I_{(i+2)_r}, I_{i_r} \otimes SWAP_{(i+1)_r,(i+2)_r}] \neq 0,$$

for all $i \in L$ and then a QCA dynamics not allowed. Therefore we will employ, as we did in the description of the BM, three tilings.

Indeed these three tilings are the same employed in section (3.1). The first one $\mathcal{T}_0$ is related to the coin operator, where the tiles are the subcells from the same cell $T_i^{(0)} = \{(i)_l, (i)_r\}$. The unitary W_0 associated to this tiling is of the same form as the unitary in the coin C Eq.(6.12),

$$W_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & p & q & 0 \\ 0 & q & p & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{6.18}$$

where again the requirements $|p|^2 + |q|^2 = 1$ and $p^*q + q^*p = 0$ are necessary. This operator acts on a two-qubit, system

$$W_0 : \mathcal{H}_{\Sigma_l} \otimes \mathcal{H}_{\Sigma_r} \to \mathcal{H}_{\Sigma_l} \otimes \mathcal{H}_{\Sigma_r}.$$

Despite the fact that W_0 acts in the full space, it only acts non-trivially on the single excitation subspace. A general QBM state is then translated as

$$|\Psi(t)\rangle = \sum_{i \in L} \left(\Psi_l(i,t) \left| \ldots, (0_l 0_r)_{i-1}, (1_l 0_r)_i, (0_l 0_r)_{i+1}, \ldots \right\rangle + \Psi_r(x,t) \left| \ldots, (0_l 0_r)_{i-1}, (0_l 1_r)_i, (0_l 0_r)_{i+1}, \ldots \right\rangle \right), \tag{6.19}$$

where $\Psi(i,t)_{l(r)}$ is the probability amplitude of finding an "excitation" at the subcell ($l(r)$) of the i-th cell at time t. Thus, after we apply the evolution corresponding to the first tiling $\left(\bigotimes_{T_j^{(0)}\in\mathscr{T}_0} W_0\right)|\Psi(t)\rangle$, we get

$$\begin{aligned} &= \sum_{i\in L}\left\{(p\Psi_l(i,t)+q\Psi_r(i,t))\left|\ldots,(0_l0_r)_{i-1},(1_l0_r)_i,(0_l0_r)_{i+1},\ldots\right\rangle\right. \\ &+ \left.(p\Psi_r(i,t)+q\Psi_l(i,t))\left|\ldots,(0_l0_r)_{i-1},(0_l1_r)_i,(0_l0_r)_{i+1},\ldots\right\rangle\right\}, \end{aligned}$$

since the action of W_0 at each cell is

$$\begin{aligned} W_0|0_l1_r\rangle &= p|0_l1_r\rangle + q|1_l0_r\rangle, \\ W_0|1_l0_r\rangle &= q|0_l1_r\rangle + p|1_l0_r\rangle. \end{aligned}$$

Now we have to move to the tiling that will interact the cells. As we explained above we can not apply the moving operator in one single operation. Let us go back to the shift operator employed into QBM, Eq.(6.8). We can split this operation in two parts as follows:

$$S = (X\otimes I)\cdot S_{\text{f-f}}, \tag{6.20}$$

where

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \tag{6.21}$$

is being applied into the coin space $X:\mathscr{H}_C\to\mathscr{H}_C$, I is the identity operator that acts in the walker position Hilbert space $I:\mathscr{H}_L\to\mathscr{H}_L$ and $S_{\text{f-f}}$ is

$$S_{\text{f-f}} = |\downarrow\rangle\langle\uparrow|\otimes\sum_i|i+1\rangle\langle i| + |\uparrow\rangle\langle\downarrow|\otimes\sum_i|i-1\rangle\langle i|. \tag{6.22}$$

Such operator is known as the flip-flop shift operator, given that $S_{\text{f-f}}^2 = \mathbb{1}$.

From the decomposition above, Eq.(6.20), we can understand better from where the next two tilings come from: one is related to $S_{\text{f-f}}$ and the other with X. Let us start with the flip-flop operator. When we interpret this operator, Eq.(6.22), in the QCA perspective we can see that if an excitation is located at the subcell i_r this operator moves the excitation to the subcell $(i+1)_l$, and if the excitation is located at the subcell i_l this operator moves the excitation to the subcell $(i-1)_r$. Therefore we can put all these subcells that are interacting in the second tiling $\mathscr{T}_1$ i.e. $T_i^{(1)} = \{i_r,(i+1)_l\}$. As we conclude above, the SWAP operator plays the role of the flip-flop operator in the QCA picture, that means we need to apply $W_1 = SWAP$ into the tiles $T_i^{(1)}$. Thus, after the evolution due these two tilings, $\left(\bigotimes_{T_j^{(1)}\in\mathscr{T}_1} W_1\right)\left(\bigotimes_{T_j^{(0)}\in\mathscr{T}_0} W_0\right)$ our general state Eq.(6.19) is updated to

$$\begin{aligned} &= \sum_{i\in L}\left\{(p\Psi_l(i,t)+q\Psi_r(i,t))\left|\ldots,(0_l1_r)_{i-1},(0_l0_r)_i,(0_l0_r)_{i+1},\ldots\right\rangle\right. \\ &+ \left.(p\Psi_r(i,t)+q\Psi_l(i,t))\left|\ldots,(0_l0_r)_{i-1},(0_l0_r)_i,(1_l0_r)_{i+1},\ldots\right\rangle\right\}. \end{aligned}$$

As a final step we need to translate the action of X in terms of QCA. From the X definition, Eq.(6.21), we see that it flips the coin state: $X|\uparrow(\downarrow)\rangle = |\downarrow(\uparrow)\rangle$. From the X action we can see that it is responsible to preserve the walker movement direction transforming the $S_{\text{f-f}}$ to the moving shift operator. From the QCA perspective we can see that it is only a SWAP between the subcells from the same cell. Therefore we can see that in the last tiling $\mathscr{T}_2$ the tiles are $T_i^{(2)} = \{i_l, i_r\}$ and $W_2 = SWAP$. Therefore after the evolution of this last tiling, that completes the action of the transition function $\mathscr{E} = \left(\bigotimes_{T_j^{(2)}\in\mathscr{T}_2} W_2\right)\left(\bigotimes_{T_j^{(1)}\in\mathscr{T}_1} W_1\right)\left(\bigotimes_{T_j^{(0)}\in\mathscr{T}_0} W_0\right)$ we get the general state for $t+1$

$$\begin{aligned}|\Psi(t+1)\rangle &= \sum_{i\in L}\left\{(p\Psi_l(i,t)+q\Psi_r(i,t))\left|\ldots,(1_l0_r)_{i-1},(0_l0_r)_i,(0_l0_r)_{i+1},\ldots\right\rangle\right.\\ &+ \left.(p\Psi_r(i,t)+q\Psi_l(i,t))\left|\ldots,(0_l0_r)_{i-1},(0_l0_r)_i,(0_l1_r)_{i+1},\ldots\right\rangle\right\}.\end{aligned}$$

From this last result we can immediately obtain the recurrence relations that describe the automaton dynamics:

$$\begin{aligned}\Psi_r(i+1,t+1) &= p\Psi_r(i,t)+q\Psi_l(i,t),\\ \Psi_l(i-1,t+1) &= p\Psi_l(i,t)+q\Psi_r(i,t).\end{aligned} \tag{6.23}$$

Now, when we compare this result with the recurrence relation in terms of amplitudes established in Eq.(6.14) we can easily conclude that they are equal. Therefore, we saw how we can codify the QBM in terms of PUQCA since both dynamics are identical at every time-step.

Before we move to the next section it is important to say that there is an alternative method to use only two tilings for this same problem. Despite the fact that we cannot apply in a single shot the moving operator in the QCA formalism, we can absorb X in the coin space and then apply only two operators for the QBM dynamics, as follows,

$$S\cdot(C\otimes I) = (X\otimes I)\cdot S_{f-f}\cdot(C\otimes I) \rightarrow S_{f-f}\cdot\left(C'\otimes I\right),$$

where $C' = C\cdot X$ as long as we do corrections on the initial and final states. In order to see that, suppose that we have to apply the unitary t times, then

$$\begin{aligned}U^t &= \left((X\otimes I)\cdot S_{f-f}.(C\otimes I)\right)^t\\ &= (X\otimes I)\cdot\left(S_{f-f}.(C\cdot X\otimes I)\right)^{t-1}\cdot S_{f-f}.(C\otimes I)\\ &= (X\otimes I)\cdot\left(S_{f-f}.(C\cdot X\otimes I)\right)^{t-1}\cdot S_{f-f}.(C\otimes I)\cdot\left(X\cdot X^{-1}\otimes I\right)\\ &= (X\otimes I)\cdot\left(S_{f-f}.\left(C'\otimes I\right)\right)^t\cdot\left(X^{-1}\otimes I\right).\end{aligned}$$

We can conclude from the calculations above that we can apply only two operators if we translated the initial state $\left|\psi(0)\right\rangle$ to $X^{-1}\left|\psi(0)\right\rangle$ and the final state to $X^{-1}\left|\psi(t)\right\rangle$. Then, we can do the translation for the QCA employing two tilings, relating C' to W_0 as long as we do corrections in the initial and final QCA states.

6.2.1.3 Weyl equation via QCA

Like in chapter 3, where we saw how the PCA can be applied as a numerical method for the diffusion equation, the ones that describe the Brownian motion and random walk, we will see how we can apply the PUQCA as a numerical method for the Weyl equation, a relativistic wave equation proposed by Hermann Weyl in 1928 [82] for describing massless spin-1/2 particles called Weyl fermions. This equation is a two-component equation to describe massless spin-1/2. This result was obtained extending the results of the relativistic equation that describes massive spin-1/2 particles, Dirac equation [11], achieved by Paul Dirac. The Weyl equation is essentially the Dirac equation for massless particles $m = 0$, and its compact form, using the natural units, is given by

$$i\gamma_\mu \partial^\mu \psi = 0, \tag{6.24}$$

where γ^μ with $\mu = 0,1,2,3$ are the gamma matrices that obey the Clifford algebra,

$$\{\gamma^\mu, \gamma^\nu\} = 2\eta^{\mu\nu} I,$$

where $\eta^{\mu\nu}$ is the Minkowski metric and I is the identity matrix. The Weyl spinor in Eq.(6.24) is composed by the left, $\psi^{(-)}$, and right handed spinors,[2] $\psi^{(+)}$

$$\psi = \begin{pmatrix} \psi^{(-)} \\ \psi^{(+)} \end{pmatrix}. \tag{6.25}$$

The goal of this section is to show how we can apply the PUQCA to simulate the Weyl equation in flat two-dimensional (1+1) spacetime, like it was done in [34] via quantum walks. In this flat two-dimensional (1+1) spacetime, that gives only $\mu = 0,1$ in Eq.(6.24), the Clifford algebra can be represented by matrices acting on two-component spinors $\gamma_0 = \sigma_1$ and $\gamma^1 = -\sigma_1\sigma_3 = i\sigma_2$, where

$$\sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

are the Pauli matrices. Now, applying these matrices into Eq.(6.24) and from the fact that $\partial^0 = \partial/\partial t$ and $\partial^1 = \partial/\partial x$ we get two first-order PDEs

$$\begin{aligned} \frac{\partial \psi^{(-)}(x,t)}{\partial t} - \frac{\partial \psi^{(-)}(x,t)}{\partial x} &= 0, \\ \frac{\partial \psi^{(+)}(x,t)}{\partial t} + \frac{\partial \psi^{(+)}(x,t)}{\partial x} &= 0. \end{aligned} \tag{6.26}$$

The discrete version of Eqs.(6.26) above can be achieved from the PUQCA only choosing $p = 1$ and consequently $q = 0$ in the operator W_0 used in Eq.(6.18). Doing that our recurrence relation established in Eq.(6.23) changes to

$$\begin{aligned} \Psi_r(x,t+1) &= \Psi_r(x-1,t), \\ \Psi_l(x,t+1) &= \Psi_l(x+1,t), \end{aligned} \tag{6.27}$$

[2] We say that a particle is right (left) handed when the its spin is (anti)parallel to p, its momentum.

where we translated the equations above and changed the notation $i \in L$ to $x \in L$. The function Ψ_l plays the role of the left handed spinor, Ψ_r plays the role of the right handed spinor.

In order to confirm the claim about Eqs.(6.27) to be the discrete version of the PDEs showed in Eq.(6.26) we have to take the continuum limit of Eqs.(6.27). To start with this calculation we first need to move from the discrete time and space $x, t \in \mathbb{Z}$ to continuous variables $x, t \in \mathbb{R}$. Thus, our next step is to analyze the behavior of the quantum automaton in the limit of infinitely short time step τ and a infinitesimal lattice spacing λ, like we did in chapter 3. Then, rewriting Eq.(6.27) employing continuous variables yields

$$\begin{aligned} \Psi_r(x, t+\tau) &= \Psi_r(x-\lambda, t), \\ \Psi_l(x, t+\tau) &= \Psi_l(x+\lambda, t). \end{aligned} \tag{6.28}$$

There is a simple way to handle this computation. We can subtract on both sides of the top equation the spinors Ψ_r and subtract in both sides of the bottom equation the spinors Ψ_l, then

$$\begin{aligned} \Psi_r(x, t+\tau) - \Psi_r(x, t) &= \Psi_r(x-\lambda, t) - \Psi_r(x, t), \\ \Psi_l(x, t+\tau) - \Psi_l(x, t) &= \Psi_l(x+\lambda, t) - \Psi_l(x, t). \end{aligned}$$

Now if we multiply in both equations the left equality above by τ/τ and the right by λ/λ we get

$$\begin{aligned} \tau \frac{\Psi_r(x, t+\tau) - \Psi_r(x, t)}{\tau} &= \lambda \frac{\Psi_r(x-\lambda, t) - \Psi_r(x, t)}{\lambda}, \\ \tau \frac{\Psi_l(x, t+\tau) - \Psi_l(x, t)}{\tau} &= \lambda \frac{\Psi_l(x+\lambda, t) - \Psi_l(x, t)}{\lambda}, \end{aligned}$$

or

$$\begin{aligned} \frac{\Psi_r(x, t+\tau) - \Psi_r(x, t)}{\tau} &= \frac{\lambda}{\tau} \frac{\Psi_r(x-\lambda, t) - \Psi_r(x, t)}{\lambda}, \\ \frac{\Psi_l(x, t+\tau) - \Psi_l(x, t)}{\tau} &= \frac{\lambda}{\tau} \frac{\Psi_l(x+\lambda, t) - \Psi_l(x, t)}{\lambda}. \end{aligned}$$

From the equations above, it is clear from what we learned in chapter 4 that we are dealing with a first order approximation for the first derivatives. Besides, except the right part of the equation for Ψ_r, the approximation employed is called forward difference, whereas the right part for Ψ_r is the backward difference. Moreover, in this continuous limit computation, we can see that we are dealing with equations that have linear dispersion relation λ/τ. In fact, this ratio λ/τ gives to us the velocity of the particle propagation. Since we are dealing with a massless relativistic particle its velocity is c and as we are using natural units $c = 1$. Finally, we can take the limit of both terms λ and τ to zero in the last expression to confirm that both equations are the discrete version of Eq.(6.26).

6.2.2 Final considerations

In this chapter, the first one of the quantum models of computation, we introduced the **partitioned unitary quantum cellular automata**, which we believe to be a version that has a more

clear definition when we compared with the others presented in the begin of this chapter. The meaning of "more clear" here is in respect to the fact that with PUQCA we can easily extend it to higher dimensions and complicated geometries, as we will see in the next chapter. Moreover, in this new version, we are not supposed to check relations like Eq.(6.3) to see if we have a QCA well defined, as long as we work with its right definition. Perhaps within this new definition people will become more interested in the QCA computation model as a way to explore different problems. Besides, our PUQCA construction was such that the quantization of its classical counterpart, the partitioned CA, chapter 3, was straightforward. The simplicity of our model becomes clear when we translated the QBM to quantum cellular automata. Furthermore, we also could explore the quantum CA as an alternative model for a numerical method for partial differential equations. We saw how the PUQCA can, without difficulty, simulate a relativistic equation, the Weyl equation.

Until here the QCA seems to be a computational tool as powerful as its classical counterpart to explore quantum physics. In addition, we also expect this model to be a good candidate to explore complex systems as well as to study emergent phenomena like its classical counterpart, chapter 5. Furthermore, in the next chapter, we will show some results that suggest that the QCA is indeed a powerful computational tool, like the CA, for simulating physical processes.

CHAPTER 7

QUANTUM WALKS VIA QUANTUM CELLULAR AUTOMATA

In the last chapter we introduced the **partitioned quantum cellular automata** (PUQCA). We saw two examples of how this quantum model of computation can be useful to simulate quantum systems e.g. the quantum Brownian motion, which from now on we will call coined quantum walk in agreement with the community, and the Weyl equation, the relativistic equation to describe massless fermions. Until now we gave motivations to use QCA based on its simple structure in terms of local operators and its discrete formulation which might be handy in order to describe complex quantum systems. These are basically the same kind of motivations used to employ the PCA to describe classical systems. In the end, QCA provides a different picture to deal with the same problems described by different quantum models of computations, as the quantum circuit model [88] and quantum walks. Like, these other models, the QCA is a universal model of computation [57, 80] and thus choosing another model of computation will provide an equivalent simulation. However, there is another characteristic of the QCA that we did not mention until now, which can be the most important aspect about this computational model. The QCA structure matches quantum computers architectures employed into most quantum computers available now and the next to come [12, 31, 41].

When we see illustrations of how the qubits interactions are arranged into the D-wave [Fig.(7.1)] and IBM [Fig.(7.2)] quantum computers, which employ superconducting devices [12, 41, 69], it is quite clear from these illustrations that these device are closer to QCA than to circuit models, the most employed language in quantum information theory. The circuit model assumes that each qubit can interact with all others, even if they are distant, what is not possible (without a large overhead) in present implementations.

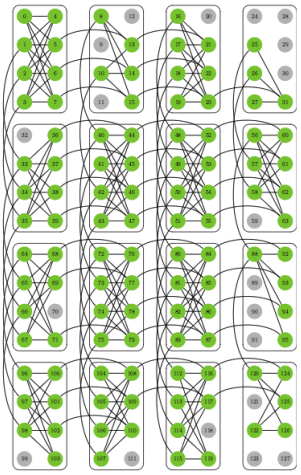


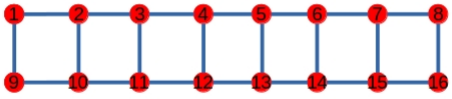
Figure 7.1: **Qubits and couplers in the D-Wave device.** The grey vertices represent non-working qubits. Figure from [12].

Alternatively, we can see in these illustrations that each qubit can only interact with few others, obeying some neighborhood scheme, in the same spirit of the cellular automata structure. In Fig.(7.1) we can think the interaction in each box achieved by the first tiling, following the PUQCA definition, and the interactions between these boxes achieved by the second tiling. In the illustration Fig.(7.2-b) of the IBM 16-qubit[1] quantum computer, we show how we can think this device in terms of a 3-tiling problem.

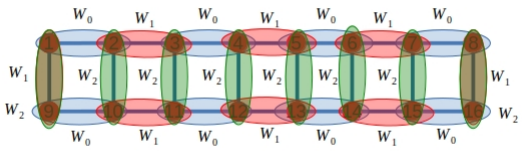
Despite the fact that in the last chapter we only applied the PUQCA to one-dimensional problems, its structure based in the tiling formalism provides a powerful tool to deal with more complicated geometries, as the ones showed in figures (7.1) and (7.2). Therefore if the community starts to see some interesting problems in terms of QCA and if they were concern about the real interaction between the qubits, they can start to describe their problems in terms of QCA instead of other models. On the other hand, even the people that help to develop the chips structures can become motivated to put the qubit interaction in such a way that they respect exactly the QCA formalism of interaction. Moreover, we only need a handful of a different unitaries.

Given these previous motivations, now we will show our new results on QCA. We expect that with the results that we will show in this chapter together with everything that we have seen about QCA model, people realize the power of this model of computation. Our main goal in this chapter is to start exploring the model of quantum cellular automata, as a suitable platform for quantum algorithms. We do so by showing that various models of quantum walks can be translated to a quantum cellular automata dynamics using the same amount of resources (effective Hilbert space dimension). More concretely we show how to implement two models of QWs – namely, the coined and the staggered QW with Hamiltonians, which are known to encompass a large class of QW models [61] – within the partitioned QCA model

[1] https://www.research.ibm.com/ibm-q/



a)



b)

Figure 7.2: **Qubits interaction via waveguide resonator in the IBM 16 qubits device.** In figure (a) we illustrated the interaction between the 16 qubits in the chip of the IBM quantum computer. Although the interaction between the qubits are via coplanar waveguide resonator [69], and thus the interaction between them can happen in distance scale of μm, which is relatively long, since we are dealing with interaction under quantum regime, the interaction of each qubit is restricted to the only three nearest qubits, except the ones located in the left and right boundaries. In figure (b) we show how the PUQCA can easily, using three tilings, incorporate this structure of interaction.

7.1 CQW$_d$ ⊆ QCA

In chapter 6 we saw how we can translate the coined quantum walk to the partitioned quantum cellular automata, Def.(6.3). We showed how this translation works in a one-dimensional lattice and we employed the ordinary quantum walk definition for a line, [42]. This translation was our starting point to construct a general recipe to move from the coined quantum walk model to QCA. In our general translation we are not restricted to only a one-dimensional lattice, we can translate coined quantum walk over a d-regular graph (CQW$_d$) to PUQCA. In order to achieve this goal we move to a more sophisticated (CQW$_d$) definition [58] to cope with general dynamics in arbitrary finite graphs, which we will present now.

Given a graph $G = G(V,E)$ its vertices are associated to the walker's classical positions, while its edges define the possible directions the walker can take in a single step. To the position $i \in V$ and direction $(i,j) \in E$ we associate the state $|(i,j),i\rangle$. Defining $\mathcal{N}_i^G$ as the graph-neighborhood of a vertex i, $\mathcal{N}_i^G = \{j|(i,j) \in E\}$, the total Hilbert space associated to the quantum walk is given by

$$\mathcal{H}_G = \text{span}(\{|(i,j),i\rangle\,|(i,j) \in E,\ i \in V,\ j \in \mathcal{N}_i^G\}).$$

As each edge is connected to exactly two vertices, the dimension of $\mathscr{H}_G$ is $2|E|$. We should stress the fact that we are not in the case where the Hilbert space $\mathscr{H}_G$ is a tensor product of a space associated to the edges and another space associated to the vertices, like we did in chapter 6.

Despite the fact that we are employing a different definition for the CQW than we used in the last chapter, its dynamics is established in the same manner: first the "flip" of a quantum coin, by the coin operator C, and then a coin-dependent coherent displacement, by the shift operator S. As before, given a walker in the state $|(i,j),i\rangle$, the coin operator must create a superposition of all allowed directions, i.e., a superposition of the states $|(i,k),i\rangle$ for all $k \in \mathscr{N}_i^G$. The action of the coin operator $C : \mathscr{H}_G \to \mathscr{H}_G$ for this current definition can be decomposed in blocks for each vertex. So instead of writing the global coin operator acting in $\mathscr{H}_G$, we can write the correspondent operator for each vertex space, $\mathscr{H}_G^i$ with

$$\mathscr{H}_G^{(i)} = \text{span}\left(\{|(i,j),i\rangle\}|\,(i,j) \in E, j \in \mathscr{N}_i^G\right),$$

and $\mathscr{H}_G = \oplus_{i\in V}\mathscr{H}_G^{(i)}$. Thus, for a d-regular graph, each such block is a d-dimensional Hilbert space. Then, employing this decomposition lead us to

$$C = \bigoplus_{i\in V} C_i, \tag{7.1}$$

where each C_i is a unitary acting on $\mathscr{H}_G^{(i)}$. In the cases where the coin flip does not depend of the vertex, what happens in most cases of interest, each coin operator C_i, $i \in V$, is identical. Now, in order to finish the walker step we must apply the shift operator $S_\pi : \mathscr{H}_G \to \mathscr{H}_G$, which is again a global operator and it is not vertex independent. Since the walker can only walk "through" the edges, the action of the most general shift operator is given by:

$$S_\pi\,|(i,j),i\rangle = |(\pi(i),j),j\rangle\,, \forall i \in V \text{and} \forall\,(i,j) \in E, \tag{7.2}$$

where $\pi(i) \in \mathscr{N}_j^G$. π is a permutation function between all possible edges from a given vertice. Thus, it fixes the walker's direction after the step. For a d-regular graph it implies that there are only $d!$ different shift operators. Furthermore, since the π function is vertex independent, this part can be absorbed into the coin operator as long as we change the initial and final states, like we showed in the last chapter. Thus, from a given shift operator, we always can choose π as the identity operator and keep with the same walker dynamics as long as we translate the content of π for the coin operator. Fixing π as the identity permutation the shift operator takes the following form

$$S_I\,|(i,j),i\rangle = |(i,j),j\rangle\,,$$

which is the flip-flop operator, Eq.(6.22), in this new definition, given that $S_I^2 = \mathbb{1}$. In summary, since the walker evolution is composed by these two operators, coin and shift, the one time step of the quantum walker is given by the unitary $U_G : \mathscr{H}_G \to \mathscr{H}_G$, whose action can be written as:

$$U_G = S_I \cdot C. \tag{7.3}$$

Now with the same spirit of chapter 6 we will translate a general one-dimension coined quantum walk, within this definition, to the partitioned unitary quantum cellular automata. The goal of this example is to see how to construct a general recipe, starting from the simplest case.

