Scenery Reconstruction

Dissertation zur Erlangung des Doktorgrades
der Fakultät für Mathematik
der Universität Bielefeld

vorgelegt von

Angelica Yohana Pachon Pinzon

August 2009

Gedruckt auf alterungsbeständigem Papier nach DIN ISO-9706

Contents

ACKNOWLEDGEMENTS

To my parents, Dora and Guillermo, for their amazing support and encouragement all the way through from the beginning to the end of this project. To my brother and sisters, Camilo, Rocio and Laura, for all their love which did not change despite the distance but on the contrary grew stronger every day.

To my advisers Prof. Friederich Goetze and Ph.D Heinrich Matzinger for believing in me and guiding me through the writing of this thesis.

To Professor Serguei Popov at Campinas University in Brazil for his assistance and encouragement when I needed it the most.

To the International Graduate College (IGK), colleges, professors and administrative staff. Without their support this project would have never be completed. My sincere appreciation to you

I would like to extend my heartfelt gratitude to my faith families at International Baptist Church, Bielefeld and my brothers and sisters in Christ of the Bible study group who gave me happiness through an amazing period of growing and walking in the Lord.

To my friends for always being there no matter what day, what time or whatever the reason, thank you. Sascha, thanks for being there and listening to me when I needed to talk with you. Daniella, thanks for your happiness, you make me feel like being at home. Caryl, Sarah, Sada, Rina, Anne and Suman, thanks for your good example - you never get tired of thinking of other people. Regina and Lars, thank you for your patience and for introducing me to the real world without any judgment. Nils, thank you for helping me - from the beginning until now you have always been there. Sven, thanks for your patience and for the several talks in German over many a nice cup of coffee. To my friends in Colombia, Angela, Anita and Julio, thanks for loving me as I am, for understanding me and for supporting me in those moments of absolutely sadness. To everybody that has been a part of my life and whom I might have failed to mention, thank you very much.

Now to him who is able to do immeasurably more than all we ask or imagine, according to his power that is at work within us. Ephesians 3:20.

THANKS ALMIGHTY GOD

Chapter 1

INTRODUCTION: SCENERY RECONSTRUCTION

The basic scenery reconstruction problem can be described as follows: suppose that to each $z \in \mathbb{Z}$ a color from the set $\{0, 1, \ldots, C-1\}$ is assigned. This defines a coloration of $\mathbb{Z}$ which we call " scenery" and denote by ξ. Formally, a scenery ξ is a map $\xi : \mathbb{Z} \to \{0, 1, \ldots, C-1\}$.

Suppose a simple random walk $\{S(t)\}_{t\in\mathbb{N}}$ starts to move on these colored integers registering the color it sees at each time $t > 0$ and producing a new sequence of colors. At time t, the random walk sees the color $\chi_t := \xi(S_t)$. The color record $\chi_0\chi_1\chi_2\ldots$ is denoted by χ. The question which arises naturally, is to which extent the original coloring ξ can be reconstructed given only one realization of the observations χ?

Under appropriate restrictions, it is often possible to reconstruct the scenery ξ when given only one realization of the observations χ. These restrictions are:

- Two sceneries are called equivalent if one of them is obtained from the other by translation and/or reflection. If ξ and ξ^* are equivalent, we can in general not distinguish whether the observations come from ξ or ξ^*.
- In [14], Lindenstrauss shows that there exist sceneries which can not be reconstructed. So, instead one shows that typical sceneries can be reconstructed: the scenery is taken to be the outcome of a random process and one tries to show that almost all sceneries can be reconstructed up to equivalence.
- One can only hope to be able to reconstruct a scenery up to translation and reflection and the reconstruction works in the best case only *almost surely* (*a.s.*)
- The reconstruction problem is impossible to solve for dimensions larger than 2 if we use a simple random walk (or any spatially homogeneous random walk), because when the dimension is at least 3, the random walk is transient.

We illustrate the problem with an example.

Example 1.0.1 *Take a portion of the scenery ξ and the beginning of the path from $\{S(t)\}_{t\in\mathbb{N}}$ as given by Figure 1.1. We observe the sequence of observations χ obtained*

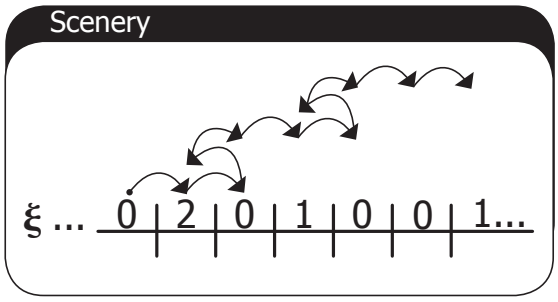


Figure 1.1: The arrows over ξ indicate the beginning of the path taken by S.

by following the path S on ξ. The first observations from χ look as follows:

0	*2*	*0*	*2*	*0*	*1*	*0*	*1*	*0*	*0*	*1*	*...*

The Scenery Reconstruction Problem consists in trying to reconstruct ξ up to equivalence, given only the observations χ. In practice, one tries to reconstruct a finite piece of ξ close to the origin with only a finite number of observations first. Most theoretical results are about reconstructing the whole scenery ξ with an infinite amount of observations. These are usually obtained by putting together finite pieces which where previously reconstructed from finite many observations.

In the next section, we present the history of the Scenery Reconstruction problem.

There exist various theoretical results on scenery reconstruction, but almost none lead to a practical algorithm. The proofs in [24] and [21], that a finite piece of scenery can be reconstructed in polynomial time seem too cumbersome to be implemented.