7.1.1 One dimensional example

This example that we will show here is the same one presented in chapter 6, the one that gives the quantum Brownian motion, translated to this CQW definition showed above.

As we saw above, this current CQW employs a graph perspective, thus the first thing that we have to do is translate the lattice perspective to the graph one. Since we worked within an infinite one-dimensional lattice, here we move to a infinite 2-regular graph. In such a graph perspective, the set of vertices is $V = \mathbb{Z}$, and the set of edges is $E = \{(i, i+1) | i \in V\}$ and, as before, the neighborhood, that here we call by vertex-neighborhood, is $\mathcal{N}_i^G = \{i-1, i+1\}$. We associate the state $|(i, i+1), i\rangle$ if the walker is on the vertex i moving to the right and $|(i, i-1), i\rangle$ if the walker is on the vertex i moving to the left, respectively. While in the example of chapter 6 the coin superposes the spin states, that determines the walker direction, the coin here superposes these two directions of movement $|(i, i+1), i\rangle$ and $|(i, i-1), i\rangle$. Thus, similarly to the the coin used in Eq.(6.12), this can be done by choosing $C_i : \mathscr{H}_G^{(i)} \to \mathscr{H}_G^{(i)}$ as a SU(2) operator

$$C_i = \begin{pmatrix} p & q \\ q & p \end{pmatrix} \forall i \in V,$$

with p, $q \in \mathscr{C}$ respecting the unitarity constraints $|p|^2 + |q|^2 = 1$ and $p^* q + q^* p = 0$. The total coin operator is then $C = \bigoplus_{i \in V} C_i$. Like we did in the last chapter we will use the "moving" as the shift operator. This operator acts on $\mathscr{H}_G$ as,

$$\begin{aligned} S_X |(i, i-1), i\rangle &= |(i-2, i-1), i-1\rangle, \\ S_X |(i, i+1), i\rangle &= |(i+2, i+1), i+1\rangle. \end{aligned}$$

The relation between the moving shift S_X to the flip-flop S_I one is simply: $S_X = X \cdot S_I$, where $X = \bigoplus_{i \in V} X_i$ with

$$X_i = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \forall i \in V.$$

A single step evolution is then given by $U = X \cdot S_I \cdot C$. As we said before, the last operator X can be absorbed into the coin operator. Now, in order to get the recurrence relation for this example, we will follow the same steps that we did in the previous chapter, which are to write the state of the system at time t,

$$|\psi(t)\rangle = \sum_{i \in V} \left(\psi_{(i,i-1)}(i,t) |(i, i-1), i\rangle + \psi_{(i,i+1)}(i,t) |(i, i+1), i\rangle \right), \tag{7.4}$$

where $\psi_{(i,j)}(i,t) := \langle (i,j), i \, | \psi(t) \rangle$ is the amplitude of the walker being located at the vertex i pointing to the vertex j at time t, and compute $U |\psi(t)\rangle$. After we do this last calculation we can

easily write the recurrence relations that govern the quantum walk dynamics

$$\psi_{(i-2,i-1)}(i-1,t+1) = p\psi_{(i,i-1)}(i,t)+q\psi_{(i,i+1)}(i,t), \tag{7.5}$$

$$\psi_{(i+2,i+1)}(i+1,t+1) = q\psi_{(i,i-1)}(i,t)+q\psi_{(i,i+1)}(i,t), \tag{7.6}$$

which is exactly the same achieved in Eq.(6.14) as we expected, since both have the same coin and shift operators. The difference arises only from the way that we write these operators. While in chapter 6 we used the ordinary CQW definition, in terms of a tensor product between the coin and the particle position, here we used a global Hilbert space associated with the graph G.

Now as we did in the previous chapter let us translate this CQW dynamics into a PUQCA. As we pointed above we are dealing with the same dynamics achieved for the QBM problem. Thus, the translation will be exactly the same showed in (6.2.1.2). Nevertheless, we will show this translation step by step, since we are dealing with another QW definition. In order to do that, each vertex $i \in V$ we change to a cellular automaton cell and as before each cell is divided into two subcells since each vertex has two neighbors. Here, instead at labeling these subcells by i_0 and i_1, we will label them by i_{i-1} and i_{i+1}, once we want use the graph properties into the QCA translation and to establish a general procedure in the end. Again in each subcell we place a two-dimensional quantum system (a qubit)

$$\mathscr{H}_{(i)_{(i-1)}} \cong \mathscr{H}_{(i)_{(i+1)}} \cong \mathbb{C}^2.$$

One main difference from this translation to the one showed in chapter 6 is in the way that we encode the quantum walk state into the automaton state. For the quantum walk state $|(i,i-1),i\rangle$ the automaton state is $|\ldots,(0,0)_{i-1},(1_{i-1},0_{i+1})_i,(0,0)_{i+1},\ldots\rangle$ and for the quantum walk state $|(i,i+1),i\rangle$ the automaton state is $|\ldots,(0,0)_{i-1},(0_{i-1},1_{i+1})_i,(0,0)_{i+1},\ldots\rangle$. The excited subcell has the same meaning showed before and indicates the walker position and its movement direction. All the encoding is done within the single excitation subspace.

Moving forward we need to show the set of unitary operators in the QCA formalism that we need to employ in order to achieve the QW dynamics established in Eq.(7.5). Indeed, we have to employ the same unitary operators used in the last chapter, which are:

1. $$W_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & p & q & 0 \\ 0 & q & p & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

   that plays the role of the coin C that also fulfilling the requirements that $|p|^2+|q|^2=1$ and $p^*q+q^*p=0$. As before this operator is associate with the first tiling $\mathscr{T}_0$ where now the tiles are written as follows $T_i^{(0)} = \{(i)_{i-1},(i)_{i+1}\}$, and thus $W_0 : \mathscr{H}_{\Sigma_{(i,i-1)}} \otimes \mathscr{H}_{\Sigma_{(i,i+1)}} \rightarrow \mathscr{H}_{\Sigma_{(i,i-1)}} \otimes \mathscr{H}_{\Sigma_{(i,i+1)}}$, and it is written in the basis $\{|(0,0)_i\rangle, |(0,1)_i\rangle, |(1,0)_i\rangle, |(1,1)_i\rangle\}$.

2.

$$W_1 = SWAP$$

that plays the role of the flip-flop shift operator S_I. Again this operator, that is associated with the second tiling $\mathscr{T}^1$, is responsible for interaction between subcells of neighbors cells in the QCA picture $\mathscr{N} = \{j-1, j, j+1\}$. The tiles here are written as $T_i^{(1)} = \{(i)_{i+1}, (i+1)_i\}$.

3.

$$W_2 = SWAP$$

that plays the role of the X operator, the action that will encode the moving shift operator. As we know this operator acts inside of each cell. Then the tiles in $\mathscr{T}^2$ are given by $\{(i)_{i-1}, (i)_{i+1}\}$.

Now in order to show that the three operators above mimic the QW dynamics Eq.(7.5) we need first to write a general state $|\Psi(t)\rangle$ for the automaton at time t

$$\sum_{i\in\mathbb{Z}} \begin{array}{l} \big[\Psi_{(i,i-1)}(i,t)\big|\ldots,(0,0)_{i-1},(1_{i-1},0_{i+1})_i,(0,0)_{i+1},\ldots\big\rangle + \\ \quad + \Psi_{(i,i+1)}(i,t)\big|\ldots,(0,0)_{i-1},(0_{i-1},1_{i+1})_i,(0,0)_{i+1},\ldots\big\rangle\big], \end{array}$$

where $\Psi_{(i,i\pm1)}(i,t)$ is the probability amplitude of finding an "excitation" at subcell $i\pm1$ of the i-th cell at time t (note that this initial state is the same of Eq.(7.4) encoded in the QCA), and then apply the transition function

$$\mathscr{E} = \left(\bigotimes_{T_j^{(2)}\in\mathscr{T}_2} W_2\right)\left(\bigotimes_{T_j^{(1)}\in\mathscr{T}_1} W_1\right)\left(\bigotimes_{T_j^{(0)}\in\mathscr{T}_0} W_0\right)$$

using the operators showed above. Doing this calculation, which is exactly the same done in the last chapter, we obtain the recurrence relations that describe the automaton dynamics:

$$\begin{aligned} \Psi_{(i-2,i-1)}(i-1,t+1) &= \quad p\Psi_{(i,i-1)}(i,t) + q\Psi_{(i,i+1)}(i,t), \\ \Psi_{(i+2,i+1)}(i+1,t+1) &= \quad q\Psi_{(i,i-1)}(i,t) + p\Psi_{(i,i+1)}(i,t). \end{aligned}$$

Then, we set the same recurrence relations for the 1-d coined quantum walk, as we expected.

7.1.2 General Recipe

Now we give a prescription to find the QCA correspondent to a given coined QW on a d-regular graph. As input we take a d-regular graph $G = G(E,V)$ (or lattice L) where the quantum walk is defined; the coin operator C; and the shift operator S_π. As output we must return a complete PUQCA whose evolution is the same as the CQW. The steps to find this translation are enumerated below.

1. The number of cells in the automaton is given the number of vertices, $|V|$, of graph G. As the graph is d-regular, each cell is split in d subcells. We place one qubit in each subcell, and then a total of $|V|.d$ qubits are employed (see the resources discussion below).

2. The automaton neighborhood scheme $\mathscr{N}$ is determined by the graph-neighborhood $\mathscr{N}^G$, by the simple inclusion of the "central" cell: $\mathscr{N}_i = \mathscr{N}_i^G \cup \{i\}$.

3. To each CQW$_d$ state $|(i,j),i\rangle$ we associate the automaton (single excitation subspace) state $\left|\ldots(0_h,\ldots,1_j,\ldots,0_m)_i\ldots\right\rangle$, where h,m and all other subindex labeling the subcells of cell i belong to $\mathscr{N}_i^G$. While the subindex i gives us in which cell the excitation is located, the subindex j tells us its subcell location (corresponding to the movement direction). In this way the space associated to each cell is $\mathscr{H}_\Xi = \left(\mathbb{C}^2\right)^{\otimes d}$.

4. To simulate the CQW$_d$ dynamics within the automata language three tilings are sufficient [2]. The first tiling is related to the action of the coin, with each tile given by all subcells that belongs to the same cell: $T_i^{(0)} = \{(i)_j | j \in \mathscr{N}_i^G\}$. The second tiling is devoted to the simulation of the flip-flop shift acting on neighboring cells. From the definition of S_I we can see that in terms of QCA we have the subcell j in the cell i interacting with the subcell i of the j-th cell, and vice-versa. Therefore $T_{(i,j)}^{(1)} = \{(i)_j,(j)_i\}$ where $(i,j) \in E$. The third tiling is responsible for simulating the permutation that connects S_I to S_π. As this operation is "local" in each vertex, then $T_i^{(1)} = \{(i)_j | j \in \mathscr{N}_i^G\}$.

5. To each tiling we associate one unitary operator. To the first tiling, the unitary operator $W_0 : \mathscr{H}_\Xi \to \mathscr{H}_\Xi$ is directly related to the unitary operator C_i by employing the unary representation for the cell states (see two steps above) within the single excitation subspace. Out of this subspace the action of W_0 is trivial, being completed by ones in the diagonal entries. Since the flip-flop operator is translated as a swap between the two subcells of neighboring cells, the unitary $W_1 : \mathscr{H}_{(i)_j} \otimes \mathscr{H}_{(j)_i} \to \mathscr{H}_{(i)_j} \otimes \mathscr{H}_{(j)_i}$, for $(i,j) \in E$, is the SWAP gate between them. Lastly, $W_2 : \mathscr{H}_\Xi \to \mathscr{H}_\Xi$ implements the permutation π on cell i, by encoding the operator π_i in the same way as for the coin operator.

These steps give the full translation between a CQW$_d$ and a QCA.

Before we move on, we compare the resources required in each model. The dimension of the Hilbert space in the quantum walk model is $2|E|$, which is equal $|V|d$ due to the assumed d-regularity of the graph. For the corresponding QCA at first sight we would need a Hilbert space of dimension $|V|.2^d$. Nevertheless, as we only need the single excitation subspace for our construction, whose dimension is d, we in fact only use an effective Hilbert space with dimension $|V|.d$. Therefore, both models require the same amount of resources.

[2] Here again the action of the third tiling can be absorbed in the action of the first one, plus modifications in the initial and final state. We however present the translation with three tilings for clarity reasons.

7.2 SQW ⊆ QCA

We start by describing the staggered quantum walk with Hamiltonians (SQWH) over a graph $G = G(V,E)$ [60]. As before, to each vertex $i \in V$ we associate a unit vector $|i\rangle$, with $\langle i|j\rangle = \delta_{ij}$ for all $i,j \in V$. As such, to the vertices of G we associate the Hilbert space $\mathscr{H}_V = \text{span}(\{|i\rangle \mid i \in V\})$. Crucial to the SQWH is the concept of a graph *tessellation*: A graph tessellation $\Im$ is a partition of the graph into complete subgraphs, i.e., into cliques. Such a partition directly induces a partition of $\mathscr{H}_V$: let α be an element of $\Im$, then $\mathscr{H}_V = \bigoplus_{\alpha\in\Im}\mathscr{H}_\alpha$, where $\mathscr{H}_\alpha = \text{span}(\{|i\rangle \mid i \in \alpha\})$. Each element α is called a polygon, as it is related to a clique. It is now simple to define a rank-one projector $|\alpha\rangle\langle\alpha|$ into $\mathscr{H}_\alpha$, by defining the vector

$$|\alpha\rangle = \sum_{i\in\alpha} a(i)|i\rangle,$$

where $a(i) \in \mathbb{C}$ and $\sum_{i\in\alpha}|a(i)|^2 = 1$. Note that the coefficients a do not depend on the polygon in a given tessellation, but they do depend on the label given to each vertex within a polygon [61]. A dynamics that does not connect different subspaces $\mathscr{H}_\alpha$ can be given by the Hamiltonian operator associated with this tessellation as

$$H_\Im = 2\sum_{\alpha\in\Im}|\alpha\rangle\langle\alpha| - \mathbb{1}. \tag{7.7}$$

If we want to check that $H_\Im$ does not connect different subspaces we just need to take two vertices from different polygons in the same tessellation $\Im$, for instance $i \in \alpha_1$ and $j \in \alpha_2$, and check that $\langle i|H_\Im|j\rangle = 0$, which is a straightforward calculation as we can see below

$$\begin{aligned}
\langle i|H_\Im|j\rangle &= \left\langle i\left|2\sum_{\alpha\in\Im}|\alpha\rangle\langle\alpha| - I\right|j\right\rangle, \\
&= 2\sum_{\alpha\in\Im}\langle i|\alpha\rangle\langle\alpha|j\rangle - \delta_{ij}, \\
&= 2(\langle i|\alpha_1\rangle\langle\alpha_1|j\rangle + \langle i|\alpha_2\rangle\langle\alpha_2|j\rangle), \\
&= 0.
\end{aligned}$$

Such operator is known as the orthogonal reflection of the graph [60], and it is Hermitian and unitary, implying that $H_\Im^2 = \mathbb{1}$. The dynamics generated by this Hamiltonian is then $U_\Im = \exp(i\theta H_\Im)$, with $\theta \in [0, 2\pi]$. This propagator respects the partition of $\mathscr{H}_V$ into subspaces related to the tessellation polygons, as $U_\Im = \bigoplus_{\alpha\in\Im}U_\alpha$ with

$$U_\alpha = e^{-i\theta}\mathbb{1}_\alpha + 2i\sin(\theta)\sum_{i,j\in\alpha}a^*(i)a(j)|i\rangle\langle j|; \tag{7.8}$$

where $\mathbb{1}_\alpha := \sum_{i\in\alpha}|i\rangle\langle i|$ is the identity operator in the α subspace.

If the dynamics of the walker was to be given simply by the propagator $U_\Im$, then a walker starting in a vertex $i \in V$ would remain stuck in the polygon that contains such a vertex. As Szegedy noticed [72], a walker dispersion over a graph can be obtained without a coin if we

alternate propagation operators, with each of them acting within a different subspace-partition of $\mathscr{H}_V$. At this point we observe that a given tessellation contains all the vertices of a graph, but not necessarily all its edges. In [61] it was defined a set of tessellations, a *tessellation cover* $\{\Im_0,...,\Im_{N-1}\}$, whose union also covers the edge set. Each tessellation $\Im_k$ induces a different subspace-partition of $\mathscr{H}_V$, with associated Hamiltonian $H_{\Im_k}$ constructed in the same way as in Eq.(7.7). One time-step evolution of the SQWH is then generated by the operator

$$U = \prod_{k=0}^{N-1} e^{i\theta_k H_{\Im_k}} \tag{7.9}$$

with $\theta_k \in [0,2\pi]$ for all $k \in \{0,\ldots,N-1\}$. We are now ready to show how to translate the SQWH

model into a QCA one. As previously, before giving a general recipe we first show a simple example of such a translation.

7.2.1 One dimensional example

In this example, we consider a SQWH over a 1-d lattice. The vertex set is $V = \mathbb{Z}$, and thus $\mathscr{H}_V = \text{span}(\{|i\rangle \mid i \in \mathbb{Z}\})$. The smallest tessellation cover for such 2-regular (infinite) graph is composed of two tessellations $\{\Im_0, \Im_1\}$, with $\Im_0 = \{$  $\mid i \in \mathbb{Z}\}$ and $\Im_1 = \{$  $\mid i \in \mathbb{Z}\}$. These tessellations are shown Fig.(7.3). Each tessellation induces a different partition of $\mathscr{H}_V$ as $\bigoplus_{\alpha_k \in \Im_k} \mathscr{H}_{\alpha_k}$, with $k \in \{0,1\}$. For this example we take general projectors into each polygon-subspace via the vectors

$$\left|\text{}\right\rangle = a_0 |2i\rangle + \tilde{a}_0 |2i+1\rangle,$$
$$\left|\text{}\right\rangle = a_1 |2i+1\rangle + \tilde{a}_1 |2i+2\rangle,$$

for all $i \in \mathbb{Z}$, and where the coefficients are constrained to $|a_k|^2 + |\tilde{a}_k|^2 = 1$ with $k \in \{0,1\}$. With these projectors we follow Eq.(7.7) to construct the evolution operator as

$$U = e^{i\theta_1 H_{\Im_1}} e^{i\theta_0 H_{\Im_0}}.$$

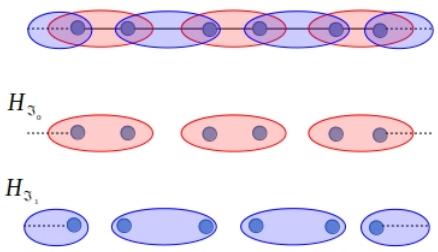


Figure 7.3: This picture shows the two tessellations used in the one-dimensional example of translation between the SQWH and the PUQCA. Within the SQWH the operator U_0 (U_1) acts on the red (blue) polygons. In the PUQCA picture, this is translated to the action of W_0 (W_1) on the qubits in the red (blue) tiles. Figure from [22].

As the evolution propagator is composed by the product of similar operators, $e^{i\theta_k H_{\Im_k}}$, acting in similar ways in different partitions of $\mathscr{H}_V$, below we only show how to translate a single of these operators, say $e^{i\theta_0 H_{\Im_0}}$, into automata language. Still within the SQWH language, with the aid of Eq.(7.8), the propagator in each subspace is

$$\begin{aligned} U_{\overset{2i}{\bullet}\!\!-\!\!\!-\!\!\overset{2i+1}{\bullet}} &= e^{-i\theta_0}\left(|2i\rangle\langle 2i| + |2i+1\rangle\langle 2i+1|\right) + 2i\sin(\theta_0)\left(|a_0|^2|2i\rangle\langle 2i|\right. \\ &+ \left. a_0^*\tilde{a}_0|2i\rangle\langle 2i+1| + a_0\tilde{a}_0^*|2i+1\rangle\langle 2i| + |\tilde{a}_0|^2|2i+1\rangle\langle 2i+1|\right). \end{aligned}$$

The evolution operator for this tessellation is then $U_{\Im_0} = \sum_{i\in\mathbb{Z}} U_{\overset{2i}{\bullet}\!\!-\!\!\!-\!\!\overset{2i+1}{\bullet}}$. Let a general walker state at time t be expressed as $|\psi(t)\rangle = \sum_{i\in\mathbb{Z}}\psi(i,t)|i\rangle$, where $\psi(i,t)$ is the probability amplitude for vertex i at time t. The state after the action of $U_{\Im_0}$ is then given by

$$\begin{aligned} U_{\Im_0}|\psi(t)\rangle &= \sum_{i\in\mathbb{Z}}\left\{\left[e^{-i\theta_0} + 2i\sin(\theta_0)|a_0|^2\psi(2i,t) + 2i\sin(\theta_0)a_0\tilde{a}_0^*\psi(2i+1,t)\right]|2i\rangle\right. \\ &+ \left.\left[e^{-i\theta_0} + 2i\sin(\theta_0)|\tilde{a}_0|^2\psi(2i+1,t) + 2i\sin(\theta_0)\tilde{a}_0a_0^*\psi(2i,t)\right]|2i+1\rangle\right\}. \end{aligned}$$

This is the evolution that we want to simulate within the quantum cellular automata model.

For the QCA simulation, in each lattice vertex we place one qubit. The two tessellations needed for the walker dynamics will now give us two tilings. The polygons of the first tessellation determine now the tiles of the first tiling: $T_i^{(0)} = \{2i, 2i+1\}$. Similarly, for the second tiling the correspondence implies the tiles $T_i^{(1)} = \{2i+1, 2i+2\}$. There are two possibilities to define the cellular automaton cell: first is to take the tiles of the first tiling as forming a single cell with two subcells; second is to consider each vertex as a single cell with no subcell division. Both cases are equivalent, and we take the second choice as it simplifies the simulation description. The encoding of the walker state into the QCA framework is then simply given by $|i\rangle \to |\ldots, 0_{i-1}, 1_i, 0_{i+1}, \ldots\rangle$,

for all $i \in V$. A general state for the automaton at time t is then written as

$$|\Psi(t)\rangle = \sum_{i\in\mathbb{Z}} \Psi(i,t)|\ldots,0_{i-1},1_i,0_{i+1},\ldots\rangle,$$

with $\Psi(i,t)$ the amplitude of finding on "excitation" at the i-th qubit at time t. Now we need to simulate the evolution operator $U_{\Im_0}$ with the action of the first tiling. To that we note that each polygon in $\Im_0$ corresponds exactly to a tile in $\mathscr{T}_0$. Therefore, the propagator $U_{\Im_0} = \bigoplus_{\alpha_0\in\Im_0}$ is then translated into $\bigotimes_{T_i^{(0)}\in\mathscr{T}_0} W_0$, where

$$W_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & e^{-i\theta_0} + 2i\sin(\theta_0)|a_0|^2 & 2i\sin(\theta_0)a_0\tilde{a}_0^* & 0 \\ 0 & 2i\sin(\theta_0)a_0^*\tilde{a}_0 & e^{-i\theta_0} + 2i\sin(\theta_0)|\tilde{a}_0|^2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

when written in the computational basis $\{|0_{2i},0_{2i+1}\rangle, |0_{2i},1_{2i+1}\rangle, |1_{2i},0_{2i+1}\rangle, |1_{2i},1_{2i+1}\rangle\}$. With such a encoding, the evolution given by $\bigotimes_{T_i^{(0)}\in\mathscr{T}_0} W_0$ leads a general state of the automaton at time t to the state:

$$\sum_{i\in\mathbb{Z}} \Big\{ \Big[e^{-i\theta_0} + 2i\sin(\theta_0)|a_0|^2\Psi(2i,t) + 2i\sin(\theta_0)a_0\tilde{a}_0^*\Psi(2i+1,t) \Big] |\ldots,0_{2i-1},1_{2i},0_{2i+1},0_{2i+2},\ldots\rangle$$

$$+ \Big[e^{-i\theta_0} + 2i\sin(\theta_0)|\tilde{a}_0|^2\Psi(2i+1,t) + 2i\sin(\theta_0)\tilde{a}_0a_0^*\Psi(2i,t) \Big] |\ldots,0_{2i-1},0_{2i},1_{2i+1},0_{2i+2}\ldots\rangle \Big\}.$$

After decoding, this state is exactly equivalent to the SQWH state shown in Eq.(7.10).

This example shows that the tessellations in the SQWH play the role of the tilings in the QCA, and the set of operators $\{W_i\}$ in the QCA are obtained from the polygon-subspace operators associated with the tessellations of the SQWH.

7.2.2 General Recipe

The general procedure to translate a SQWH into a PUQCA takes as input a d-regular graph $G = G(E,V)$ (or lattice L), a tessellation cover $\{\Im_k\}_{k=0}^{N-1}$ with polygons α_k for each tessellation, and a set of coefficients $\{a_k(i)\}_{i=0}^{|\alpha_k|-1}$. As result the procedure outputs a well-formed PUQCA by following the subsequent recipe.

1. Starting from SQWH on a d-regular graph $G(V,E)$, we use the same graph for the QCA.
2. Since the graph is the same, the neighborhood scheme for the corresponding automaton is also the same: $\mathscr{N}_j = \{v_i | (v_j,v_i) \in E\}$.
3. To each state $|v_j\rangle$ of the computation basis of the SQWH we associate the automaton state $|\ldots,0_{j-1},1_j,0_{j+1},\ldots\rangle$ (single excitation subspace).
4. For each tessellation of the SQWH we have an equivalent tiling in the QCA. That is, the vertices that belongs to the polygon $\tau_k^{(i)}$ of tessellation i yield the elements in the tile $T_k^{(i)}$.

5. For each local unitary operator $\exp(i\theta_i H_i)$ of the SQWH, we build an equivalent unitary operator W_i, that acts on tile $T_j^{(i)}$. To finish the description of this step, we need to obtain the block matrices associated with the polygons of a tessellation. From the Hamiltonian structure (7.7) of the SQWH, the entries of these blocks are given by $O_{l,m} = \langle v_l | \exp(i\theta_i H_i) | v_m \rangle$, where $v_l, v_m \in \tau_k^{(i)}$. The unitary operator W_i is built from O employing the unary representation.

This completes the translation from the SQWH to the PUQCA model. As we have done before, let us analyze the resources required by the PUQCA formalism for a given a SQWH dynamics. For a graph $G = G(E,V)$, the Hilbert space associated with the SQWH is a $|V|$-dimensional one. For the PUQCA we employed $|V|$ qubits, yielding a $2^{|V|}$-dimensional total space. However, here again, we only use the single-excitation subspace, which is $|V|$-dimensional. Once more, the effective amount of resources required by the PUQCA are the same as the original model.

7.3 Final considerations

With this chapter we finish our discussion on the topic about quantum cellular automata. In chapter 6 we proposed a new QCA definition that we believe to be a more clear definition than the previous ones. In the previous chapter we started to see the potentialities of this model of computation, and we tried to convince the reader that this model deserves the same attention as its classical counterpart. In the current chapter we continued to give motivations for this model, but exploring other aspects of the QCA formalism. We saw that the technologies employ hardware structures where the qubit interactions match the local aspect of the interaction employed by the QCA. After these arguments and claims about the potentialities of the QCA we showed our main results using the partitioned unitary cellular automata. We explained how we can translate the main QWs models, coined quantum walk and Staggered quantum walk with Hamiltonians, into a PUQCA dynamics. Since in [61] the authors showed how these two QWs models encompass the others flavors of QW, we are convicted that the PUQCA can cover all the others flavors, at the least when they are employed in d-regular graphs.

One question that is immediately raise by our translation results (QW→QCA) is whether quantum walks and quantum cellular automata are equivalent, i.e., if there is for every QCA an correspondent QW (QW←QCA). Of course this depends on how strict the definitions of the models are. For instance, it is usually accepted that in a coined quantum walk, the shift operator does not create superposition of the walker's position states, it is just a permutation operator. When translating CWQ into PUQCA, this implied that all the interactions between cells were simply SWAP gates. Therefore, if a given PUQCA has an interaction between cells other than a SWAP gate, our results suggest that there is no equivalent coined quantum walk to such a PUQCA.

Since that there are no doubts about the power of the QWs to simulate physics, from quantum [13] to relativistic theories, [3, 26, 50], and as a platform to develop quantum algorithms, in

particular quantum search algorithms [59], with the results presented here we can conclude that the QCA model is at least as powerful as QWs, with the advantage of bring more experimentally friendly. Therefore, we hope that our results will serve as the catalyst for the development of a whole new phenomenology of simulation of quantum systems.

CHAPTER 8

QUANTUM HPP

Through out this thesis we saw different problems in physics that can be handled with cellular automata. In particular in chapter 2 we saw how we can apply the partitioned cellular automata (PCA) to simulate particles that can collide, a model known as HPP [37]. Although the HPP rule is quite simple, it was an extremely important model, since it was the first model applied in the area of *molecular dynamics* and it gave inspiration to more sophisticated models, as the FHP [32], a discrete model for the Navier Stokes equation. Furthermore, the HPP rule captures the main aspects of the interactions of the particles, namely the number of particle and momentum conservation during the collision, as we showed in chapter 2.

Working with the quantum counterpart of the PCA, we become motivated to generalize the HPP's to collision rule to the partitioned unitary quantum cellular automata (PUQCA). The idea was to consider quantum phenomena as superposition and interference for the quantum version of the HPP interaction. With this model we would be able to study molecular dynamics and thermodynamical properties from a quantum perspective. However, we are aware of how difficult it is to work with many quantum particles using classical computers, since the Hilbert space dimension increases exponentially with the number of particles. Thus, even the description as three particles in one-dimensional lattice is already a hard problem for our ordinary computers. But we know that what we called particles in the HPP formulation are just excitations in the QCA perspectives, a fact that we can take advantage in the quantum HPP version. But again, a naive PUQCA implementation in a 5×5 two-dimensional lattice requires 25 cells with 4 subcells each one, since each excitation has two degree of freedom. We learned in chapter 7 that each cell in this case demands four qubits. Thus, it would requires in the end 100 qubits, a Hilbert dimension equivalent to 2^{100}, which is out of reach for our classical computers. Despite the fact that are some tricks to handle many particles [40], we did not explore them so far and we took

another direction to explore this rule in a quantum version.

During our investigations of how to bring problems implemented by quantum walks (QW) to quantum cellular automata we became familiar with QW's structures and how to use them for physics, as we briefly showed in chapter 6. Moreover, during our incursions in this quantum model of computation we saw a few results where people used two walkers either with or without interaction between them for different proposals. For instance, in [67] they applied two interacting quantum walks to understand their dynamics on a percolation graph, where they only did numerical analysis for one-dimensional lattice with 80 points (nodes, vertices). Now in [10] they used two interacting and noninteracting walkers to propose a quantum algorithm to determine if two graphs are isomorphic (related to each other by a relabeling of the graph vertices) with up 64 vertices. Finally in [9], like in [10], they applied two quantum walkers to the graph isomorphism problem with up 40 vertices and they also studied the entanglement dynamics between the two walkers in a 1D lattice. After we had seen these examples and with our wish to propose a quantum version to the HPP. Problem we thought that two quantum walkers in a coined model with the HPP interaction could be a good starting point.

The goal of this chapter is to introduce a model of two quantum walkers that can collide following the HPP interaction proposed in [37]. Despite the fact that this rule is described in terms of PCA, we took advantage of our results showed in chapter 7 to bring the HPP to the QWs formalism, the reverse path showed in chapter 7. Differently from what we expected at first the quantum phenomenon in this model appears only when the particles are alone, undergoing independent QW dynamics. On the other hand the interaction between them does not create any superposition, following the HPP rule. Differently from the previous results, that we briefly commented here, initially we are not interested in apply these two walkers to the graph isomorphism problem. Our initial interests are to try to understand the dynamics generated by this new model and investigate the entanglement dynamics between the two walkers. But, we saw, so far, three possible applications that can motivate the study of this model: to use these two quantum walkers with the HPP interaction to try to get an advantage in search problems [59]; to study thermodynamic effects, defining quantum thermodynamical quantities like temperature, similarly as done in [64]; to apply the quantum HPP (QHPP) to the graph isomorphism problem, as the previous works.

8.1 Coined model

There are three main goals in this section, the first one is to review the coined quantum walk model, focusing it on a two-dimensional lattice, the second one is to extend this model for two quantum walkers and finally to show the mathematical rule for a two quantum walkers model following the HPP interaction.

8.1.1 CQW in L^2

Differently from what we did in chapter 7, we will work with the coined model quantum walk where the Hilbert space is a tensor product of two parts: one related with the state space of the particle and other with the coin-space,

$$\mathscr{H}_C \otimes \mathscr{H}_{L^2}. \tag{8.1}$$

For a two-dimensional lattice with a grid-length 1 the spatial part is spanned by basis states $\{|x,y\rangle : x,y \in \mathbb{Z}\}$ and if we work with a lattice size N, $\mathscr{H}_{L^2} = \{|x,y\rangle : x,y = 0,1\ldots N-1\}$. The coin space $\mathscr{H}_C$, spanned by the four possible directions on the two-dimensional lattice, namely left, right, up and down of the particle, has dimension 4 and its computational basis can be denoted by $\{|c_x,c_y\rangle : 0 \leq c_x,c_y \leq 1\}$. Each one of these states is related with the direction of the particle. For the x part we have

$$\begin{aligned} |0\rangle &\quad \rightarrow \\ |1\rangle &\quad \leftarrow \end{aligned}$$

and for the y part

$$\begin{aligned} |0\rangle &\quad \uparrow \\ |1\rangle &\quad \downarrow \end{aligned}$$

Then, composing these elements we get the following directions for the particle displacement

$$\begin{aligned} |00\rangle &\quad \nearrow \\ |01\rangle &\quad \searrow \\ |10\rangle &\quad \nwarrow \\ |11\rangle &\quad \swarrow \end{aligned} \tag{8.2}$$

These four coin states are interpreted here as the possible **momentum directions** of the particle. This momentum interpretation is extremely important for this model, as we will see.

The generic state of the walker at time t is described by

$$|\psi(t)\rangle = \sum_{c_x,c_y=0}^{1} \sum_{x,y=0}^{N-1} \psi^t_{c_x c_y}(x,y) |c_x c_y, xy\rangle,$$

where $\psi^t_{c_x c_y}(x,y)$ is the amplitude of the walker being located at the point x,y with the momentum direction given by c_x,c_y at time t, which obey the normalization condition

$$\sum_{c_x,c_y=0}^{1} \sum_{x,y=0}^{N-1} \left|\psi_{c_x,c_y}(x,y,t)\right|^2 = 1,$$

for all time t.

As we know, in this model we have a unitary V, that acts in the particle state at time t given as output the particle state at time $t+1$, $V : |\psi_t\rangle \rightarrow |\psi_{t+1}\rangle$, which is composed by two operators. The first one is the coin operator C that acts only in the coin space and the second one that acts in all space, called shift operator S. Thus, in the end we have the following form for our unitary

$$V = S \cdot (C \otimes I), \tag{8.3}$$

where I is the identity operator acting in $\mathscr{H}_{L^2}$. Under the action of a four dimensional generic coin in the base state $|c_x c_y, xy\rangle$, we have

$$C \otimes I \, |c_x c_y, xy\rangle = \frac{\sum_{c_x,c_y=0}^{1} \alpha_{c_x c_y} |c_x c_y, xy\rangle}{\sqrt{\sum_{c_x,c_y=0}^{1} |\alpha_{c_x c_y}|^2}},$$

where $\alpha_{c_x c_y} \in \mathbb{C}$.

In our model we will employ a shift operator that allows the walker only to move diagonally, see figure (8.1), respecting the possible momentum directions of the walker, as we showed in (8.2),

$$S \, |c_x c_y, xy\rangle = |c_x c_y, \left(x + (-1)^{c_x}\right), \left(y + (-1)^{c_y}\right)\rangle . \tag{8.4}$$

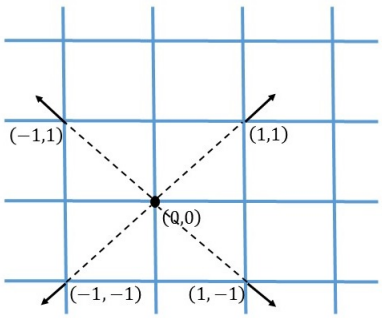


Figure 8.1: The four possible particle displacements coming from the four coins states. Then, starting with a particle at the point $(0,0)$ it does not have the option of going in a straight line. For instance, going to the point $(1,0)$ is not allowed from this initial position.

For instance, if $c_x = 0$ and $c_y = 0$, the values of x and y are incremented by one unit, which means that if the walker leaves position $(0,0)$, it will go to $(1,1)$, that is, it goes through the main diagonal of the lattice, figure (8.1).

8.1.2 Two quantum particles with HPP interaction

To add a new particle in this model first we need to increase our Hilbert space. If we have $\mathscr{H}_1$ for the first particle and $\mathscr{H}_2$ for the second one, our new Hilbert space is a tensor product of both

$$\mathscr{H}_1 \otimes \mathscr{H}_2, \tag{8.5}$$

where each one has the same format of (8.1). Thus, the generic state of the two walkers at time t is described by

$$|\psi(t)\rangle = \sum_{c_1,c_2}\sum_{l_1,l_2} \psi^t_{c_1c_2}(l_1,l_2)|c_1,l_1\rangle \otimes |c_2,l_2\rangle\,, \tag{8.6}$$

where $\psi^t_{c_1c_2}(l_1,l_2)$ is the amplitude of the walker one being located at the point $l_1 \in (x,y)$ with momentum direction given by $c_1 \in (c_x,c_y)$ and the walker two being located at the point $l_2 \in (x,y)$ with momentum direction given by $c_2 \in (c_x,c_y)$, both in time t. In case we do not have interaction between the particles, we have two operators that act independently, V_1 and V_2, where both are the same that we showed in Eq.(8.3). But here we propose a local interaction when these particles are in the same point on the lattice with opposite directions, following the HPP rule.

Like we saw in chapter 2, a natural choice for this collision is to change their momentum from p to $-p$, however, it is not a good choice here. Working with indistinguishable particles this rule is equivalent to the case where these particles cross themselves without interaction, thus we will keep with the same collision rule of HPP. We can see in Fig.(8.2) two examples, from the total of four, how the collision rule works under the perspective of diagonal displacement adopted here.

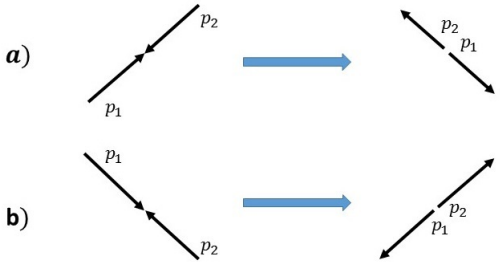
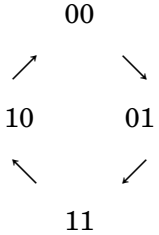


Figure 8.2: Collision rule

In the present work we took the convention that every time that we have two particles at the same point with opposite momentum the collision between them will change their direction by ninety degrees clockwise. For instance, if a particle has $|00\rangle$ as its coin state before the collision, then after the interaction, its coin state move to $|01\rangle$. The scheme below sums up this convention.



Now in order to get this interaction we present a new unitary operator V_{Int}. This operator, as the previous ones, is composed by two parts,

$$V_{\text{Int}} = (S_1 \otimes S_2) \cdot \mathscr{C}, \tag{8.7}$$

where $\mathscr{C}$ is the collision operator that acts in the total Hilbert space (8.5) and S_1 and S_2 are the shift operators that act independently in each particle. The collision operator has a trivial behavior in the particle position, always projecting to the same point. However, in the coin space there are four non trivial operations, which correspond to the four possible collisions. Therefore, the operator V_{Int} will act non trivially as long as both particles are at the same point, which means

$$U = \begin{cases} V_{\text{Int}} & \text{if } (xy)_1 = (xy)_2, \\ V_1 \otimes V_2 & \text{otherwise.} \end{cases} \tag{8.8}$$

Now we need to understand how the operator V_{Int} works, which ultimately means knowing the action of $\mathscr{C}$. Suppose now we have both particles at the same point, then, in this case the collision operator will only act non-trivially when the particles have opposite directions. In this case we apply our rule convention

$$\mathscr{C}\left|c_x c_y, xy\right\rangle \otimes \left|c'_x c'_y, xy\right\rangle = \begin{cases} \left|\bar{c'}_y c'_x, xy\right\rangle \otimes \left|\bar{c}_y c_x, xy\right\rangle & \text{if } c'_x c'_y = \bar{c}_x \bar{c_y}, \\ \left|c_x c_y, xy\right\rangle \otimes \left|c'_x c'_y, xy\right\rangle & \text{otherwise,} \end{cases} \tag{8.9}$$

where $\bar{c} = c \oplus 1$, sum mod 2. From the operation defined above, we can see that when we have two particles located in the same point of space the action of $\mathscr{C}$ does not give us any superposition. The superposition appears only when we have one particle.

We will finish this part summing up the total action of U, where from now on we will write this operator as U_{HPP}, since it is the unitary that implements the quantum version of the HPP rule,

$$U_{\text{HPP}}\left|c_x c_y, xy\right\rangle \otimes \left|c'_x c'_y, xy\right\rangle =$$

$$\begin{cases} \left|c_y c'_x, (x+(-1)^{c_y})\left(y+(-1)^{c'_x}\right)\right\rangle \left|\bar{c}_y c_x, \left(x+(-1)^{\bar{c}_y}\right)(y+(-1)^{c_x})\right\rangle & \text{if } (xy) = (x'y)' \text{ and } (c'_x c'_y) = (\bar{c}_x \bar{c_y}) \\ \left|c_x c_y, (x+(-1)^{c_x})(y+(-1)^{c_y})\right\rangle \left|c'_x c'_y, \left(x+(-1)^{c'_x}\right)\left(y+(-1)^{c'_y}\right)\right\rangle & \text{elif } (xy) = (x'y') \text{ and } (c'_x c'_y) \neq (\bar{c}_x \bar{c_y}) \\ V_1\left|c_x c_y, xy\right\rangle \otimes V_2\left|c'_x c'_y, x'y'\right\rangle & \text{otherwise} \end{cases}$$

The extension for more particles can be achieved taking the same steps, however we need to keep in mind the computational resources required. In case we have only one particle in a

two dimensional lattice $N \times N$ the Hilbert dimension is given by $\mathrm{Dim}(\mathscr{H}) = 4N^2$, thus if we employ n particles the dimension of the new Hilbert space will increase exponentially in n, $\mathrm{Dim}\left(\mathscr{H}^{\otimes n}\right) = 4^n N^{2n}$.

8.2 Dynamics analysis

This section is reserved to try to understand the dynamics of the two interacting quantum walkers. Instead of presenting the numerical results achieved from the QHPP model and then trying to explain them, we will discuss the dynamics of each part separately, since both the HPP part and the coined walker without interaction are well understood. Only after we will show our results.

Before we continue in this section, it is important we say now that although our model was described in terms of a periodic boundary condition, all results that we will present here the walkers do not cross the boundary. Thus, in all cases, the evolution goes until there is some probability of the walkers arrive on the border. Besides, in order to facilitate the understanding of the dynamics and to simplify the notation, we will often work with the arrows notation introduced in (8.2) for the momentum direction, for instance $|00\rangle = |\nearrow\rangle$.

- HPP

  Let us analyze some possible behaviors of the two walkers under the HPP interaction, without the action of the coin, where in the end we are only interest in analyzing the action of V_{Int}. Despite the fact that the behavior of many particles which can collide, following the HPP interaction, is hard to predict, as we saw in chapter 2, for two particles the analysis becomes easier. There are two interesting cases to see for two particles: **the non-interaction behavior** and **the periodic interaction**.

  The non-interaction behavior: Unlike the multiparticles undergoing the HPP interaction, where there are many collisions happening simultaneously, with our two particles with periodic boundary conditions, in the most cases we will not see any interaction going on. For instance, if we have two particles starting from different points with the same momentum direction, or opposite direction, but with some translation displacement between them, these particles can not arrive at the same point simultaneously and thus they will never interact. Another case to consider is when the particles can meet at the same point, but they have not opposite direction, for instance, with the particle one with $\nearrow$ as its momentum direction and the second $\nwarrow$. All these cases included here the particles remains moving in the same direction for all time. In the equation below we illustrate one of these cases

$$U_{\mathrm{Int}}^{t}(|\nearrow\rangle |x_1, y_1\rangle \otimes |\nearrow\rangle |x_2, y_2\rangle) = |\nearrow\rangle |x_1 \oplus t, y_1 \oplus t\rangle \otimes |\nearrow\rangle |x_2 \oplus t, y_2 \oplus t\rangle ,$$

  where $x_1 \oplus t$ is sum mod N, N is the box length. Now let us move to the other case.

The periodic interaction: now imagine a initial condition where we have these two particles colliding at the center of the lattice, where N is a odd number. We are aware that in the collision instant the particles can be in one of the options showed in Fig.(8.2). As a consequence of the collision rule, the particles will flip their direction and they will keep moving freely on the lattice. However, due to the periodic boundary condition, these particles will meet themselves again at the same point of the previous collision, again with opposite velocity direction. Therefore, with a period $t_p = N$ these particles will collide again. Thus, at each collision these particles flip their direction until they return to their initial condition, that happens after four collision. The equation below exemplifies the case

$$U_{\text{Int}}^{\alpha t_p}\left(\left|c_x, c_y\right\rangle |x,y\rangle \otimes \left|c'_x, c'_y\right\rangle |x,y\rangle\right) = \left|c_x, c_y\right\rangle |x,y\rangle \otimes \left|c'_x, c'_y\right\rangle |x,y\rangle ,$$

where $\alpha = 4n$ with $n \in \mathbb{Z}$ and $x = y = (N+1)/2$. Despite the fact we only have analyzed one specific case, we will see a similar behavior for collisions in different points as long as they have opposite direction.

From this short analyzes we can conclude that the HPP interaction will not lead the two walkers become coupled. In contrast, this rule leads the walkers spread inside the lattice, as we showed in chapter 2 for many particles.

We know that in the QHPP model the quantum superposition comes from the coined evolution. Thus, in order to understand only this part of the model, V, Eq.(8.3) we will do a similar analysis showed for the HPP part.

- 2D CQW

  Since we only applied the Grover coin in our model

$$C = G = \frac{1}{2}\begin{pmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{pmatrix}, \tag{8.10}$$

  we will only consider this coin in this dynamics analysis. AS we did in chapter 6 we can establish the recurrence relation in terms of amplitudes and probabilities for this model.

  The four possible momentum directions are established from the four-dimensional coin space $\mathscr{H}_C = \mathbb{C}^4$ and thus the state vector can be expressed as the spinor of four components

$$|\Psi(t)\rangle = \sum_{x,y=0}^{N-1} \begin{bmatrix} \psi_{00}^t(x,y) \\ \psi_{01}^t(x,y) \\ \psi_{10}^t(x,y) \\ \psi_{11}^t(x,y) \end{bmatrix} |x,y\rangle , \tag{8.11}$$

where the subscripts denote the same four directions showed in (8.2). Then, the probability of finding the walker in the point (x,y) at time t for each coin state is

$$\begin{aligned} P^t_{00}(x,y) &= \left|\psi^t_{00}(x,y)\right|^2, \\ P^t_{01}(x,y) &= \left|\psi^t_{01}(x,y)\right|^2, \\ P^t_{10}(x,y) &= \left|\psi^t_{10}(x,y)\right|^2, \\ P^t_{11}(x,y) &= \left|\psi^t_{11}(x,y)\right|^2, \end{aligned}$$

and the probability of finding the walker in the point (x,y) at time t is

$$P^t(x,y) = \left|\psi^t_{00}(x,y)\right|^2 + \left|\psi^t_{01}(x,y)\right|^2 + \left|\psi^t_{10}(x,y)\right|^2 + \left|\psi^t_{11}(x,y)\right|^2. \tag{8.12}$$

In order to understand this dynamics better we can express the recurrence relation for both quantities, the amplitudes and the probabilities of each component. After a straightforward computation using the Grover coin (8.10) we get

$$\begin{aligned} \psi^{t+1}_{00}(x,y) &= \frac{1}{2}\left(-\psi^t_{00}(x-1,y-1) + \psi^t_{01}(x-1,y-1) + \psi^t_{10}(x-1,y-1) + \psi^t_{11}(x-1,y-1)\right), \\ \psi^{t+1}_{01}(x,y) &= \frac{1}{2}\left(\psi^t_{00}(x+1,y-1) - \psi^t_{01}(x+1,y-1) + \psi^t_{10}(x+1,y-1) + \psi^t_{11}(x+1,y-1)\right), \\ \psi^{t+1}_{10}(x,y) &= \frac{1}{2}\left(\psi^t_{00}(x-1,y+1) + \psi^t_{01}(x-1,y+1) - \psi^t_{10}(x-1,y+1) + \psi^t_{11}(x-1,y+1)\right), \\ \psi^{t+1}_{11}(x,y) &= \frac{1}{2}\left(\psi^t_{00}(x-1,y-1) + \psi^t_{01}(x-1,y-1) + \psi^t_{10}(x-1,y-1) - \psi^t_{11}(x-1,y-1)\right). \end{aligned}$$

Working with these quantities we can easily establish the recurrence relation in terms of probabilities, for instance

$$\begin{aligned} P^{t+1}_{00}(x,y) &= \frac{1}{4}\left[P^t(x-1,y-1) - \psi^t_{00}(x-1,y-1)\left(\psi *^t_{01}(x-1,y-1) + \psi *^t_{10}(x-1,y-1)\right.\right. \\ &+ \left.\left.\psi *^t_{11}(x-1,y-1)\right) + \psi^t_{01}(x-1,y-1)\left(\psi *^t_{10}(x-1,y-1) + \psi *^t_{11}(x-1,y-1)\right)\right. \\ &+ \left.\psi^t_{10}(x-1,y-1)\psi *^t_{11}(x-1,y-1) + h.c\right]. \end{aligned}$$

Adding the probability equation expressed above to the other three analogous terms $P^{t+1}_{01}(x,y)$, $P^{t+1}_{10}(x,y)$ and $P^{t+1}_{11}(x,y)$ we get the follow result

$$\begin{aligned} P^{t+1}(x,y) &= \frac{1}{4}\left(P^t(x-1,y-1) + P^t(x+1,y-1) + P^t(x-1,y+1)\right. \\ &+ \left.P^t(x-1,y-1)\right) + \text{interference terms.} \end{aligned} \tag{8.13}$$

From the equation above it is clear that the first four terms represent the classical part that can be achieved from the Random Walk in a two-dimensional lattice and the interference terms come from the quantum part. Starting from some initial state $\left|\psi(0)\right\rangle$, with the expression (8.13) we can establish the probability distribution for $t' > 0$ working with this expression recursively. In particular the following initial condition

$$\left|\psi(0)\right\rangle = \frac{1}{2}(|\nearrow\rangle - |\searrow\rangle - |\nwarrow\rangle + |\swarrow\rangle)|x = \lfloor N/2\rceil, y = \lfloor N/2\rceil\rangle, \tag{8.14}$$

where $\lfloor \cdot \rceil$ is the nearest integer function of real number and N is the lattice size, yields a probability distribution for the Grover coin as shown in figure (8.3).

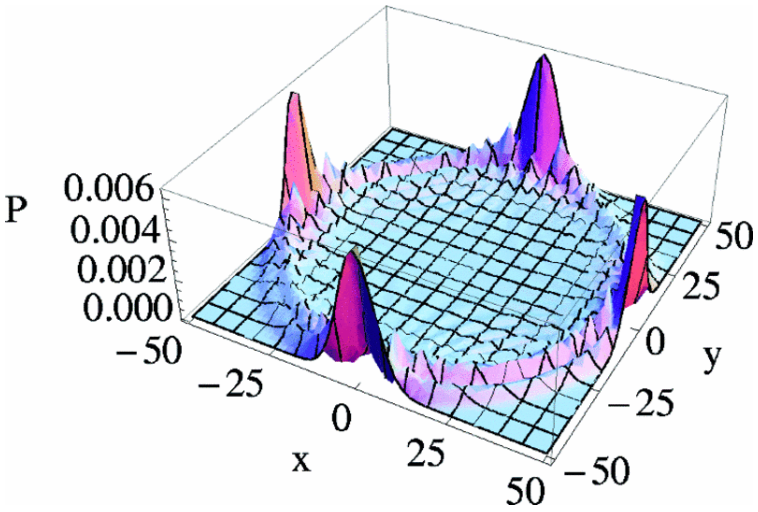


Figure 8.3: This figure gives the spatial probability distribution at time $t = 50$ for the Grover walker. The initial condition employed for the state is the one showed in Eq.(8.14). Figure from [25].

The superposition in the coin state (8.14) is the one that gives the largest standard deviation for the Grover coin, as computed analytically in [66].

Now we are ready to present our results and then discuss the dynamics achieved using the knowledge of each separate part discussed here.

We used six different initial states $|\psi(0)\rangle$ in order to learn some characteristics of this model. There are two initial states that were extremely important for the verification of the model. With one of them we could turn off the interaction between the particles, and with the other we could see only the HPP part acting.

8.2.1 Numerical results

We used a 30×30 lattice size during our numerical investigations, marked from 0 to 29. Thus, the center of the lattice, which is a reference point for our analysis, is (14,14). We employed six different initial states where three of them are separable product states and the others three are maximally entangled states between the coins. These last states, that are generalization of the Bell state for a four qubits (the maximally entangled bipartite state to a higher dimensions),

were proposed in [71]. The initial states are:

$$|\text{Sep1}\rangle = \frac{1}{2}(|\nearrow\rangle - |\searrow\rangle - |\nwarrow\rangle + |\swarrow\rangle)|14,14\rangle \otimes \frac{1}{2}(|\nearrow\rangle - |\searrow\rangle - |\nwarrow\rangle + |\swarrow\rangle)|14,14\rangle, \tag{8.15}$$

$$|\text{Sep2}\rangle = \frac{1}{2}(|\nearrow\rangle - |\searrow\rangle - |\nwarrow\rangle + |\swarrow\rangle)|13,13\rangle \otimes \frac{1}{2}(|\nearrow\rangle - |\searrow\rangle - |\nwarrow\rangle + |\swarrow\rangle)|15,15\rangle, \tag{8.16}$$

$$|\text{Grover}\rangle = \frac{1}{2}(|\nearrow\rangle - |\searrow\rangle - |\nwarrow\rangle + |\swarrow\rangle)|14,14\rangle \otimes \frac{1}{2}(|\nearrow\rangle - |\searrow\rangle - |\nwarrow\rangle + |\swarrow\rangle)|13,13\rangle, \tag{8.17}$$

$$|\text{Ent1}\rangle = \frac{1}{2}(|\nearrow\rangle|\nearrow\rangle + |\searrow\rangle|\searrow\rangle + |\nwarrow\rangle|\nwarrow\rangle + |\swarrow\rangle|\swarrow\rangle)|14,14\rangle \otimes |14,14\rangle, \tag{8.18}$$

$$|\text{Ent2}\rangle = \frac{1}{2}(|\nearrow\rangle|\swarrow\rangle + |\searrow\rangle|\nwarrow\rangle + |\nwarrow\rangle|\searrow\rangle + |\swarrow\rangle|\nearrow\rangle)|14,14\rangle \otimes |14,14\rangle, \tag{8.19}$$

$$|\text{Ent3}\rangle = \frac{1}{2}(|\nearrow\rangle|\nwarrow\rangle + i|\searrow\rangle|\swarrow\rangle - |\nwarrow\rangle|\nearrow\rangle + i|\swarrow\rangle|\searrow\rangle)|14,14\rangle \otimes |14,14\rangle. \tag{8.20}$$

We can see that in the first three initial states, the separable product states, we used the same superposition in the coin state given in Eq.(8.14).

8.2.1.1 Probability distribution

In this part we will show our results achieved for the probability distribution of just one particle for $t = 15$, that means that we applied the operator U_{HPP} fifteen times. The one particle probability distribution, $P^t(x_1, y_1)$, after t steps is given by the following expression

$$P^t(x_1,y_1) = \sum_{c_{x_1},c_{y_1}=0}^{1} \sum_{c_{x_2},c_{y_2}=0}^{1} \sum_{x_2,y_2=0}^{29} \left|\langle c_{x_1}c_{y_1}, x_1y_1| \otimes \langle c_{x_2}c_{y_2}, x_2y_2| (U_{\text{HPP}})^t |\psi(0)\rangle\right|^2. \tag{8.21}$$

- $|\text{Grover}\rangle$

  We initiate showing this result, since it is the one that we have noninteracting walkers. Thus, they have a typical behavior of the Grover coin, the reason that we called this state "Grover". These noninteracting behavior can be easily understood, since the walkers started from neighboring points. Then, during the evolution they always alternate their position, as we illustrated in Fig.(8.4). In Fig.(8.5) we can see the probability distribution of one walker at time $t = 15$. This amount of evolution leads the walker arrive in the boundary of the lattice, as we expected. We can compare the distribution achieved in Fig.(8.3) with the one showed in Fig.(8.5) and see that they agree.

- $|\text{Ent1}\rangle$

  In the last result we saw a case where the operator U_{HPP} is reduced to V_1 in the first and V_2 in the second particle. Now let us see the case where U_{HPP} is reduced to U_{Int}. Doing a quick analysis of $|\text{Ent1}\rangle$, Eq.(8.18), we can see that the four terms of the entangled state are the ones that we have both particles with the same momentum direction. From our proposed model (8.8), this state leads to the action of V_{Int}, since they are at the same point. Now, moving to the analysis of (8.7) we see that from the prescription of $\mathscr{C}$, Eq.(8.9), this

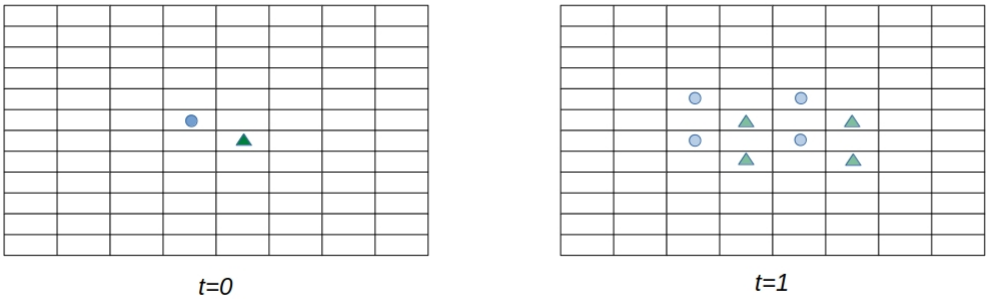


Figure 8.4: We can see that with this type of initial condition |Grover2⟩ the particles will not interact during their evolution. The circle represents one walker and the triangle represents the other one. At time $t = 0$ each one is located on distinct points of the lattice, but they are neighbors points. Subsequently, at time $t = 1$ each one spread diagonally into the four possible directions, but they do not touch each other. If we continue with the same kind of analysis we will see that they will stay with alternate positions.

operator will act trivially, since the particles do not have opposite momentum directions. Then, the action of V_{Int} is reduced to the shift operator in each walker, see figure (8.6). The four components in the superposition of |Ent1⟩ will remain the same during all evolution, since we will not have the action of V_1 and V_2. Despite the fact that the QHPP model does not give us coupled particles, this coupled behavior here was enforced by the initial condition. Besides, our model does not allow these particle to decouple. Therefore, the four peaks achieved in the probability distribution at $t = 15$, Fig.(8.7), are expected from the analysis we did here.

- |Sep1⟩

  Now we are in the case where the QHPP dynamics shows up, but not completely, as we will explain. As we can observe from Eq.(8.15) the two particles are located at the same point and both have the same superposition in the coin state, like the one of (8.14). This superposition in both walkers leads to 16 terms, where 4 terms are the ones where the walkers have the same momentum direction. Thus, these terms represent the superpositions of coupled particles, that will be preserved along the evolution, like we saw in |Ent1⟩. At the same time we have 4 terms where both particles have opposite directions, that will suffer the collision from the beginning After the first time step we will see four terms that represent the coupled particles, four that suffered collision and the remainder that will keep with their momentum direction. In the second step, except for the 4 coupled states, the coin operator will lead to a superposition and subsequently, after the shift operator action, we will see amplitudes terms from all the four diagonal points returning to the starting point.

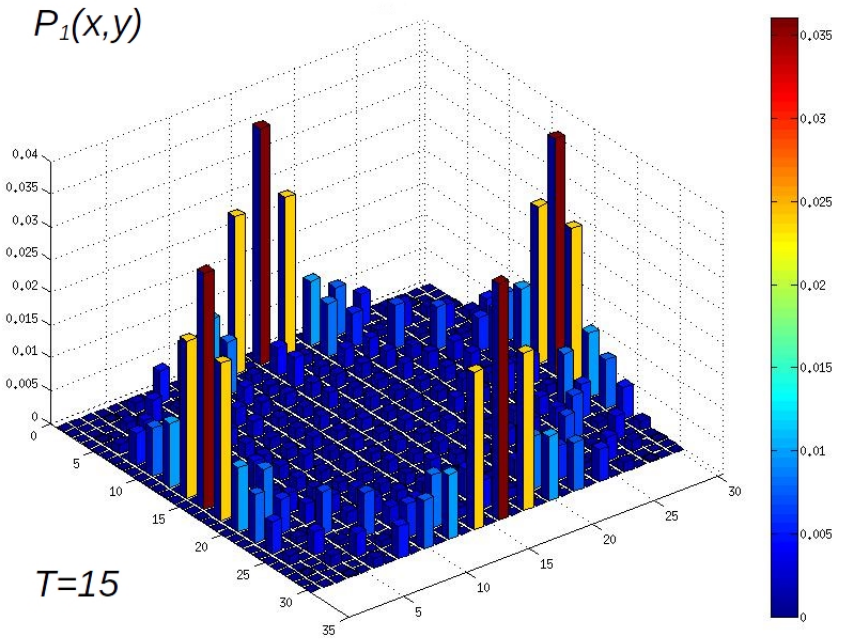


Figure 8.5: Probability distribution for one walker after 15 steps, where the initial state is given by |Grover⟩.

Consequently, these superposition terms at the initial point have opposite directions and thus, they will suffer interaction. During the entire evolution we will see amplitudes go back and forth into the initial point, which give to us a peak in the center of the lattice, as we can confirm in Fig.(8.8). The other terms around the lattice will remains moving freely without interaction, following the Grover dynamics. Therefore, in the end of the evolution, $t = 15$, we expect to see terms that represent coupled particles and others that represent the superposition terms that did not face any interaction. We cold confirm in our numerical analyses Fig.(8.8) these predicted results.

- |Sep2⟩

  Now we are in the case where we will not have coupled particles, since each one started in different points. After the first time step we will see amplitudes terms going to the center of the lattice, with opposite momentum direction and amplitudes terms spreading diagonally to the remaining direction. While the terms in the center will start to feel the interaction, the others amplitudes terms will spread freely along the lattice. We can visualize this idea in Fig.(8.9). Then, from this quick analysis we can expect the pattern of collision in the center and the Grover speeding around the other directions, that can be confirmed in Fig.(8.9).

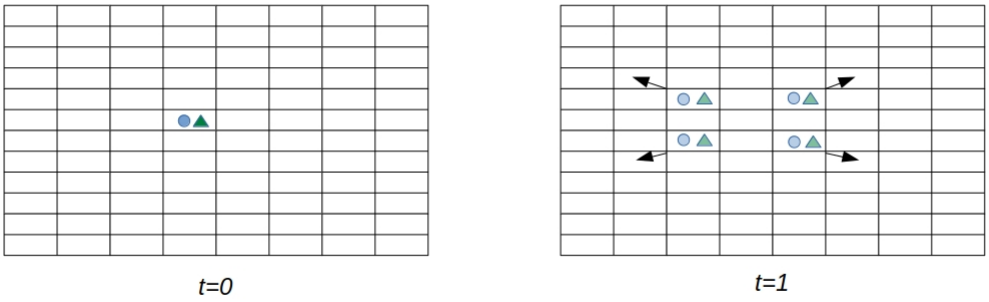


Figure 8.6: We can see that with this type of initial condition |Ent1⟩ the particles remains coupled during the entire evolution. The arrows indicates the movement direction of both particles that will not suffer any superposition during their evolution.

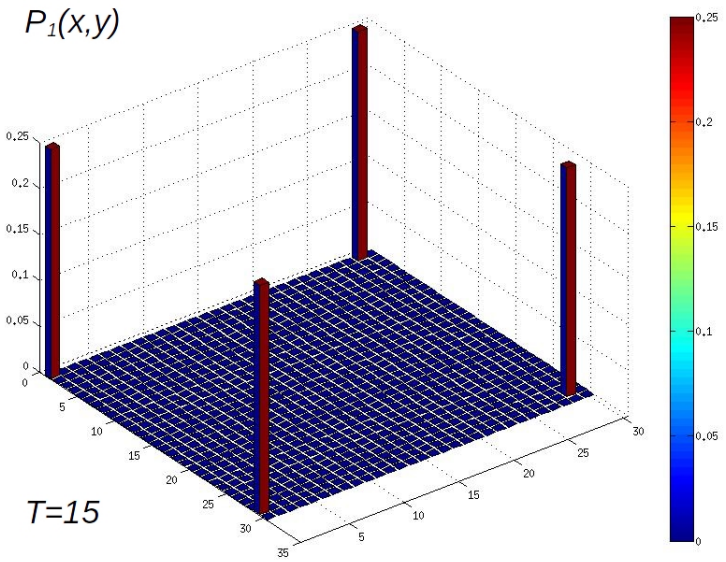


Figure 8.7: Probability distribution for one walker after 15 steps, where the initial state is given by |Ent1⟩.

- |Ent2⟩

  We are in the case of entanglement in the coin space between the particles, where the momentum directions are opposite Eq.(8.19). These four terms will lead to a strong effect of collision, different of the previous examples, since all terms begin colliding at time $t = 0$. In our numerical result Fig.(8.11) we could see a big peak in the center of the lattice, that

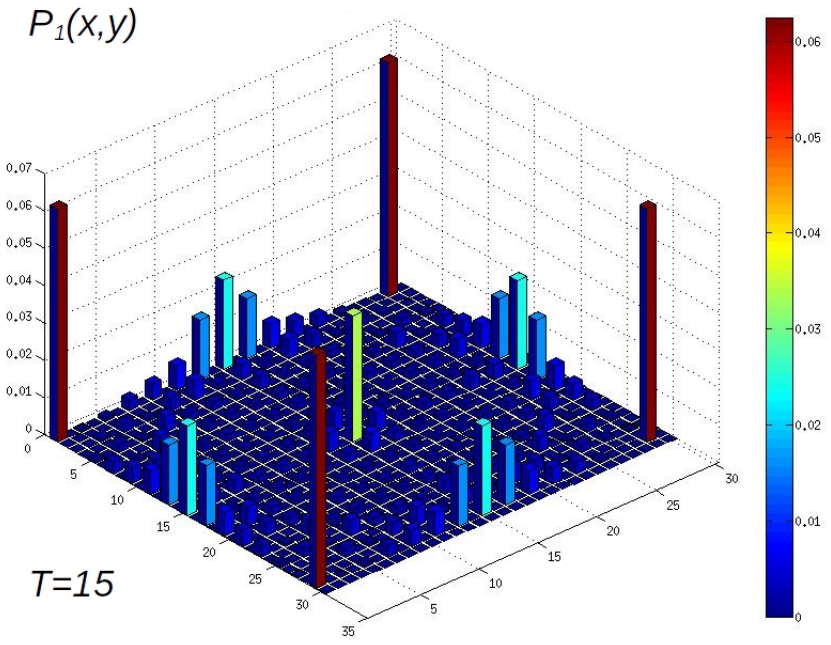


Figure 8.8: Probability distribution for one walker after 15 steps, where the initial state is given by |Sep1⟩.

we know to come from the interaction part. The remaining amplitudes, related with the noninteracting part, are small in this case.

- |Ent3⟩

  We are in the case really similar to |Ent2⟩. But now in this entangled state (8.20) the momentum directions are not opposite, which leads to a slightly different pattern in the probability distribution little different Fig.(8.12), since the collisions started only in the second step. But the peak at the center remained strong.

8.2.1.2 Standard deviation

As we said in the beginning of this chapter, we are interested in applying this model for some algorithms, for instance for search problems [59]. The Grover coin, for example, with the initial condition given by (8.14) is the one that gives the highest value for the standard deviation, a model often applied for algorithms, compared with other initial conditions and with other coins, like the Hadamard coin [59]. Thus, in order to analyze the QHPP model for future algorithms we computed the standard deviation for all these initial conditions explored until here. We used |Grover⟩ as our comparison criteria for the performance of our model, since it is (8.17) the initial state that yields the behavior of noninteracting walkers and the one that have the highest value

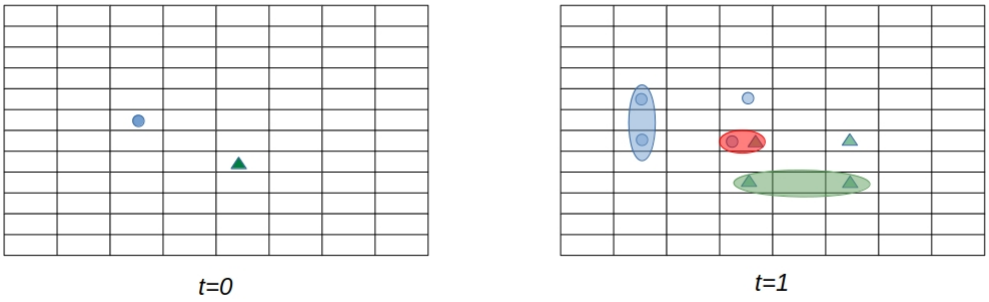


Figure 8.9: At time $t = 0$ the state |Sep2⟩ represents both particles localized in different points. Subsequently, at time $t = 1$, we can see amplitudes terms related with the collision in the center. We marked by red the region related with the collision and by green and blue the part of the dynamics that will be strongly related with the Grover dynamics with the initial state given by (8.14), for each walker.

for the standard deviation in the coined walkers models. The formula employed for the standard deviation of the position of one particle was

$$\begin{aligned} \sigma(t) &= \sqrt{\sigma_x^2(t)+\sigma_y^2(t)}, \\ &= \sqrt{\sum_x (x-\bar{x})^2 P^t(x) + \sum_y (y-\bar{y})^2 P^t(y)}, \end{aligned} \tag{8.22}$$

where

$$\begin{aligned} \bar{x} &= \sum_x x P^t(x), \\ \bar{y} &= \sum_y y P^t(y). \end{aligned}$$

The results in Fig.(8.13) show that there are two initial cases that overpassed the standard deviation values of |Grover⟩. The first is |Ent1⟩, where we have only amplitude terms of coupled particles. The second is |Sep1⟩ which also contain terms of coupled particles. Coupling between particles that seems to be the dominant factor that leads them to have a higher standard deviation value than |Grover⟩.

The |Sep2⟩ is a case that deserves our attention. This state characterizes well the QHPP model, since it is the case that we do not enforce the walkers to be coupled, and have a high value for the standard deviation. From Fig.(8.13) we can see that although |Sep2⟩ has the standard deviation value smaller than |Grover⟩ they are close.

8.2.1.3 Distance

In the standard deviation analysis, we only investigated the probability distribution behavior of one particle, without taking into consideration the probability behavior of the second walker.

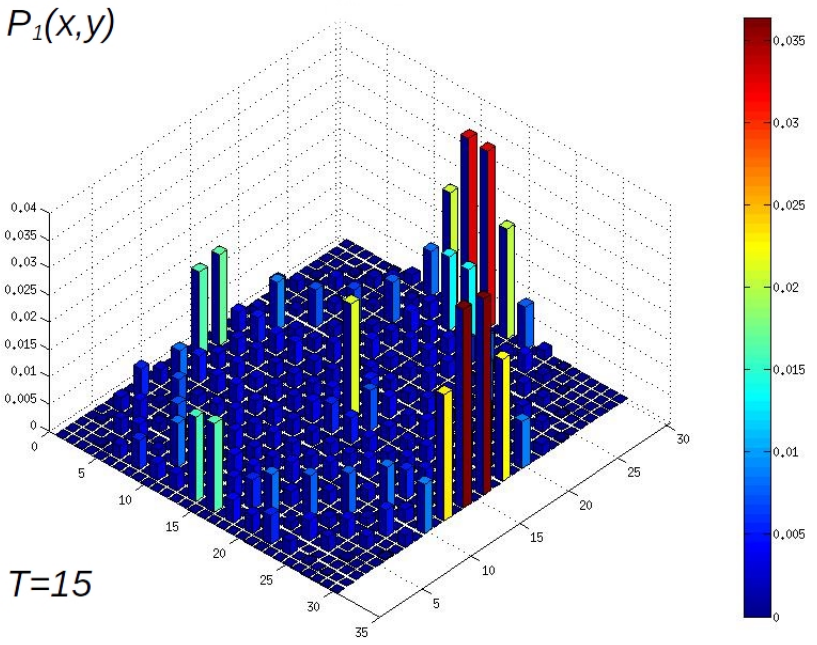


Figure 8.10: Probability distribution for one walker after 15 steps, where the initial state is given by |Sep2⟩.

Although the probability distribution established is influenced by the interaction with the second walker we do not have information whether the particles are close or distant during their evolution. Thus, with the goal of capturing this information in some way, we used the following formula

$$D(t) = \sum_{x_1,y_1} \sum_{x_2,y_2} P^t(x_1,y_1;x_2,y_2)\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}\,. \tag{8.23}$$

In this expression we are computing at each time step t the probability of we have particle one at the point (x_1, y_1) and the second at the point (x_2, y_2) multiplied by the euclidean distance between them, taking the sum for all points of the lattice. This equation contains information if the particles become closer or distant along their evolution. For instance, $D(t)$ for |Ent1⟩ is zero during the whole evolution, since we have only amplitude terms associated with coupled particles.

In Fig.(8.14) we can see our results. The fact that we have |Sep2⟩ with bigger value during the total evolution is due the fact that the particles started with the highest distance, compared with the other states. In fact, the |Grover⟩ state that has the higher inclination curve, suggesting that the noninteracting particles are the case where they stay more distant of each other. Moreover the states |Sep1⟩ and |Sep2⟩ have basically the same curve inclination which is also true for |Ent2⟩ and |Ent3⟩, where the last states are the ones with the smallest inclination. We can understand

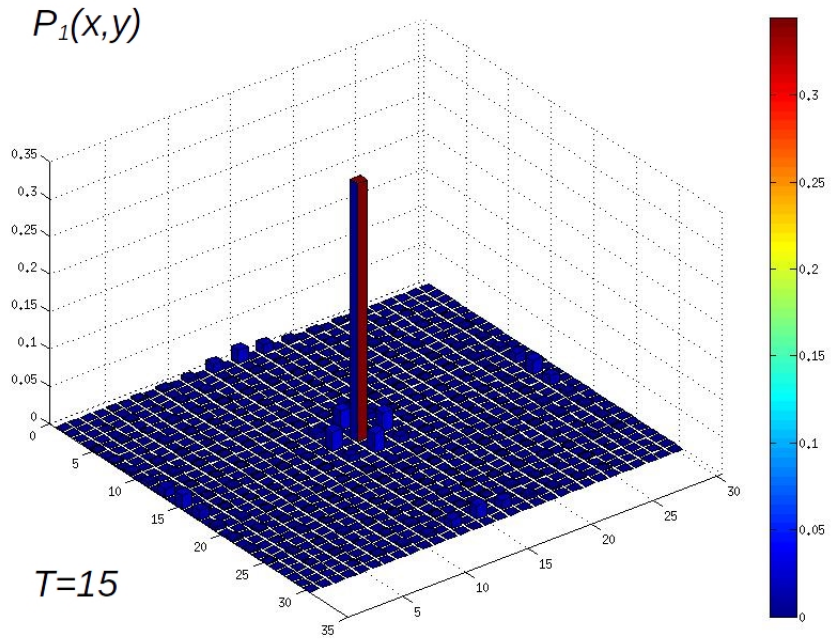


Figure 8.11: Probability distribution for one walker after 15 steps, where the initial state is given by |Ent2⟩.

the small inclination for the entangled initial states because of their strong collision behavior that leads the particles to become closer in the center of the lattice.

8.3 Entanglement between the particles

There are already works where the amount of entanglement between the coin and the position is investigated [2, 46]. Thus, we can consider that for a single particle this study was well explored. Despite the fact that there is an analysis of entanglement between two interacting walkers [9], these studies are just at their beginning. In [9] they used the von Neumann entropy to measure the total entanglement between the subsystems describing each of the two particles in quantum walks. Although in [9] they did this investigation on arbitrary graphs, they focused their analyses on two walkers on the infinite line. During their investigations they could see, for instance, the entanglement between the particles to increase as they started to interact.

Similarly to the results briefly discussed above, we will investigate the dynamics of entanglement between the degree of freedom of first walker and the degree of freedom of the second walker on the two-dimensional lattice using the von Neumann entropy S

$$S(\rho_1) = -\mathrm{Tr}\left(\rho_1 \log \rho_1\right), \tag{8.24}$$

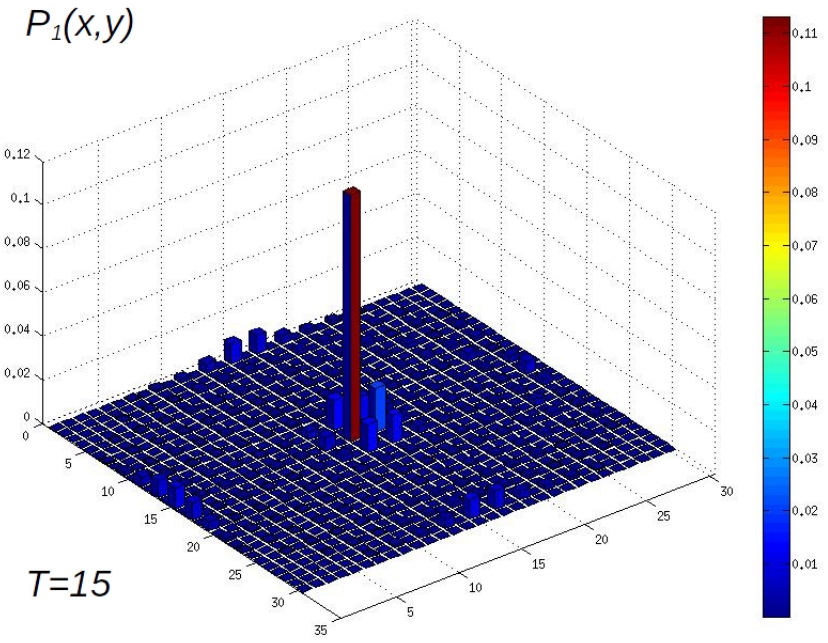


Figure 8.12: Probability distribution for one walker after 15 steps, where the initial state is given by |Ent3⟩.

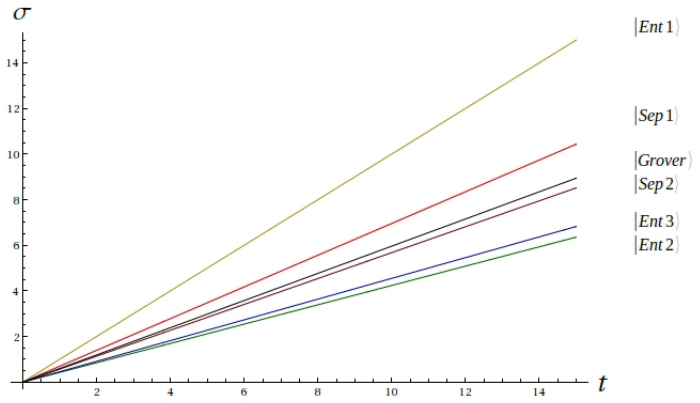


Figure 8.13: Standard deviation of the quantum HPP model on the two-dimensional lattice.

with $0\log_2 0 := 0$. Since the trace is invariant under similarity transformations and the density matrix ρ_1 has real, nonnegative eigenvalues λ_i, Eq.(8.24) can be written in terms of λ_i as follows

$$S(\rho_1) = -\sum_i \lambda_i \log \lambda_i. \tag{8.25}$$

ρ_1 is the reduced density matrix that belongs to $\mathscr{L}(\mathscr{H}_1)$ the set of all linear operators acting on $\mathscr{H}_1 = \mathscr{H}_C \otimes \mathscr{H}_{L^2}$. The matrix ρ_1 is obtained by tracing the density matrix $\rho = |\psi\rangle\langle\psi|$ over the

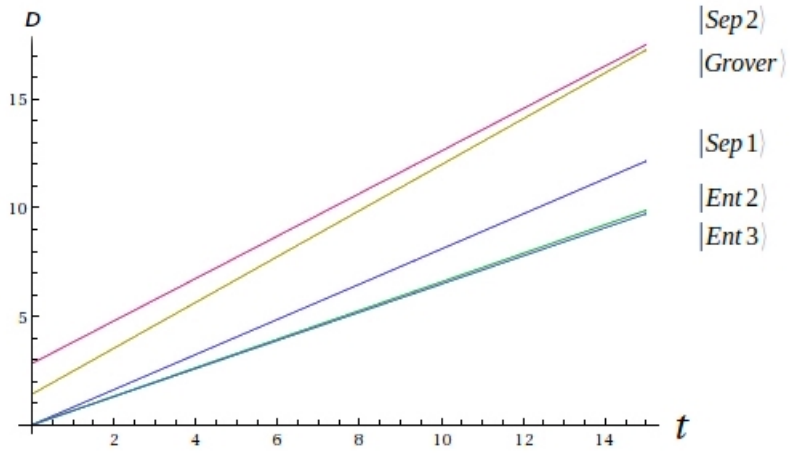


Figure 8.14: Distance analyses of the quantum HPP model on the two-dimensional lattice.

subsystem 2,

$$\rho_1 = \mathrm{Tr}_2\left(\rho\right),$$

where $|\psi\rangle$ is the pure state in $\mathscr{H}_1 \otimes \mathscr{H}_2$ given by Eq.(8.6). We do not have to concern in doing this study with the second walker, since $S\left(\rho_1\right) = S\left(\rho_2\right)$. This equality holds since for bipartite system both reduced density matrix share the same eigenvalues (modulo multiplicities of zero). We can check it easily via the Schmidt decomposition [28]. Moreover, we are aware that complete mixed states are the ones that yields the maximum von Neumann entropy,

$$S\left(\frac{I}{d}\right) = \log_2 d,$$

where $d = \mathrm{Dim}\left(\mathscr{H}_1\right) = 4N^2$ and I the $d \times d$ identity matrix. Thus, since we know the lattice size we have already an upper bound for the amount of entanglement:

$$S\left(\rho_1\right) \leq 2\log_2(2N). \tag{8.26}$$

In this part of the work we used a smaller lattice, 20×20, since the numerical computation of the entropy has a high cost. We can understand better the complexity related with this calculation from the complexity involved in computing the eigenvalues of a $d \times d$ matrix, which is $O(d^3)$. Thus, since we have to compute the eigenvalues of ρ_1 in order to calculate (8.25) and d, in this case, is $4N^2$ we can see why this computation is difficult for bigger lattices.

Similarly to the results presented for the probability distribution, we will show and discuss the results for each initial state. We will not include the states |Grover⟩ and |Ent1⟩ in this discussion. The reason to not include |Grover⟩ is that this state does not have interaction between the walkers, and thus the entropy remains zero during the entire evolution. For the state |Ent1⟩ we are in the case where the particles remain coupled during all evolution, and thus the entropy remains constant equal two since this state is the case of the maximally entangled states between the coins, $\log_2(4) = 2$. Both cases were confirmed numerically.

- |Sep1⟩

  For this state we observed in Fig.(8.15) that only at $t = 3$ the entanglement between the walkers appears. If we observe this state and its dynamics, we can see that in this time instant there are amplitude terms that just left the center of the lattice after suffering one collision. We also can see from Fig.(8.15) that after $t = 6$ the entropy remains almost constant. We can understand it from the fact that there are several amplitude terms that do not suffer interaction during the evolution and consequently the entanglement stops to increase.

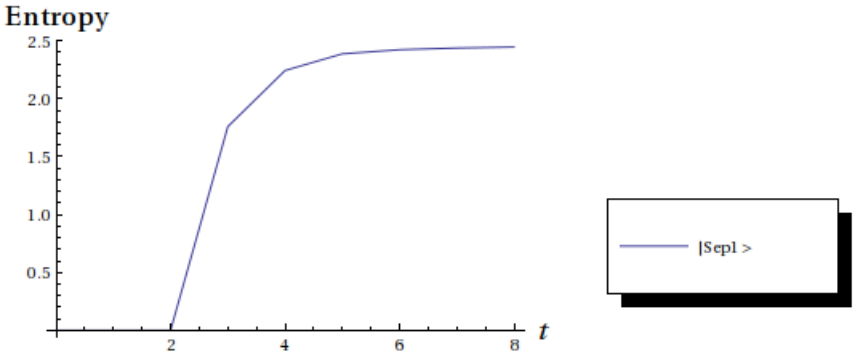


Figure 8.15: Entropy evolution starting from |Sep1⟩.

- |Sep2⟩

  For this state we also could see the entanglement between the particles increasing during the evolution. In Fig.(8.16) we can see that different from the previous case entanglement starts to appear earlier. We can understand it from the fact that in (8.16) they started in different points but again at time $t = 2$ there are amplitude terms that just left the center of the lattice after a collision. We also can see from the results in Fig.(8.16) that the entanglement did not increase much when compared with |Sep1⟩. We can understand it from the fact that we have less amplitude terms interacting in this state. Indeed, we can check that while we have four amplitude terms that collide at $t = 2$ in |Sep2⟩, here we have two amplitude terms related with the collision at time $t = 1$.

  From $t = 1$ until $t = 7$ we could see that these points increased logarithmic. We could fit these points by the following curve

  $$a \log_2(bt), \tag{8.27}$$

  for small t since we have a bound estimated in Eq.(8.26), with $a = 0.668$ and $b = 1.064$.

- |Ent2⟩

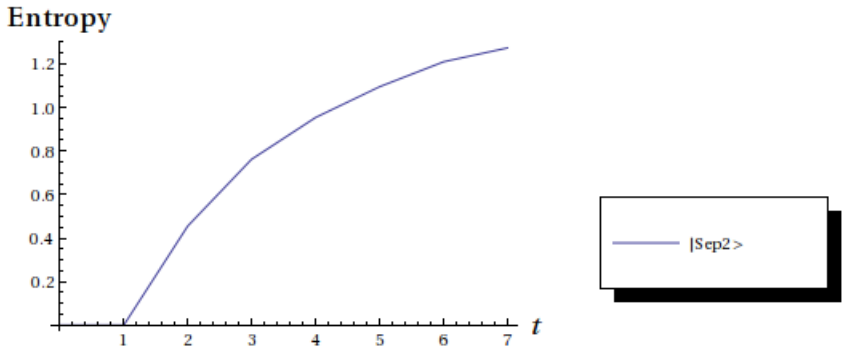


Figure 8.16: Entropy evolution starting from |Sep2⟩.

Here we are in the case where the particles are already entangled at $t = 0$. In our results, Fig.(8.17), we observed that the entanglement remains constant until $t = 4$. Moreover, although it starts to increase at time $t = 5$ this growth is small. It means that the collision effects were not able to entangle more these particles and that might have many other amplitude terms that do not collide during the evolution.

Like we did for |Ent2⟩ we could fit the growth from $t = 4$ until $t = 7$ by a logarithm curve (8.27), but with parameters given by $a = 0.094$ and $b = 3.26$.

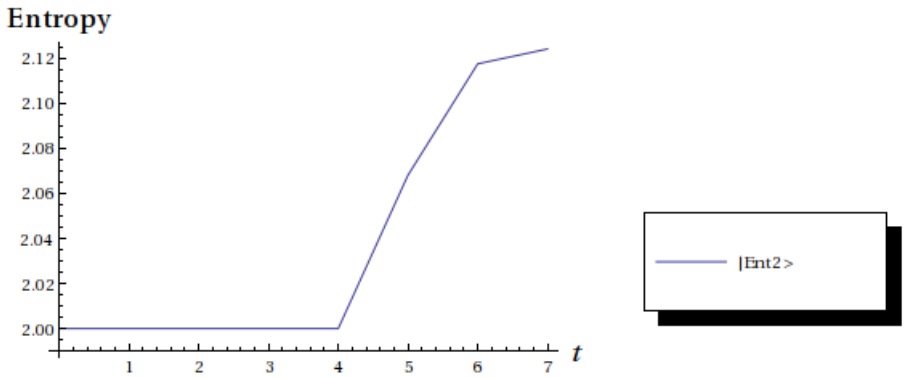


Figure 8.17: Entropy evolution starting from |Ent2⟩.

- |Ent3⟩

  We are again in the case where the particles are maximally entangled states between the coins at time $t = 0$. In Fig.(8.18) we can see that the entanglement increases with time. But differently from previous states, this growth is bigger. This result suggests to us that this state undergoes more interactions during the evolution.

  We also can see a logarithm increase from $t = 2$ until $t = 7$ with $a = 0.703$ and $b = 4.718$.

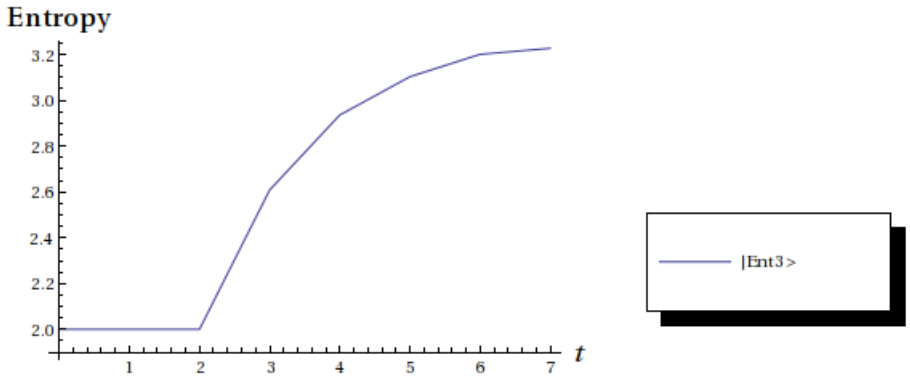


Figure 8.18: Entropy evolution starting from |Ent3⟩.

It is important to point out that in all these numerical analyses for entropy the upper bound (8.26) , which in this case is 10.644, was respected.

8.3.1 Final considerations

In this chapter we introduced a quantum walk model for two interacting walkers following the interaction rule of HPP, proposed in [37], that we called by QHPP model. In this first moment we only studied, by numerical analysis, some dynamics' aspects of this model. From six different initial states we could confirm that we successfully implemented the model, since we could recover the expected results for the Grover and HPP part. We have already got results, like the ones presented about the standard deviation, that point to us that QHPP can be a good model to be used in some algorithms, since from just these six initial states we found cases that give higher values for the standard deviation when compared with the Grover state. Furthermore, we could see how easily this model generates entanglement between the particles, which can be further explored in future works, when we, for example, define temperature between these parts like it was done in [64].

For the next steps we expect to do an analytical study for this model, to see if we can get a better understanding of the QHPP. At the same time we will investigate the behavior of this model for search algorithms, since we believe that with two particles we can gain some speedup compared with other quantum search algorithms. In addition, in this chapter we could explore two quantum walkers in a two-dimensional lattice, something that had never done before. With this ability we can try to use this type of interaction for graph isomorphism test to a larger number of different graphs. We can either try to use QHPP to do this certification test or use the previous interactions proposed in [9, 10], since we can use our computational ability to test these previous models for different graphs.

Hardy, Pomeau and Pazzis proposed in [37] the first model for molecular dynamics. Despite the fact that the model is able to capture the essential features for a realistic problem, particle

and momentum conservation, the capability of this model to simulate a real gas is poor, due to a lack of isotropy. We can observe this drawback in Fig.(8.19). There we can see that if we put a high particle density in the middle of the lattice we can see that the particles do not propagate identically in all directions. The main reason is the lattice structure used for this model. In order

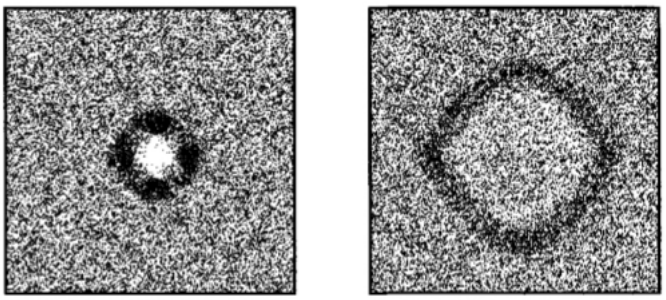


Figure 8.19: HPP simulations initiated with a high particle density in the center of the lattice, left figure. Then, after some time steps, we can see the particles do not propagate identically in all directions, right figure. This behavior it is not expected for real gas process. Figure from [17].

to solve this isotropy problem and propose a more realistic model to fluids, Frisch, Hassalacher and Pomeau proposed in [32] a model, with the same spirit as HPP lattice gas, that became known as FHP model for fluids. In this model they used an hexagonal lattice, where each site has six neighbors. This model solves the isotropy problem. Until today this model is used to simulate fluids, as it is a discrete model for the Navier Stokes, a non-linear differential equation [17].

With the same spirit here we might had given the startup for studies of more realistic "quantum fluids" (any system that exhibits quantum mechanical effects at the macroscopic level). Moreover we can try to use the same lattice structure of the FHP with its collision rule to see the behavior for two quantum walkers and try to move to more walkers and see if the quantum FHP model could be a better candidate for search algorithms.

CHAPTER 9

QUANTUM ALGORITHM FOR SIMULATING THE WAVE EQUATION

During the recent years we notice more and more specialized quantum algorithms appearing. As, there are some quantum computers available (D-wave [12], IBM [41, 69]) and others to come (Microsoft[1], Google [31]), people seek to see quantum computers solving problems with a better performance than our ordinary computers, for instance, spending less time to compute the same task. In particular, we saw during these last years many results focusing in quantum algorithms for numerical methods, that can lead a great impact on the industry. We can emphasize the results established in [38], that current is known as HHP algorithm, since the authors last name are Harrow, Hassidim and Loyd. The authors proposed a quantum algorithm to solve linear systems of equations, where they could get exponential speedup compared with the best classical algorithms, but only in special cases, for instance the matrix should be sparse. Some restrictions established in [38] were solved in [20].

It is quite obvious the importance of results like that, since it can be useful in many distinct areas, from academy to industry. Keeping in the same line, we can cite some results which proposed quantum algorithms to solve differential equations. In [7], for instance, Berry asked about the possibility to use quantum computers to solve differential equations. Berry knew that solve differential equations via numerical method is equivalent to solve linear systems and that the matrices achieved after the discretization are sparse (see chapter 4). Moreover he was aware about the results of [38] that gives great speedups in the runtime to solve linear system when A is a sparse matrix. To be more precise the runtime in [38] to solve linear systems like $Ax = b$ is given by

$$\tilde{O}\left(\log(N)s^4\kappa^2/\epsilon_L\right),$$

where $A \in N \times N$, s the sparsity of A and κ the condition number (see App.(B)) of A and ϵ_L is the

[1] https://www.microsoft.com/en-us/quantum/

allowable error. From these previous results and from the fact that any linear dependent equation can be converted to a first-order linear differential equation Eq.(4.54) they build a quantum algorithm to solve the differential equation in time $O(\text{poly}(D\log N))$ instead $\Omega(N^D)$ achieved in the classical algorithms, chapter 4. After that, in 2017, Berry, Childs ,Ostrander and Wang [16] extended the results of [7], proposing a quantum algorithm for differential equations with an exponential improvement over previous quantum algorithms for this problem. These types of algorithms are useful to simulate many classical physical systems, thus they have great potential for applications in distinct areas.

In this chapter we will maintain the line of results for differential equation. The result that we will present here [21] is a quantum algorithm for simulating the wave equation. Although, the previous quantum algorithms can solve the wave equation problem, we developed an extremely specialized quantum algorithm that can solve only the wave equation. We took a different approach to solve this problem that yields an exponential speedup compared it with the ordinary classical methods and also a better performance even when we compared with the previous quantum algorithm for differential equations.

We saw in chapter 4 that when we apply the well knowing numerical methods for differential equation, like the finite difference method [44] and the finite element method [91], we can see that the complexity involved to solve general linear differential equations have time complexity scaling exponentially with D the lattice spacing, $\Omega(T(l/h)^D)$ where h is the lattice spacing, l is the lattice size and T is the evolution time for the wave equation. However, we will present results via Hamiltonian simulations where the time complexity is $\tilde{O}(TD^2/h)$ Moreover, when we employ the previous quantum algorithms [7, 16] to solve the wave equation problem our results gave a quadratic improvement in the complexity of state preparation (l/h) against $(l/h)^2$, where this quadratic improvement is achieved in exchange for being specialized for solving wave equations rather than general linear differential equations. When we say specialized here we mean that we do not have the HHP algorithm as the main subroutine for our algorithm. Unlike the previous results [7, 16] for the wave equation simulation we did it by Hamiltonian simulations and the complexity of our method is achieved from the number of necessary gates to implement the unitary e^{-iHT}.

Therefore we will show an alternative quantum algorithm for the wave equation problem, that has a better performance even when we compared with other quantum algorithms. Unlike the previous results that are more interested in proposing the method and confirm that their time complexity is better than the classical counterpart, we show in practice few numerical examples of wave simulations via Hamiltonian simulations, imposing either Dirichlet or Neumann boundary conditions. Like in [20] we consider as our primary application the simulation of scattering in complicated geometries, as illustrated in figure 9.1. Moreover, like we did in chapter 4 we provide numerical evidence that our approach accurately simulates the wave equation with appropriate behavior at boundaries.

We strongly recommend the reader see the chapter 4 first, where we solved the wave equation applying the finite difference method. Besides in chapter 4 we present a powerful discretization method via graph theory that has a key role for the quantum algorithm that we will show here.

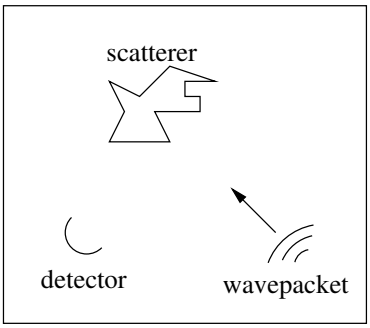


Figure 9.1: For a given initial wavepacket and a given scatterer, we would like to estimate the resulting spatial distribution of wave intensity resulting at some later time t. In particular, one may wish to know the total intensity captured by a detector occuppying some region of space. This can be estimated using a quantum simulation in which the wavefunction directly mimics the dynamics of the solution to the wave equation. The final intensity in the detector region is equal to the probability associated with the corresponding part of the Hilbert space, which can be estimated from the statistics resulting from a projective measurement. Figure from [21].

9.1 Algorithm

Before we go to the technical details of the algorithm we will discuss the strategy adopted here to construct a quantum algorithm for the wave equation problem.

We are aware that quantum algorithms demand unitary operators, thus we need them for our wave equation simulation. Then, our goal is to try to convert the numerical computation in terms of unitary operations. We were successful in this task by taking advantage of the incidence matrix, chap.(4) as we will explain now.

Differently from chap.(4), in our algorithm we only considered space discretization. Thus starting with

$$\frac{d^2}{dt^2}\phi = v^2\nabla^2\phi,$$

after the discretization, we are faced with the task of simulating

$$\frac{d^2}{dt^2}\phi = -\frac{1}{h^2}L\phi,$$

via quantum computation, where again h is the lattice spacing and we are already considering L as the graph Laplacian (4.45). The way that we encode this problem in terms of unitary operators is constructing a off-diagonal Hamiltonian with a block form, where there block matrices are the incidence matrix B,

$$H = \frac{1}{h}\begin{bmatrix} 0 & B \\ B^\dagger & 0 \end{bmatrix}. \tag{9.1}$$

We will now explain how this Hamiltonian provides the wave equation. In chapter 4 we saw that incidence matrices and the Laplacian operator are defined in terms of graphs [19]. We saw that B is a $|V| \times |E|$ matrix, where V is the vertex set and E the edge set from a graph G_h. Then, from some graph G_h there are these two sets and from G_h we can construct the incidence matrix. Since we have defined a Hamiltonian there is a Hilbert space where it is defined $H \in \mathscr{H}$. From what we know about B is quite natural we decomposed our Hilbert space as follows

$$\mathscr{H} = \mathscr{H}_V \oplus \mathscr{H}_E, \tag{9.2}$$

where $\mathscr{H}_V$ is the vertex space and $\mathscr{H}_E$ is the edge space. As a next step we will write the Schrödinger equation, employing the Hamiltonian above Eq.(9.1) into some vector $|\psi\rangle \in \mathscr{H}$

$$|\psi\rangle = \begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix},$$

where $|\phi_V\rangle \in \mathscr{H}_V$ and $|\phi_E\rangle \in \mathscr{H}_E$, and thus to use the property $BB^\dagger = L$ to codify the wave equation via Hamiltonian simulation. Following these steps we first write the Schrödinger equation

$$\frac{d}{dt}\begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix} = \frac{-i}{h}\begin{bmatrix} 0 & B \\ B^\dagger & 0 \end{bmatrix}\begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix} \tag{9.3}$$

then, we can see that

$$\begin{aligned} \frac{d^2}{dt^2}\begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix} &= \frac{-i}{h}\begin{bmatrix} 0 & B \\ B^\dagger & 0 \end{bmatrix}\frac{d}{dt}\begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix}, \\ &= \frac{-1}{h^2}\begin{bmatrix} 0 & B \\ B^\dagger & 0 \end{bmatrix}^2\begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix}, \\ &= \frac{-1}{h^2}\begin{bmatrix} BB^\dagger & 0 \\ 0 & B^\dagger B \end{bmatrix}\begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix}. \end{aligned}$$

where we adopted the natural units, which implies that the Planck constant $\hbar$ is equal to one. Then we can see that the full subspace $\mathscr{H}_V$ evolves according to a discretized wave equation. We should emphasize the fact that we are simulating the wave equation, a second order differential equation via Schrödinger's equation, a first order differential equation in time. This is possible since we have a Hamiltonian that interacts these two subspaces $\mathscr{H}_V$ and $\mathscr{H}_E$ in such a way that one subspace evolves according the wave equation.

Now we should ask ourselves what are the advantages in run this equation via a quantum algorithm. A crucial part now is to investigate the complexity involved in this quantum version. We begin this analysis looking at the dimension of Hilbert space for this problem. From (9.2) we see that $\dim(\mathscr{H})$ is is equal to the number of vertices of the graph plus the number of edges: $|V| + |E|$. In particular, for a hypercubic region of side-length l in D-dimensions, discretized

into a hypercubic grid of lattice spacing h, one has $|V| = (l/h)^D$, (see chapter 4, section (4.2.3) to understand better this equality) and $|E| = D(l/h)^D$. Thus, the number of qubits needed is $\log_2\left[(1+D)(l/h)^D\right]$. There are more two necessary informations to compute the complexity of this quantum algorithm, the largest matrix element of H, that has magnitude $1/h$, and the number of nonzero matrix elements in each row or column of H that is at most $2D$. This sparsity value of H can be easily understood taking into account that the Hamiltonian is composed by the incidence matrix which has the sparse pattern like the adjacency matrix analyzed in chapter 4, section (4.2.3).

What we did until this moment was to convert the wave equation simulation to a Hamiltonian simulation. Putting in another way, we converted a second order differential equation to a first order one, which is a technique well known, similar to the translation showed in Eq.(4.55). Then, if we stop here and proceed with this simulation via classical algorithms the complexity of the problem still the same. However, there are quantum algorithms that give better runtime to Hamiltonian simulations when the Hamiltonian is a sparse matrix, which is our case, as we use the results of [8]. There they showed that the unitary time evolution e^{-iHT} to within ϵ can be achieved using a quantum circuit of

$$g = O\left[\tau\left[n + \log^{5/2}(\tau/\epsilon)\right]\frac{\log(\tau/\epsilon)}{\log\log(\tau/\epsilon)}\right], \tag{9.4}$$

gates, where $\tau = s\|H\|_{\max}T$, where $\|H\|_{\max}$ is the largest matrix element of H in absolute value, s = sparsity of H, and n = number of qubits. For the Hamiltonian of (9.1), $s = 2D$, $\|H\|_{\max} = 1/h$, and $n = \log_2[(1+D)(l/h)^D]$, and therefore the total complexity of simulating the time-evolution is

$$\begin{aligned} g &= O\left[\frac{DT}{h}\left(\log\left[(1+D)(l/h)^D\right] + \log^{5/2}\left(\frac{2DT}{h\epsilon}\right)\right)\frac{\log\left(\frac{2DT}{h\epsilon}\right)}{\log\log\left(\frac{2DT}{h\epsilon}\right)}\right], \\ &= \tilde{O}\left(\frac{TD^2}{h}\right), \end{aligned} \tag{9.5}$$

where the notation $\tilde{O}$ indicates that we are suppressing logarithmic factors.

The remaining considerations are the implementation of desired boundary conditions, the preparation of an initial state implementing the desired initial conditions, errors induced by discretizing the wave equation, and the relative probability to obtain the desired state related with the function at time T computation. In the following sections analyze each of one of these topics commented here.

The topic related with the implementation of desired boundary conditions was explored in chapter 4, section (4.2.2). Since we already know to implement both Dirichlet and Neumann boundary conditions by the incident matrix, we know how to deal with these boundary conditions in our quantum version for the wave equation problem. Therefore we can go to the initial condition part.

9.2 Initial conditions

The first step in our quantum algorithm is to prepare a quantum state $|\phi_V\rangle \oplus |\phi_E\rangle \in \mathscr{H}$ corresponding to desired initial conditions $\phi(x,t)$ and $\partial\phi(x,t)/\partial t$ at $t = 0$. Our method for preparing the initial state and its complexity varies depending on the specific type of initial conditions.

As a first example, consider a line-segment with Dirichlet boundary conditions, discretized into four lattice sites. In this case, by (9.1) and (4.46), we have

$$H = \frac{1}{h}\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{9.6}$$

This can be viewed as a discretization of

$$H = \begin{bmatrix} 0 & \frac{d}{dx} \\ -\frac{d}{dx} & 0 \end{bmatrix}, \tag{9.7}$$

where we used the the forward and backward difference approximation in Eq.(9.6). More generally, in an arbitrary number of dimensions, the Hamiltonian (9.1) can be seen as a discretization of

$$H = \begin{bmatrix} 0 & \vec{\nabla}^T \\ -\vec{\nabla} & 0 \end{bmatrix}. \tag{9.8}$$

Consequently, for an arbitrary $|\phi_0\rangle = \sum_x \phi(x,0)|x\rangle$ and $|\dot{\phi}_0\rangle \equiv \sum_x \frac{\partial\phi(x,0)}{\partial t}|x\rangle$ and from the Schrödinger equation

$$\frac{d}{dt}\begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix} = -i\begin{bmatrix} 0 & \vec{\nabla}^T \\ \vec{\nabla} & 0 \end{bmatrix}\begin{bmatrix} |\phi_V\rangle \\ |\phi_E\rangle \end{bmatrix},$$

that yields the follow relation

$$\frac{d}{dt}|\phi_V\rangle = -i\vec{\nabla}\cdot|\phi_E\rangle,$$

we one must prepare a corresponding initial quantum state that is a solution to

$$\begin{aligned} \phi_V &= \phi_0 \\ \vec{\nabla}\cdot\vec{\phi}_E &= i\frac{d}{dt}\phi_0, \end{aligned} \tag{9.9}$$

where we used the notation $\phi_0 = \phi(x,0)$. In more than one dimension, the equation (9.9) does not uniquely determine ϕ_E since $\vec{\nabla}\times\vec{\phi}_E$ is unspecified. (In one dimension ϕ_E is determined up to an additive constant.) In the remainder of this section we consider how to compute a solution to (9.9) and how to prepare the initial state $|\phi_V\rangle \oplus |\phi_E\rangle \in \mathscr{H}$ on a quantum computer for general cases of interest.

9.2.1 General Case

In the general case we may imagine that we are given efficient quantum circuits preparing the states $|\phi_0\rangle = \sum_x \phi(x,0)|x\rangle$ and $|\dot{\phi}_0\rangle = \sum_x \frac{\partial\phi(x,t)}{\partial t}|x\rangle$, (from now on we will write the equations only in terms of the amplitudes of $|\phi_V\rangle$ and $|\phi_E\rangle$.) The discrete analogue of (9.9) is, via our incidence-matrix discretization:

$$\phi_V = \phi_0 \tag{9.10}$$

$$-\frac{i}{h}B\phi_E = \dot{\phi}_0. \tag{9.11}$$

At second order and above, the solution to $\frac{i}{h}B\phi_E = \dot{\phi}_0$ is non-unique in general since the number of edges, $D(l/h)^D$, in the graph G_h exceeds the number of vertices $(l/h)^D$. Thus, the number of columns of B exceeds the number of rows by a factor of order D, the number of spatial dimensions. The non-unique solutions come from the basic concepts of linear algebra, since although $\dot{\phi}_0 \in \mathrm{Im}(B)$, where $B : \mathscr{H}_E \to \mathscr{H}_V$, B is not injective in general. One valid solution is to use as our quantum initial state

$$\begin{bmatrix} \phi_V \\ \phi_E \end{bmatrix} \propto \begin{bmatrix} \phi_0 \\ ihB^+\dot{\phi}_0 \end{bmatrix} \tag{9.12}$$

where B^+ denotes the Moore-Penrose pseudoinverse of the matrix B. A Moore-Penrose pseudoinverse has the property that the image of B^+ is the orthogonal complement of the kernel of B. Recall that B is a map from $\mathscr{H}_E \to \mathscr{H}_V$, which at first order (i.e. when B is the signed incidence matrix of a graph) and in the continuum limit can be interpreted as a divergence. The Helmholtz decomposition theorem says that any twice-differentiable vector field can be decomposed into a curl-free component and a divergence-free component, which means in this case

$$\begin{aligned} \vec{\phi}_E &= -\vec{\nabla}a_1(\vec{r}) + \vec{\nabla}\times\vec{a}_2(\vec{r}) \\ &= \vec{\phi}_{E_l} + \vec{\phi}_{E_t}, \end{aligned}$$

where $\vec{\phi}_{E_l}$ is the longitudinal component of $\vec{\phi}_{E_l}$ and $\vec{\phi}_{E_t}$ the transverse one. Thus, $\phi_E = -ihB^+\dot{\phi}_0$ corresponds in the continuum limit to the solution to the following system of equations.

$$\vec{\nabla}\cdot\vec{\phi}_E = -i\dot{\phi}_0 \tag{9.13}$$

$$\vec{\nabla}\times\vec{\phi}_E = 0, \tag{9.14}$$

since $-ihB^+\dot{\phi}_0$ corresponds the longitudinal term of $\vec{\phi}_E$. To construct the state (9.12) we can use the quantum linear systems algorithm of [16]. Specifically, we wish to prepare the state proportional to the solution to $Ax = b$ where

$$A = \begin{bmatrix} \mathbb{1} & 0 \\ 0 & ih^{-1}B \end{bmatrix} \tag{9.15}$$

$$b = \begin{bmatrix} \phi_0 \\ \dot{\phi}_0 \end{bmatrix} \tag{9.16}$$

since This can be done using the quantum linear systems algorithm of [16], which is a result that improved the runtime of the HHL algorithm, whose time complexity is $\widetilde{O}(\kappa)$, where κ is the condition number of A, which in this case is equal to the condition number of the incidence matrix B.

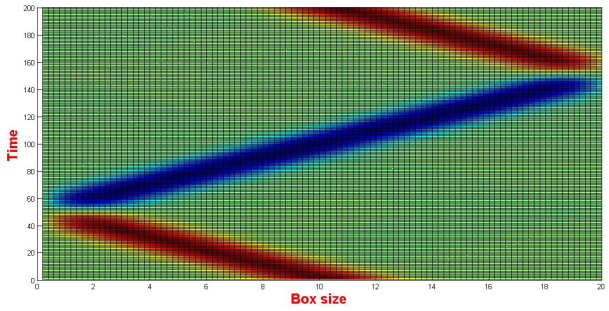
9.3 Numerical examples

The above analysis can be confirmed by numerical examples, as shown in this section. In all cases one sees that the dynamics and implementation of initial conditions and boundary conditions are consistent with theoretical expectations. Our quantum algorithm is implemented on a gate model quantum computer, and time evolution is discretized into a sequence of elementary gates via the method of [8]. The error induced by this time discretization is rigously upper bounded in [8]. Thus the focus of our numerical study is to investigate the errors induced by spatial discretization and verify the implementation of boundary conditions and initial conditions.

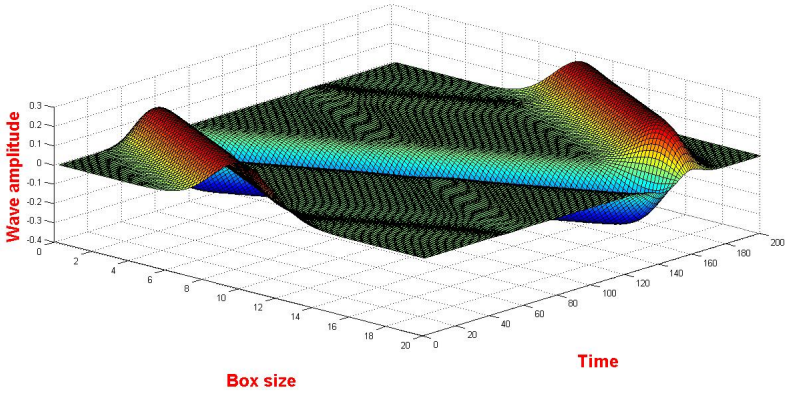
In our simulation we need to choose a sufficiently small timestep to ensure stability of the numerical method. We achieve this by taking

$$\Delta t < h,$$

in order to keep our numerical analysis stable, as we saw in chapter 4, [43]. In small examples we verified the accuracy of the numerical solution to the differential equations by comparing against direct computation of the entire unitary operator e^{-iHt} applied to the initial state vector.

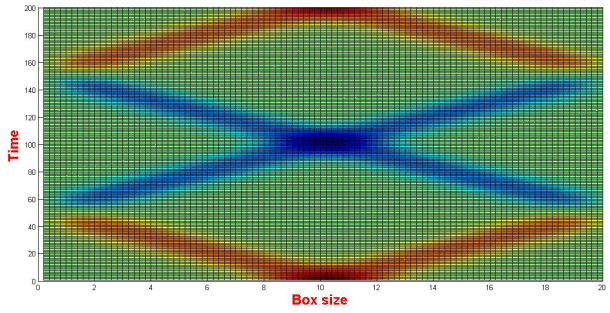


a)

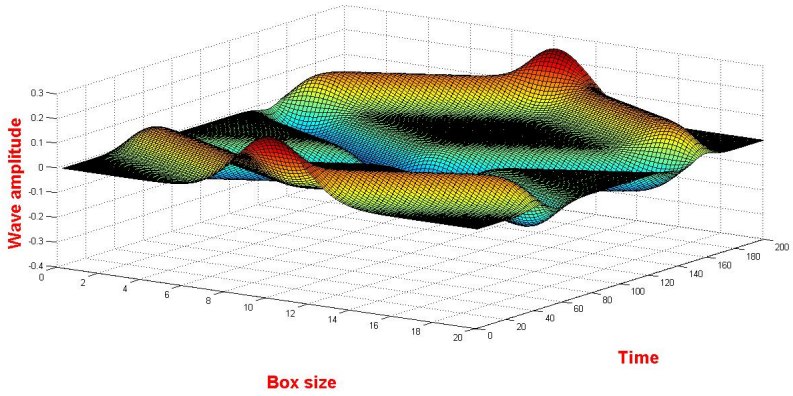


b)

Figure 9.2: **Shape preserving on line segment Dirichlet.** Here we consider the case of a rigidly-translating wavepacket. We can see two different views of the same wave packet starting in the middle point in a box with size 20, where space is represented by the x-axis while in the y-axis we have the time. We can see the packet going back and forward between the extremes of the box. Although its wave amplitude is preserved in time, when the wave packet arrives at the end points its amplitude in inverted with its propagation's direction. The red color gives us the positive amplitude against the blue one with negative value. In this example we choose lattice spacing $h = 0.2469$ and gaussian wavepacket width $\sigma = 1.6$, where σ plays the role of the standard deviation, when we think the gaussian wavepacket as the probability density function. Figures from [21].

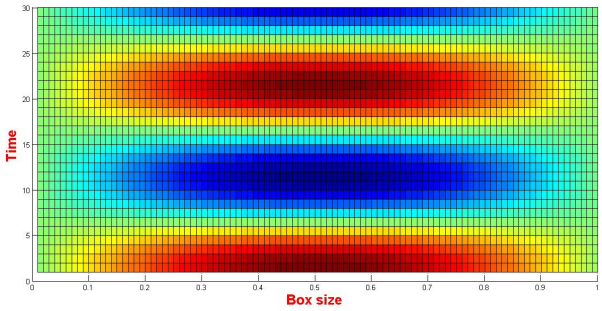


a)

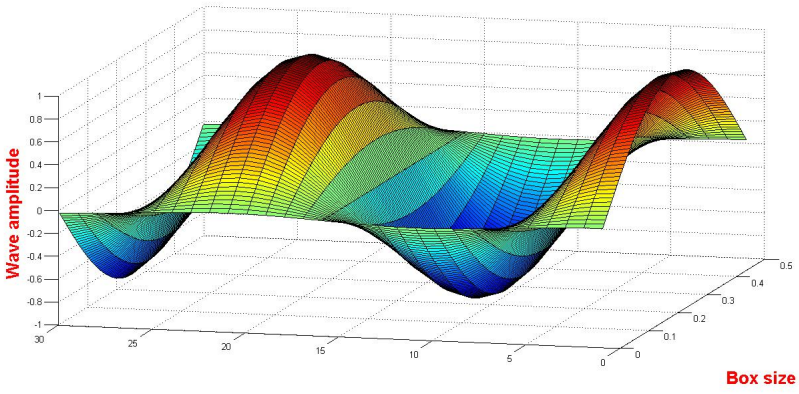


b)

Figure 9.3: **Spreading wave on line segment Dirichlet.** In these figures we kept with the same parameters used for the previous plots, changing only the initial condition for $\vec{\phi}_E$. Now we can see the wave spreading equally for both sides. Both waves amplitudes reflect in the boundary and then arrive in the lattice center at the same time, but with the amplitude inverted. Figures from [21].



a)



b)

Figure 9.4: **Standing wave.** Here we consider a standing wave, which can be described analytically by $\phi(x,t) = \cos(\omega t)\sin(\pi x)$. This can be simulated by Schrödinger's equation employing the follow initial conditions $\phi_V = \phi_0 = \sin(\pi x)$ and $\phi_E = d\phi_0/dt = 0$ as long as we start with $t = 0$.Figures from [21].

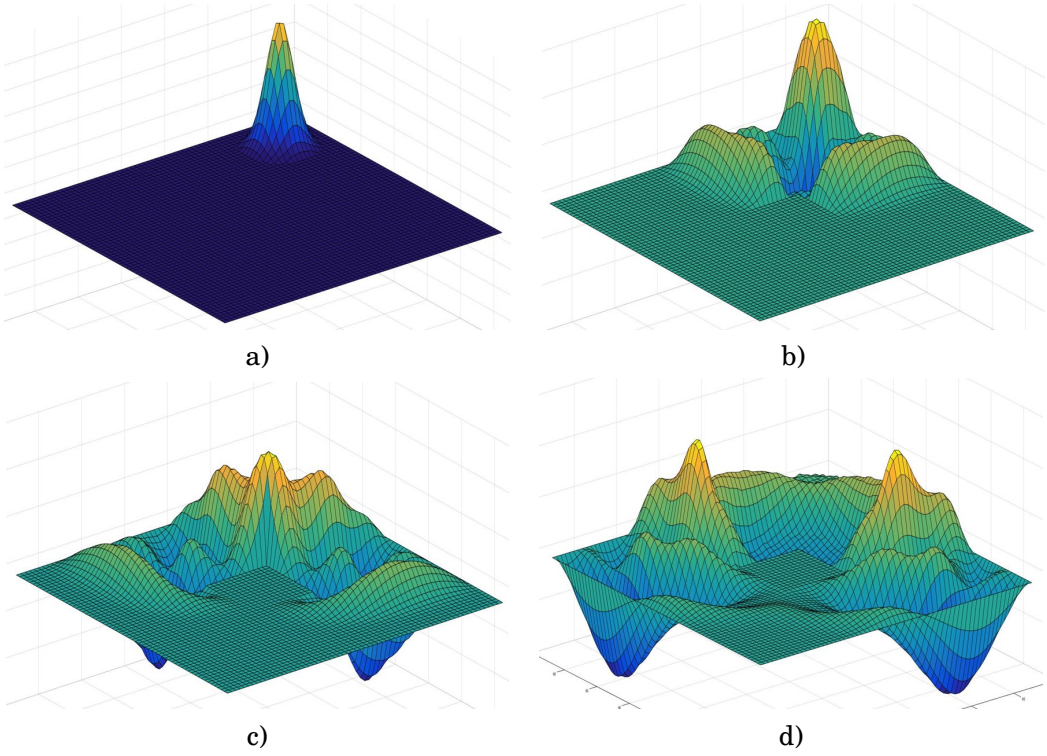


Figure 9.5: **Wave packet in a cavity.** Here the initial state is a Gaussian wave packet, but now in a two dimensional region with nontrivial boundary. Specifically, we simulate scattering of the wavepacket off a square object with Dirichlet boundary conditions. This is implemented as a square hole in the underlying discrete lattice. These four views represent the same wave packet in different time instants, where $t_a > t_b > t_c > t_d$. As in the one dimensional example, we worked with Dirichlet boundary conditions; however, the shape is not preserved. Here, the box has size ten in both axes, and we choose $h = 0.1563$ and $\sigma = 0.4$. Figures from [21].

9.4 Discretization Errors

Like we did in chapter 4, we investigate the behavior of our numerical simulations via the Q *factor*,

$$Q(t) = \frac{\left\|\phi^{4h} - \phi^{2h}\right\|_2}{\left\|\phi^{2h} - \phi^{h}\right\|_2}.$$

to quantify the discretization errors.

Differently from what we did in chapter 4 we worked with the concept of average Q from $t = 0$ to $t = 0.5$. Thus, we present these results via a table, where the parameter chosen were $h = 60$ and 0.0001 as the time step employed in the software.

$\langle Q \rangle_{\text{spreading}}$	3.98
$\langle Q \rangle_{\text{standing}}$	3.99

Then, from these values above we can conclude that our method is convergent and that the order of the Laplacian is h^2, Eq.(4.42).

9.5 Post-Processing

Since we are dealing with a quantum algorithm we expect to use it in a quantum computer, otherwise we will return to the same complexity class that the ordinary numerical methods for differential equations. Then, there is a fundamental question that we must address here: what is the success probability of we measure the desired output? Trying to answer this question is our central proposal in this chapter.

After we perform a Hamiltonian simulation from $t = 0$ to $t = T$ we are left with both states $\phi(T)$ and $B^{-1}d\phi(T)/dt$, where the first state lives in $\mathscr{H}_V$ and the last in $\mathscr{H}_E$. Then, depending on the application we may be interested in one or other state.

If our interest is to produce a state proportional to ϕ, we will show now that there is not any reasonable lower bound on success probability in this measurement, even for simple cases. Let us see this fact with a simple example. Suppose we have the following initial conditions $\phi_0 = \cos(x)$ and $\dot{\phi}_0 = 0$. Then in any other time the function can be written as $\phi(x,t) = f(t)\cos(x)$ for some f that oscillates between -1 and 1. Then in case we decide to measure our state after a time T where $f(T) = 0$, we will not see any support of this state in $\mathscr{H}_V$. Then, we can see how the post-processing depends on the initial state, thus, each case needs to be carefully analyzed.

In case we only care about a state proportional to $d\phi/dt$ we can do an estimate estimative. As before we begin measuring if the state is in $\mathscr{H}_V$ or $\mathscr{H}_E$, but now we are interested in the success probability of our state in $\mathscr{H}_E$. The resulting state is proportional to $B^{-1}d\phi/dt$, then we need to cancel B^{-1}, which is a easy task, since we only have to apply B and we can do after we get a estimation.

Inspired by [38] the procedure for matrix multiplication is

$$|B^{-1}d\phi/dt\rangle|0\rangle|0\rangle = \sum_j \alpha_j|\Lambda_j\rangle|0\rangle|0\rangle \tag{9.17}$$

$$\mapsto \sum_j \alpha_j|\Lambda_j\rangle|\tilde{\lambda}_j\rangle|0\rangle \tag{9.18}$$

$$\mapsto \sum_j \alpha_j|\Lambda_j\rangle|\tilde{\lambda}_j\rangle\left(\frac{\tilde{\lambda}_j}{C}|0\rangle + \frac{\sqrt{C^2-\tilde{\lambda}_j^2}}{C}|1\rangle\right). \tag{9.19}$$

Let us explain with more detail the procedure used above. At first we e-expresses the initial state in the eigenbasis $\{|\Lambda_j\rangle\}$. The idea here is to apply the well-known quantum algorithm to phase estimation [54] that gives the following map

$$\left|\Lambda_j\right\rangle|0\rangle \to \left|\Lambda_j\right\rangle\left|\tilde{\lambda}_j\right\rangle$$

where $\tilde{\lambda}_j$ is the binary representation of λ_j to a certain precision. In this step we have to apply the unitary $\exp(iHt)$ into the eigenstates $\left|\Lambda_j\right\rangle$ in order to get

$$e^{iHt}\left|\Lambda_j\right\rangle = e^{i\lambda_j t}\left|\Lambda_j\right\rangle.$$

Finally, in the third line we perform a controlled rotation of the second qubit. This rotation is a multiplexed gates rotation

$$R\left(|0\rangle_1|x\rangle_{n-1}\right) = R_x|0\rangle_1|x\rangle_{n-1},$$

where the subscript means the number of qubits related with the state. In our case we have

$$R_j = e^{-i\theta_j\sigma_y} = \begin{pmatrix}\cos\theta_j & -\sin\theta_j\\ \sin\theta_j & \cos\theta_j\end{pmatrix},$$

where $\theta_j = \arccos\left(\tilde{\lambda}_j/C\right)$. C is a constant of normalization and must satisfy $C \geq \sqrt{||L||}$ so that the argument under the square root is not negative. Setting it to $\Theta(\sqrt{||L||})$, the probability of measuring the last qubit in $|0\rangle$ is $\kappa(L)^{-2}$ in the worst case. Then we produce a state proportional to $d\phi(T)/dt$ conditioned on measuring the last qubit in the state $|0\rangle$.

9.6 Klein-Gordon Equation

It is quite well known that if we want to describe phenomena at high energies we need go to relativistic theories. In particular, we are interested in relativistic quantum mechanics theory since there are a group of particles, spinless particles, that are described by relativistic wave equation.

When we move to the high energy physics the kinetic energy of the particle is in the same energy scale of its mass energy, mc^2. In this case we need to include the mass term in the total

energy of the system, where in the end it will change the particle equation format, as we will see now. From elementary quantum mechanics we know that Schrödinger equation

$$i\hbar\frac{\partial\phi(x,t)}{\partial t}=\left[-\frac{\hbar^2}{2m}\nabla^2+V(x)\right]\phi(x,t),$$

where m is the particle mass and V the potential energy. This equation corresponds to a nonrelativistic equation. This equation has the follow operation form

$$\hat{E}=\frac{\hat{p}^2}{2m}+V(x),$$

where,

$$\hat{E}=i\hbar\frac{\partial}{\partial t},\quad \hat{p}=-i\hbar\nabla.$$

The idea here is to show how we can modify this equation to get a relativistic wave equation. There is a well known relativistic relation that can be our starting point

$$p^\mu p_\mu=\frac{E^2}{c^2}-\vec{p}\cdot\vec{p}=m^2c^2, \tag{9.20}$$

where p^μ with $\mu=0,1,2,3$ is the four-momentum of the particle

$$p^\mu=\{p^0,p^1,p^2,p^3\}=\{E/c,p_x,p_y,p_z\},$$

$$p_\mu=\{E/c,-p_x,-p_y,-p_z\},$$

where E is the total energy for the free particle and c is the speed of light. $p\cdot p=p^\mu p_\mu$ is the scalar product in four dimensions

$$p^\mu p_\mu=p^0p_0+p^1p_1+p^2p_2+p^3p_3.$$

Likewise the scalar product is invariant by rotations in Euclidean space, this scalar product in four dimension is invariant by Lorentz transformation in Minkowski space [2]. In the end, relation (9.20) is saying that the mass is a relativistic invariant of the particle.

Returning to Eq.(9.20), we can use the vectorial form of the four-momentum p^μ

$$\hat{p}^\mu=i\hbar\frac{\partial}{\partial x_\mu}=i\hbar\left\{\frac{\partial}{\partial t},-\nabla\right\},$$

$$\hat{p}_\mu=i\hbar\frac{\partial}{\partial x^\mu}=i\hbar\left\{\frac{\partial}{\partial t},\nabla\right\}.$$

Thus, we obtain the **Klein-Gordon equation** for free particles,

$$\hat{p}^\mu\hat{p}_\mu\phi=m^2c^2\phi,$$

[2] Minkowski space is a 4-dimensional real vector space equipped with a nondegenerate, symmetric bilinear form on the tangent space at each point in spacetime, here simply called the Minkowski inner product, with metric signature either $(-,+,+,+)$ or $(+,-,-,-)$.

or

$$\frac{1}{c^2}\frac{\partial^2\phi}{\partial t^2} - \nabla^2\phi + \frac{m^2c^2}{\hbar^2}\phi = 0. \tag{9.21}$$

From the equation above we can see that we are dealing with a wave equation with an extra factor. As we will see now our algorithm can easily contemplate this extra factor by just adding self-loops on the graph G_h. From now on we will go back to natural units, that implies both $\hbar$ and c are equal to one.

Suppose we have a graph G', from where we can construct the graph Laplacian that approximates,

$$\frac{1}{h^2}L\left(G'\right) = \nabla^2\phi + m^2\phi,$$

thus

$$\frac{\partial^2\phi}{\partial t^2} = \frac{1}{h^2}L\left(G'\right)\phi,$$

is the discretized version of Eq.(9.21). It means that our Laplacian has the whole information about the particle, which includes its mass term. In fact this graph G' can be easily achieved from a graph G that approximates the following operator

$$\frac{1}{h^2}L\left(G\right) = \nabla^2\phi,$$

that gives our ordinary wave equation, which means $L\left(G\right)$ does not have a mass term.

Beginning with G the mass term can be established by adding self loops with $W = (hm)^2$ as its weight on all vertices of G, see chapter 4 section 4.2.2. This manipulated graph is our graph G'. Finally, like before, we need to construct its incidence matrix $B\left(G'\right)$ in order to get the Laplacian,

$$B\left(G'\right)^\dagger B\left(G'\right) = L\left(G'\right).$$

Moreover, it is easy to see how this Laplacian $L(G')$ is related with the Laplacian from G, $L(G)$

$$L\left(G'\right) = L\left(G\right) + h^2m^2I,$$

where I is the identity matrix. Therefore, whereas $B(G)$ gives our ordinary wave equation, applying $B\left(G'\right)$ in our Hamiltonian gives our relativistic wave equation.

9.7 Maxwell's Equations

As a final application, we can see how our algorithm can deal with Maxwell's equation.

We are aware that in a region with no charges and no currents, such as in a vacuum, Maxwell's equations governing the time evolution of electric $\vec{E}$ and magnetic fields $\vec{B}$ take the form

$$\frac{\partial\vec{E}}{\partial t} = \nabla\times\vec{B}, \quad \frac{\partial\vec{B}}{\partial t} = -\nabla\times\vec{E}. \tag{9.22}$$

Taking the second derivative with respect to time in the left equation above yields,

$$\frac{\partial^2 \vec{E}}{\partial t^2} = \nabla \times \frac{\partial \vec{B}}{\partial t}.$$

Now, from the right equation in (9.22) we can replace $\partial \vec{B}/\partial t$ in the last equation to get

$$\frac{\partial^2 \vec{E}}{\partial t^2} = \nabla \times \nabla \times \vec{E}.$$

As the next step we can make use of the following relation

$$\nabla \times \nabla \times a = \nabla(\nabla \cdot a) - \nabla^2 a,$$

and use the fact that we do not have extra sources, that means $\nabla \cdot \vec{E} = 0$ to get in the end

$$\frac{\partial^2 \vec{E}}{\partial t^2} = -\nabla^2 E.$$

which implies that $\vec{E}$ follows the wave equation. An analogous calculation can be done for $\vec{B}$ to see that it also obeys the wave equation. From our Hamiltonian simulation method for the wave equation, it is clear that we can simulate each field separately. However, we will show an alternative method, also via Hamiltonian simulation, that can simulate both fields simultaneously and thus the electromagnetic field in vacuum.

If we consider discretizing space, then we can write Eqs.(9.22) as

$$\frac{\partial}{\partial t}\begin{bmatrix}\vec{E}\\ \vec{B}\end{bmatrix} = \begin{bmatrix}0 & C\\ -C & 0\end{bmatrix}\begin{bmatrix}\vec{E}\\ \vec{B}\end{bmatrix}$$

where C is the finite difference approximation of the curl operator. To see how to construct C, consider the following

$$\nabla \times \begin{bmatrix}a\\ b\\ c\end{bmatrix} = \begin{bmatrix}\partial c/\partial y - \partial b/\partial z\\ \partial a/\partial z - \partial c/\partial x\\ \partial b/\partial x - \partial a/\partial y\end{bmatrix} = \begin{bmatrix}0 & -\partial/\partial z & \partial/\partial y\\ \partial/\partial z & 0 & -\partial/\partial x\\ -\partial/\partial y & \partial/\partial x & 0\end{bmatrix}\begin{bmatrix}a\\ b\\ c\end{bmatrix}.$$

This suggests we should consider the linear differential equation

$$\frac{\partial}{\partial t}\begin{bmatrix}E_x\\ E_y\\ E_z\\ B_x\\ B_y\\ B_z\end{bmatrix} = \begin{bmatrix}0 & 0 & 0 & 0 & -\partial/\partial z & \partial/\partial y\\ 0 & 0 & 0 & \partial/\partial z & 0 & -\partial/\partial x\\ 0 & 0 & 0 & -\partial/\partial y & \partial/\partial x & 0\\ 0 & \partial/\partial z & -\partial/\partial y & 0 & 0 & 0\\ -\partial/\partial z & 0 & \partial/\partial x & 0 & 0 & 0\\ \partial/\partial y & -\partial/\partial x & 0 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}E_x\\ E_y\\ E_z\\ B_x\\ B_y\\ B_z\end{bmatrix} \tag{9.23}$$

Now we can discretize space into a uniform cubic lattice, since we are in a three-dimensional lattice, and approximate the differential operators using finite difference methods, chapter 4, to

simulate the EM field via Hamiltonian simulations, as we did in the previous examples. We should notice that in this case, unitarity translates to conservation of the classical energy contained in the field $\int_V |\vec{E}(\vec{x})|^2 + |\vec{B}(\vec{x})|^2$.

9.8 Final considerations

Here we presented an extremely specialized quantum algorithm for simulating the wave equation problem. We got an exponential speed up in terms of the lattice dimension D compared with the ordinary numerical methods for differential equations. Moreover, even contrasting our result with analogous quantum results, our algorithm is more efficient. For instance we obtained a quadratic speed up for the state preparation compared with [16].

In contrast with previous quantum algorithms, here we do not only proposed the method, we also saw basic examples of the wave equation simulation via Hamiltonian. Thus, one is first expectation is seeing this algorithm being applied in more complicated cases for different proposals. Moreover, we hope this algorithm can be implemented in some quantum computer in a near future.

There are few directions that we are considering to go next. We want to investigate the performance of quantum algorithms for simulating the wave equation based on finite element methods [91], rather than finite difference methods, as considered here. Finally, we are interest in extending this algorithm to more complicated wave equations, as non linear cases [73].

CHAPTER 10

CONCLUSION AND PERSPECTIVES

10.1 Conclusions

In general, when we want to propose new theories, we try to use all our knowledge about the ones well establish to try to build new models consistently. We see it very often when we want to go from classical physics to quantum and relativistic theories. In general, after we move to new models we usually attempt to recover the predictions of the classical theory, the theory that we have more access and knowledge. For the quantum models of computation, this is not different. Commonly we try to use what worked for classical models of computation to build the quantum ones. It was exactly the case of the quantum walks, which emerged from random walks, and quantum cellular automata that emerged from classical cellular automata, where the main idea is to try to use these quantum tools to explore quantum phenomena. In this thesis we tried to put this issue in evidence when we separate it into two parts, classical, part I, and quantum models of computation, part II.

Throughout this thesis, we gave emphasis to the partitioned cellular automata and its quantum counterpart. In part I, we showed the strength of this model to deal with both classical physics and classical model of computation as to motivate the use of its quantum counterpart to quantum physics and quantum model of computation. Despite the fact that in the classical models of computation our focus was to show already established results, we also started new investigations in the partitioned cellular automata, proposing a coarse-graining technique, chapter 5, for this CA class. In the same spirit of the analogous result for the coarse-graining to elementary CA [39], we investigated emergent processes from a more physical perspective. We can say that until our last analysis the core result was the emergence of stochastic process from deterministic ones, pointing to us why it is so natural to work with non-deterministic process

in nature. Moving to part II, in chapter 6 we brought the partitioned unitary quantum cellular (PUQCA), our definition to QCA that we believe to be a more clear definition compared with the previous ones. Subsequently, in chapter 7, we showed the main result of this thesis. Working with the PUQCA we could translate the main quantum walk classes into QCA. We expect that these results will boost the use of QCA in physics and other areas since the QCA is more experimentally friendly and matches with the quantum architecture used in the quantum computers available nowadays. In part II, we also started the investigation of the model that we called quantum HPP (QHPP), chapter 8. The QHPP model is a topic closely related with the QCA, since the HPP rule [37], the first gas model proposed, is described in terms of partitioned cellular automata. However, we used the quantum walk model to encode the QHPP. In chapter 8 we initiated the study of the QHPP with two quantum walkers using a two-dimensional lattice. Until the moment we only investigated numerically the new dynamics generated by two interacting quantum walkers, following the HPP as the collision rule. Moreover, we also investigated the entanglement dynamics between the particles. In this part, we could observe how the entanglement between them evolves with time and how it depends on the initial condition. Another important result presented in part II, chapter 9, was the quantum algorithm for simulating the wave equation, a result with a proposal distinct from the others. Despite the fact that the wave equation is widely applied in physics in different contexts, our interests were quite different from the others, since our main concern was to use a quantum computer to get a more efficient numerical method, in terms of the number of computations, to the wave equation. Although we achieved our results via Hamiltonian simulation, in the first moment we do not expect to learn more about physics with this algorithm, but to show that quantum computers can do the same task, in our case solve the wave equation numerically, more efficiently. We just can say that we will be able to learn more about physics with this result when we use it to simulate larger systems, that requires a large amount of memory, not possible to classical computers.

10.2 Perspectives

All the works presented in this thesis could be continued in several directions. For instance, for the one introduced in chapter 5, coarse-graining of PCA, the natural next step is to try to convert the CG technique to the quantum regime, namely to the PUQCA. Doing that we want to explore emergence under quantum perspective. In this part we can do, for instance, a similar study to the one done in chapter 5. We can try to get effective dynamics after we coarse grain some quantum phenomenon described via QCA. In this point we certainly will make use of our results of chapter 7, to translate several dynamics described in terms of quantum walks, like QWs to quantum field theories, to see what dynamics will emerge from them. Another interesting study that can be done after we construct the CG tool for the PUQCA is the study of quantum-classical transition, a very rich topic with several questions to explore. Moving to our results shown in chapter 8,

QHPP, there are several directions to take. Our next step is to try use it for quantum search algorithm, to see if will get some speed up compared with the models of one walker. In order to do that we should be able to explore these dynamics analytically. Furthermore, we hope to convert this model in terms of QCA. Given that, we can try to study this model with more particles and also try to use different lattice structures, like the one used in the FHP model (a more realistic model applied to the study real fluids).

Furthermore, we expect to see more and more quantum computers available employing more qubits. As this happens, we hope to use our results in these quantum machines. For example, we can try to implement our quantum algorithm to wave equation and see how it will behave in realistic quantum devices. Moreover, we can use the PUQCA to implement several quantum search algorithms in these computers to confirm the predictions estimated from these models.

APPENDIX A

ASYMPTOTIC NOTATIONS

When we are interested in to analyze the complexity of some algorithm, which is concern about how fast or slow is the performs of the algorithm, we are interested in estimates its performs asymptotically. There are various cases that we can only estimate bounds, for instance, upper or lower bounds, about the algorithm performs. The proposal here is just to introduce the notation used for these complexity estimations.

In order to introduce these notations we give two functions $s(n), t(n)$ on $\mathbb{N}$. Moreover, we say that two functions are **asymptotically equivalent** when

$$\lim_{n\to\infty} \frac{s(n)}{t(n)} = 1.$$

Thus, we write

- $s(n) = O(t(n))$ if there are constants c, d such that for all n,

$$s(n) \leq c \cdot t(n) + d$$

- $s(n) = \Omega(t(n))$ if $t(n) = O(s(n))$, and
- $s(n) = \Theta(t(n))$ if $s(n) = O(t(n))$, and $s(n) = \Omega(t(n))$.

In the first case, we say $s(n)$ is "Big-Oh-of" $t(n)$, whereas in the second, we might say $t(n)$ is asymptotically bounded below by $s(n)$, and in the third, we say $s(n)$ and $t(n)$ have the same "asymptotic order".

APPENDIX B

CONDITIONING NUMBER

Suppose that we have a linear system to solve

$$Ax = b,$$

and you are interest in know how a small change in b,

$$\hat{b} - b = \delta b,$$

or in A will influence the solution x,

$$\hat{x} - x = \delta x,$$

where $\hat{x}$ and $\hat{b}$ satisfies $A\hat{x} = \hat{b}$. Here our focus is the changing in b. The quantity that measures this perturbation in x, but in terms of of relative errors of b

$$\frac{\|\delta b\|}{\|b\|}, \tag{B.1}$$

and relative errors of x

$$\frac{\|\delta x\|}{\|x\|}, \tag{B.2}$$

is known as **condition number**. For linear systems this quantity depends on the matrix norm of A as follows

$$\kappa(A) = \|A\| \left\|A^{-1}\right\|, \tag{B.3}$$

and we call it by condition number of the matrix A. From Eq.(B.3) we can see that this quantity changes accordingly the norm definition employed, but the meaning will be always the same. Let us understand how this expression appears and see its format in terms of spectral norm. The best way to start our analysis is with a review of vector and matrix norms.

B.1 Vector norms

A **vector norm** $\|x\|$ measures the size of a vector $x \in \mathbb{R}^n$ by a nonnegative number and has the following properties

$$\|x\| \geq 0,\ \|x\| = 0 \Rightarrow x = 0,$$

$$\|\alpha x\| = |\alpha|\,\|x\|,$$

$$\|x + y\| \leq \|x\| + \|y\|,$$

for any $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$. From these three properties we can define many possible vectors form, for instance

$$\begin{aligned} \|x\|_1 &= |x_1| + \ldots + |x_n|, \\ \|x\|_2 &= \left(|x_1|^2 + \ldots + |x_n|^2\right)^{1/2}, \\ \|x\|_\infty &= \max\{|x_1| + \ldots + |x_n|\}. \end{aligned}$$

If we did not writhe any subscript in $\|x\|$, then the equation is valid for all three norms.

If the exact vector is x and the approximation is $\hat{x}$ we can define the relative error with respect to a vector norm as

$$\frac{\|\hat{x} - x\|}{\|x\|}.$$

B.2 Matrix norms

An $m \times n$ matrix can be considered a particular kind of vector $x = A \in \mathbb{R}^{m \times n}$ and like the vector norm the **matrix norm** gives us a real number $\|\cdot\| : \mathbb{R}^{m \times n} \to \mathbb{R}$. The same three properties showed to vector norm is valid here

$$\|A\| \geq 0,\ \|A\| = 0 \Rightarrow x = 0,$$

$$\|\alpha A\| = |\alpha|\,\|A\|,$$

$$\|A + B\| \leq \|A\| + \|B\|,$$

for any $A, B \in \mathbb{R}^{m \times n}$ and $\alpha \in \mathbb{R}$. There are more two additional properties for matrix norm that are not required of all matrix norm, which are subordinance,

$$\|Ax\| \leq \|A\|\,\|x\|, \tag{B.4}$$

and submultiplicativity

$$\|AB\| \leq \|A\|\,\|B\|.$$

For our proposal we are particularly interest in the **Induced** or **operator norms**, where the matrix norm of A is based on any vector norm $\|x\|$ as follows

$$\|A\| := \sup_{\substack{x \in \mathbb{R}^n \\ x \neq 0}} \frac{\|Ax\|}{\|x\|} = \max_{\substack{x \in \mathbb{R}^n \\ \|x\| = 1}} \|Ax\|. \tag{B.5}$$

We can now show without difficult that the operator norm satisfy all properties given above.

1. $\|A\| > 0$ if $A \neq 0$ this part is trivial and obvious from the operator norm definition;

2. $\|\alpha A\| = |\alpha| \, \|A\|$

$$\|\alpha A\| = \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|\alpha A x\| = |\alpha| \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|Ax\| = |\alpha| \, \|A\| ;$$

3. $\|A + B\| \leq \|A\| + \|B\|$

$$\|A + B\| = \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|(A + B)x\| = \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|Ax + Bx\| ,$$

now we can apply the triangular inequality

$$\leq \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \{\|Ax\| + \|Bx\|\} \leq \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|Ax\| + \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|Bx\| = \|A\| + \|B\| ;$$

4. $\|Ax\| \leq \|A\| \, \|x\|$

$$\|A\| \, \|x\| = \sup_{\substack{y \in \mathbb{R}^n \\ \|y\| \neq 0}} \frac{\|Ay\|}{\|y\|} \|x\| \leq \frac{\|Ay\|}{\|y\|} \|x\| ,$$

As y is arbitrary, we let $y = x$ and get

$$\|A\| \, \|x\| \leq \|Ax\| ;$$

5. $||AB|| \leq ||A|| \, ||B||$

$$\|AB\| = \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|(AB)x\| \leq \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \{\|A\| \, \|Bx\|\} = \|A\| \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|Bx\| = \|A\| \, \|B\| ,$$

where first we used the inequality 4 and after inequality 1.

Now we can see the operator norm for two especial cases, first form the one norm $\|A\|_1$ and then for the spectral norm $\|A\|_2$. From the vector norm we know that the one norm is given by the sum of all absolute values components from a given vector. Thus, in case of $\|A\|_1$ we first do a matrix multiplication,

$$Ax = \sum_j a_{ij} x_j$$

where a_{ij} are the matrix elements of the matrix A. Now, we use the 1-norm definition,

$$\|Ax\|_1 = \sum_i \left| \sum_j a_{ij} x_j \right| .$$

As the next step we can use the follow inequality,

$$\sum_i \left| \sum_j a_{ij} x_j \right| \leq \sum_j \left(\sum_i |a_{ij}| \right) |x_j| ,$$

and also

$$\sum_j \left(\sum_i \left| a_{ij} \right| \right) \left| x_j \right| \leq \sum_j \left(\max_j \sum_i \left| a_{ij} \right| \right) \left| x_j \right|,$$

where this last inequality comes from the fact that we changed all different row sums $\sum_j \sum_i \left| a_{ij} \right|$ to the ones where it assumes the maximum values, then

$$\sum_j \left(\max_j \sum_i \left| a_{ij} \right| \right) \left| x_j \right| = \left(\max_j \sum_i \left| a_{ij} \right| \right) \| x \|_1 .$$

Thus,

$$\| Ax \|_1 \leq \left(\max_j \sum_i \left| a_{ij} \right| \right) \| x \|_1 .$$

Finally, we can use the one norm definition to get

$$\| A \|_1 = \max_{\substack{x \in \mathbb{R}^n \\ \| x \| = 1}} \| Ax \|_1 = \max_j \sum_i \left| a_{ij} \right| .$$

Now we can see the operator norm for the spectral norm

$$\| x \|_2 = \left(x^\dagger x \right)^{1/2} .$$

From the spectral vector norm above we have

$$\| Ax \|_2 = \left(x^\dagger A^\dagger A x \right)^{1/2} . \tag{B.6}$$

Now we can use the fact that the matrix $A^\dagger A$ is a hermitian matrix and then use the eigen-decomposition of $A^\dagger A$,

$$A^\dagger A = V D V^\dagger ,$$

where

$$D = diag(\sigma_1, \ldots, \sigma_n), \quad \text{and} \quad U = \left(\phi_1, \ldots, \phi_n \right),$$

are the diagonal eigenvalue matrix and the eigenvector matrix of $A^\dagger A$, satisfying

$$A^\dagger A \phi_i = \sigma_i \phi_i ,$$

for $i = 1, \ldots, n$. Now we can return to Eq.(B.6) and use the eigen-decomposition to get

$$\| Ax \|_2 = \left(x^\dagger U D U^\dagger x \right)^{1/2} = \left(\left(U^\dagger x \right)^\dagger D \left(U^\dagger x \right) \right)^{1/2} = \left(y^\dagger D y \right)^{1/2} ,$$

where $y = U^\dagger x$. Thus, we can rewrite the equality above in terms of its elements

$$\| Ax \|_2 = \left(\sum_{i=1}^{n} \sigma_i y_i^2 \right)^{1/2} .$$

As $A^\dagger A$ is a symmetric positive definite square matrix, all of its eigenvalues are real and positive and assumed to be sorted

$$0 \leq \sigma_1 \leq \ldots \leq \sigma_n = \sigma_{max}.$$

Moreover, the new vector $y = U^\dagger x$ can be considered as a rotated version of x with its Euclidean 2-norm conserved, $||y||_2 = ||x||_2$. Now, from the fact that σ_{max} is the maximum eigenvalue of $A^\dagger A$ the follow inequality is obvious

$$\left(\sum_{i=1}^{n} \sigma_i y_i^2\right)^{1/2} \leq \left(\sigma_{max} \sum_{i=1}^{n} y_i^2\right)^{1/2} = \sqrt{\sigma_{max}}\, \|y\|_2 .$$

Therefore, from the operator norm definition

$$\|A\|_2 = \max_{\substack{x \in \mathbb{R}^n \\ \|x\|=1}} \|Ax\|_2 = \sqrt{\sigma_{max}}.$$

In case we have A as a hermitian matrix we get that $\sigma_i = \lambda_i^2$ with $i = 1, \ldots, n$, where λ_i are the eigenvalues of A and σ_i the eigenvalues of $A^\dagger A$.

Now we are ready to return to the condition number issue.

B.3 Condition numbers for linear systems

Returning to our initial question, where we have a linear system to solve $Ax = b$ and we want to know how the relative error Eq.(B.1) influences the relative error Eq.(B.2). We have that

$$A(\hat{x} - x) = \hat{b} - b.$$

Then, it is clear that

$$\|\hat{x} - x\| = \left\|A^{-1}\left(\hat{b} - b\right)\right\|.$$

Now we can make use of the inequality Eq.(B.4) to get

$$\|\hat{x} - x\| \leq \left\|A^{-1}\right\| \left\|\hat{b} - b\right\|.$$

Similarly, it is a straightforward calculation to show that

$$\frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|},$$

that yields

$$\frac{\|\delta x\|}{\|x\|} \leq \|A\| \left\|A^{-1}\right\| \frac{\|\delta b\|}{\|b\|}, \tag{B.7}$$

where the number

$$\kappa(A) = \|A\| \left\|A^{-1}\right\|,$$

is called condition number of the matrix A. It determines how much the hand side of Eq.(B.7) can be amplified. Now, it is easy to see that for the spectral norm, in case we have employed the operator norm, the condition number of the matrix A takes the following form

$$\kappa(A) = \frac{\lambda_{max}}{\lambda_{min}},$$

where λ_{max} and λ_{min} are the highest and lowest eigenvalues, respectively when A is a Hermitian matrix.

BIBLIOGRAPHY

[1] Y. AHARONOV, L. DAVIDOVICH, AND N. ZAGURY, *Quantum random walks*, Phys. Rev. A, 48 (1993), pp. 1687–1690.

[2] M. ANNABESTANI, M. R. ABOLHASANI, AND G. ABAL, *Asymptotic entanglement in 2d quantum walks*, Journal of Physics A: Mathematical and Theoretical, 43 (2010), p. 075301.

[3] P. ARRIGHI, S. FACCHINI, AND M. FORETS, *Quantum walking in curved spacetime*, Quantum Information Processing, 15 (2016), pp. 3467–3486.

[4] P. ARRIGHI AND J. GRATTAGE, *Partitioned quantum cellular automata are intrinsically universal*, Natural Computing, 11 (2012), pp. 13–22.

[5] G. K. BATCHELOR, *The effect of brownian motion on the bulk stress in a suspension of spherical particles*, Journal of Fluid Mechanics, 83 (1977), p. 97–117.

[6] H. BECHMANN-PASQUINUCCI AND A. PERES, *Quantum cryptography with 3-state systems*, Physical Review Letters, 85 (2000).

[7] D. W. BERRY, *High-order quantum algorithm for solving linear differential equations*, Journal of Physics A: Mathematical and Theoretical, 47 (2014), p. 105301.

[8] D. W. BERRY, A. M. CHILDS, AND R. KOTHARI, *Hamiltonian simulation with nearly optimal dependence on all parameters*, (2015), pp. 792–809.

[9] S. D. BERRY AND J. B. WANG, *Two-particle quantum walks: Entanglement and graph isomorphism testing*, Physical Review A, 83 (2011).

[10] S. D. BERRY AND J. B. WANG, *Two-particle quantum walks: Entanglement and graph isomorphism testing*, Phys. Rev. A, 83 (2011), p. 042317.

[11] A. BISIO, G. M. D'ARIANO, AND P. PERINOTTI, *Quantum walks, weyl equation and the lorentz group*, Foundations of Physics, 47 (2017), pp. 1065–1076.

[12] S. BOIXO, T. F. RØNNOW, S. V. ISAKOV, Z. WANG, D. WECKER, D. A. LIDAR, J. M. MARTINIS, AND M. TROYER, *Evidence for quantum annealing with more than one hundred qubits*, Nature Physics, 10 (2014), p. 218.

[13] H. BOUGROURA, H. AISSAOUI, N. CHANCELLOR, AND V. KENDON, *Quantum-walk transport properties on graphene structures*, Phys. Rev. A, 94 (2016), p. 062331.

[14] A. M. CHILDS, *Universal computation by quantum walk*, Phys. Rev. Lett., 102 (2009), p. 180501.

[15] A. M. CHILDS, D. GOSSET, AND Z. WEBB, *Universal computation by multi-particle quantum walk*, Science, 339 (2013), pp. 791–794.

[16] A. M. CHILDS, R. KOTHARI, AND R. D. SOMMA, *Quantum linear systems algorithm with exponentially improved dependence on precision*, arXiv:1511.02306, (2015).

[17] B. CHOPARD AND M. DROZ, *Cellular Automata Modeling of Physical Systems*, Cambridge University Press, 2005.

[18] M. W. CHOPTUIK, *Lectures for vii mexican school on gravitation and mathematical physics; relativistic and numerical relativity; numerical analysis for numerical relativists*. University of British Columbia, 2009.

[19] F. R. K. CHUNG, *Spectral Graph Theory*, no. 92, Conference Board of the Mathematical Sciences, 1994.

[20] B. D. CLADER, B. C. JACOBS, AND C. R. SPROUSE, *Preconditioned quantum linear system algorithm*, Phys. Rev. Lett., 110 (2013), p. 250504.

[21] P. C. COSTA, S. JORDAN, AND A. OSTRANDER, *Quantum algorithm for simulating the wave equation*, arXiv:quant-ph/1711.05394, (2017).

[22] P. C. S. COSTA, R. PORTUGAL, AND F. DE MELO, *Quantum walks via quantum cellular automata*, Quantum Information Processing, 17 (2018), p. 226.

[23] G. M. D'ARIANO AND P. PERINOTTI, *Quantum cellular automata and free quantum field theory*, Frontiers of Physics, 12 (2016), p. 120301.

[24] C. DETRAIN AND J.-L. DENEUBOURG, *Self-organized structures in a superorganism: do ants "behave" like molecules?*, Physics of Life Reviews 3, (2006), pp. 162–187.

[25] C. DI FRANCO, M. MC GETTRICK, T. MACHIDA, AND T. BUSCH, *Alternate two-dimensional quantum walk with a single-qubit coin*, Phys. Rev. A, 84 (2011), p. 042337.

[26] G. DI MOLFETTA, M. BRACHET, AND F. DEBBASCH, *Quantum walks as massless dirac fermions in curved space-time*, Phys. Rev. A, 88 (2013), p. 042301.

[27] W. DÜR, R. RAUSSENDORF, V. M. KENDON, AND H.-J. BRIEGEL, *Quantum walks in optical lattices*, Phys. Rev. A, 66 (2002), p. 052319.

[28] A. EKERT AND P. L. KNIGHT, *Entangled quantum systems and the schmidt decomposition*, American Journal of Physics, 63 (1995).

[29] L. EULER, *Institutionales calculi integrate*, Saint Petersburg, (1768).

[30] R. P. FEYNMAN, *Simulating physics with computers*, International Journal of Theoretical Physics, 21 (1982), pp. 467–488.

[31] B. FOXEN, J. Y. MUTUS, E. LUCERO, R. GRAFF, A. MEGRANT, Y. CHEN, C. QUINTANA, B. BURKETT, J. KELLY, E. JEFFREY, Y. YANG, A. YU, K. ARYA, R. BARENDS, Z. CHEN, B. CHIARO, A. DUNSWORTH, A. FOWLER, C. GIDNEY, M. GIUSTINA, T. HUANG, P. KLIMOV, M. NEELEY, C. NEILL, P. ROUSHAN, D. SANK, A. VAINSENCHER, J. WENNER, T. C. WHITE, AND J. M. MARTINIS, *Qubit compatible superconducting interconnects*, Quantum Science and Technology, 3 (2018), p. 014005.

[32] U. FRISCH, B. HASSLACHER, AND Y. POMEAU, *Lattice-gas automata for the navier-stokes equation*, Phys. Rev. Lett., 56 (1986), pp. 1505–1508.

[33] M. GARDNER, *Mathematical games: The fantastic combinations of john conway's new solitaire game "life"*, Scientific American, 223 (1970), pp. 120–123.

[34] P. D. M. GIUSEPPE, *Discrete time quantum walks: from synthetic gauge fields to spontaneous equilibration*, PhD thesis, Université Pierre et Marie Curie, 2015.

[35] D. G. GREEN, *Cellular automata models in biology*, Math. Comput. Model., 13 (1990), pp. 69–74.

[36] G. GRÖSSING AND A. ZEILINGER, *Quantum cellular automata*, Complex Syst., 2 (1988), pp. 197–208.

[37] J. HARDY, O. DE PAZZIS, AND Y. POMEAU, *Molecular dynamics of a classical lattice gas: Transport properties and time correlation functions*, Phys. Rev. A, 13 (1976), pp. 1949–1961.

[38] A. W. HARROW, A. HASSIDIM, AND S. LLOYD, *Quantum algorithm for linear systems of equations*, Phys. Rev. Lett., 103 (2009), p. 150502.

[39] N. ISRAELI AND N. GOLDENFELD, *Coarse-graining of cellular automata, emergence, and the predictability of complex systems*, Physical Review E, (2006).

[40] Z. JIANG, A. B. TACLA, AND C. M. CAVES, *Bosonic particle-correlated states: A nonperturbative treatment beyond mean field*, Phys. Rev. A, 96 (2017), p. 023621.

[41] A. KANDALA, A. MEZZACAPO, K. TEMME, M. TAKITA, M. BRINK, J. M. CHOW, AND J. M. GAMBETTA, *Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets*, Nature, 54 (2017), p. 242.

[42] J. KEMPE, *Quantum random walks:an introductory overview*, Contemporary Physics, 44 (2003), pp. 307–327.

[43] P. D. LAX AND R. D. RICHTMYER, *Survey of the stability of linear finite difference equations*, Communications on Pure and Applied Mathematics, Ix (1956), pp. 267–293.

[44] R. J. LEVEQUE, *Finite Difference Methods for Ordinaty and Partial Differential Equations*, Siam, 1955.

[45] D. LEVY, *Introduction to numerical analysis*.
Department of Mathematics and Center for Scientific Computation and Mathematical Modeling, CSCAMM, University of Maryland, September 2010.

[46] O. MALOYER AND V. KENDON, *Decoherence versus entanglement in coined quantum walks*, New Journal of Physics, 9 (2007), p. 87.

[47] R. M. MAZO, *Brownian Motion. Fluctuations, Dynamics, and Applications*, Clarendon Press. Oxford, 2002.

[48] D. A. MEYER, *From quantum cellular automata to quantum lattice gases*, Journal of Statistical Physics, 85 (1996), pp. 551–574.

[49] M. MITCHELL, *Complexity A guided Tour*, Oxford University Press, 2009.

[50] G. D. MOLFETTA AND A. PÉREZ, *Quantum walks as simulators of neutrino oscillations in a vacuum and matter*, New Journal of Physics, 18 (2016), p. 103038.

[51] A. MONTANARO AND S. PALLISTER, *Quantum algorithms and the finite element method*, Phys. Rev. A, 93 (2016), p. 032324.

[52] S. NANDI, B. K. KAR, AND P. P. CHAUDHURI, *Theory and applications of cellular automata in cryptography*, IEEE Transactions on Computers, 43 (1994), pp. 1346–1357.

[53] P. NELSON, *Biological Physics*, W. H. Freeman, 2004.

[54] M. A. NIELSEN AND I. L. CHUANG, *Quantum Computation and Quantum Information*, Cambridge University Press, 2010.

[55] Y. OMAR, N. PAUNKOVIĆ, L. SHERIDAN, AND S. BOSE, *Quantum walk on a line with two entangled particles*, Phys. Rev. A, 74 (2006), p. 042304.

[56] M. F. M. OSBORNE, *Brownian motion in the stock market*, 7 (1959).

[57] C. A. PÉREZ-DELGADO AND D. CHEUNG, *Local unitary quantum cellular automata*, Phys. Rev. A, 76 (2007), p. 032320.

[58] P. PHILIPP AND R. PORTUGAL, *Exact simulation of coined quantum walks with the continuous-time model*, Quantum Information Processing, 16 (2016), p. 14.

[59] R. PORTUGAL, *Quantum Walks and Search Algorithm*, Springer, 2013.

[60] R. PORTUGAL, *Staggered quantum walks on graphs*, Phys. Rev. A, 93 (2016), p. 062335.

[61] R. PORTUGAL, M. C. DE OLIVEIRA, AND J. K. MOQADAM, *Staggered quantum walks with hamiltonians*, Phys. Rev. A, 95 (2017), p. 012328.

[62] R. PORTUGAL, R. A. M. SANTOS, T. D. FERNANDES, AND D. N. GONÇALVES, *The staggered quantum walk model*, Quantum Information Processing, 15 (2016), pp. 85–101.

[63] P.W.SHOR, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, Proc.R.Soc.London A, (1985).

[64] A. ROMANELLI, *Thermodynamic behavior of the quantum walk*, Phys. Rev. A, 85 (2012), p. 012319.

[65] C. RUNGE, *Über eine methode die partielle differenkalgliechung constans numerisch integrieren*, Z. Math, Phys, 56 (1908), pp. 225–232.

[66] R. A. M. SANTOS, R. PORTUGAL, AND S. BOETTCHER, *Moments of coinless quantum walks on lattices*, Quantum Information Processing, 14 (2015), pp. 3179–3191.

[67] I. SILOI, C. BENEDETTI, E. PICCININI, M. G. A. PARIS, AND P. BORDONE, *Quantum walks of two interacting particles on percolation graphs*, Journal of Physics: Conference Series, 906 (2017), p. 012017.

[68] M. SMOLUCHOWSKI, *Drei vorträge über diffusion, brownsche molekularbewegung und koagulation von kolloidteilchen*, Physik. Z, (1916), pp. 557–571.

[69] M. STEFFEN, D. P. DIVINCENZO, J. M. CHOW, AND T. N. THEIS, *Quantum computing: An ibm perspective*, in IEEE/CSC & ESAS EUROPEAN SUPERCONDUCTIVITY NEWS FORUM, no. 19, 2012.

[70] K. S. M. L. STEPHEN P. JORDAN, HARI KROVI AND J. PRESKILL, *Bqp-completeness of scattering in scalar quantum field theory*, arxiv.org/abs/1703.00454, (2017).

[71] D. SYCH AND G. LEUCHS, *A complete basis of generalized bell states*, New Journal of Physics, 11 (2009), p. 013006.

[72] M. SZEGEDY, *Quantum speed-up of markov chain based algorithms*, (2004), pp. 32–41.

[73] D. TATARU, *Nonlinear wave equations*, arXiv:math/0304397, (2003).

[74] Q. K. TELESFORD, S. L. SIMPSON, J. H. BURDETTE, S. HAYASAKA, AND P. J. LAURIENTI, *The brain as a complex system: Using network science as a tool for understanding the brain*, Brain Connect, 1 (2011), pp. 295–308.

[75] T. TOFFOLI, *Cellular automata as an alternative to (rather than an approximation of) differential equations in modeling physics*, Physica D: Nonlinear Phenomena, 10 (1984), pp. 117 – 127.

[76] T. TOFFOLI AND N. MARGOLOUS, *Cellular Automata Machines*, MIT Press Series in Scientific Computation, 1985.

[77] S. E. VENEGAS-ANDRACA, *Quantum walks: a comprehensive review*, Quantum Information Processing, 11 (2012), pp. 1015–1106.

[78] J. VON NEUMANN, *Theory of Self-Reproducing Automata*, University of Illinois Press, 1996.

[79] K. M. WANG, *Physical Implementation of Quantum Walks*, Springer, 2013.

[80] J. WATROUS, *On one-dimensional quantum cellular automata*, (1995), pp. 528–537.

[81] G. H. WEISS, *Random walks and their applications: Widely used as mathematical models, random walks play an important role in several areas of physics, chemistry, and biology*, American Scientist, 71 (1983), pp. 65–71.

[82] H. WEYL, *Gruppentheorie und quantenmechanik.*, Leipzig: S. Hirzel, (1928).

[83] K. WIESNER, *Quantum Cellular Automata*, Springer New York, New York, NY, 2009, pp. 7154–7164.

[84] S. WOLFRAM, *A new kind of science*, Wolfram Media, 2002.

[85] T. G. WONG AND R. A. M. SANTOS, *Exceptional quantum walk search on the cycle*, Quantum Information Processing, 16 (2017), p. 154.

[86] W. K. WOOTTERS AND W. H. ZUREK, *A single quantum cannot be cloned*, Nature, 299 (1982).

[87] X.-S. YANG AND Y. YOUNG, *Cellular automata, pdes, and pattern formation*, (2010).

[88] A. C.-C. YAO, *Quantum circuit complexity*, in Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science, Nov. 1993.

[89] W.-C. YUEH, *Explicit inverses of several triagonal matrices*, Applied Mathematics E-notes, (2006), pp. 74–83.

[90] F. ZÄHRINGER, G. KIRCHMAIR, R. GERRITSMA, E. SOLANO, R. BLATT, AND C. F. ROOS, *Realization of a quantum walk with one and two trapped ions*, Phys. Rev. Lett., 104 (2010), p. 100503.

[91] O. ZIENKIEWICS, R. TAYLOR, AND J. ZHU, *The finite element method its basis and fundamentals*, Elsevier, 2013.