For the first time, we present an algorithm which is easy to implement. This is the content of chapter 2, where we show how to reconstruct a finite piece of scenery close to the origin. For the reconstruction we only use polynomially many observations (in the length of the piece). The scenery is taken to have 3 colors. The algorithm uses a method which is also used for DNA-reconstruction: if first obtains micro-strings which it then assembles. The micro-strings are of logarithmic order in the length of the piece to be reconstructed. The algorithm reconstructing the micro-strings needs exponential time in the size of the micro-strings. However, exponential of logarithmic leads to polynomial!
To reconstruct the micro-strings one uses a representation of the scenery ξ on a 3-regular tree $T = (E_T, V_T)$ equipped with a non-random coloring: $\psi : V_T \to \{0, 1, 2\}$. For now

assume the scenery ξ to be non-random at the origin. Assume also that our tree T has a root v_0, where the color coincides with ξ_0:

$$\psi(v_0) = \xi_0.$$

We request that for every vertex v of the tree, the three adjacent vertexes v_1, v_2, v_3 have all 3-colors represented:

$$\{\psi(v_1), \psi(v_2), \psi(v_3)\} = \{0, 1, 2\}.$$

There is unique nearest neighbor walk path R on T generating the sequence ξ on the tree and such that $R(0) = v_0$. In other words, there is a unique $R : \mathbb{Z} \to V_T$, such that

$$R_{t+1} \text{ is adjacent to } R_t, \ \forall t \in \mathbb{Z},$$

whilst

$$\psi(R_z) = \xi_z, \forall z \in \mathbb{Z}$$

and $R_0 = v_0$. The usefulness of the representation of ξ as nearest neighbor walk is as follows: imagine a bendable (double-infinite) tube. Put marks at distance 1 length-unit from each other in the tube. Chose one mark to be the 0-th mark. Then numerated the marks with all the numbers from $\mathbb{Z}$ in the order as they appear in the tube. Color the marks. For this let mark number z have color ξ_z. Hence, the color sequence in the tube corresponds the double-infinite sequence ξ. Imagine that the edges in the tree T are of length 1-unit. We bend the tube over the tree, so that every mark is placed over a vertex with the same color. In this way, the tube defines a nearest neighbor path on the tree T. This nearest neighbor walk path is exactly equal to R, the representation of ξ as such a path. Hence, we have found an intuitive image for R. The way this is useful goes as follows: remember that our problem starts with a random walk S"reading" the scenery ξ and thus producing a new color record χ given by

$$\chi = \chi_0\chi_1\chi_2 \ldots = \xi(S_0)\xi(S_1)\xi(S_2) \ldots,$$

which can also be written as

$$\chi = \xi \circ S. \tag{1.0.1}$$

Since, R generates ξ as color record on the tree T, we have $\xi = \psi \circ R$. Using the last equation together with 1.0.1, we obtain

$$\chi = \xi \circ S = (\psi \circ R) \circ S = \psi \circ (R \circ S). \tag{1.0.2}$$

Note that $R \circ S$ represents the nearest neighbor path

$$R(S_0), R(S_1), R(S_2), \ldots$$

on the tree T, which is obtained by composing R and S. More intuitively, $R \circ S$ can be viewed as " the random walk S inside the tube R, when we fold the tube over the tree T as described before". We can thus look at the tube as a bendable referential (just like in

general relativity! :)). But equation 1.0.2, tells us that the nearest neighbor path $R \circ S$ on the tree T, generates the color record χ. This means that given χ, we can reconstruct $R \circ S$. We have at this stage the following situation: a transparent tube R folded over the tree T, (tube is invisible) and within the folded tube a random walk which is visible. The tube is transient. So, there will be vertexes v_x and v_y which get visited only once by R. We take the time when $R \circ S$ goes in shortest time from v_x to v_y. This corresponds to a"straight walk" by the random walk S. During such a straight walk, the observations χ are simply a copy of the piece of the scenery ξ crossed during that time. (For the details, see chapter 2).

This idea of a transparent folded tube serving as referential to the random walk S, is very appealing intuitively. We could easily draw the situation with color-pencils. But it is not clear how to implement this efficiently with a computer. In Chapter 4, we present an essential contribution for the computer implementation: the geometric ideas are translated into an algebraic formalism, which is easy to compute. For this we identify every vertex v of the tree T with a finite string: the string we chose is the color-record obtained when we move in shortest way from the root v_0 to v. Assume that the nearest neighbor path R is located on the vertex v at time $z > 0$:

$$R(z) = v.$$

The string representing v can be obtained by taking the color record of the finite observations:

$$\psi(R_0)\psi(R_1)\dots\psi(R_z) = \xi_0\xi_1\dots\xi_z, \tag{1.0.3}$$

modulo $aba = a$. By "modulo $aba = a$", we mean that any substring of type aba gets replace by a. We proceed until the string can not be simplified any further. The string thus obtained is the one representing v. For this, it is not important where we first apply our simplification rule. The end-result is always the same. For any color record obtained by a nearest neighbor walk on the tree marching from v_0 to v, when simplified maximally modulo $aba = a$, always leads to the same string. Assume that the random walk S at time t is located at the point z:

$$S_t = z,$$

and as before let $R_z = v$. Then, the nearest neighbor walk $R \circ S$ at time t is located in v. This implies applying our simplification rule to the color record made by $R \circ S$ up to time t, yields the string corresponding to v. In other words, when taking the string

$$\chi_0\chi_1\cdots\chi_t,$$

modulo $aba = a$, we obtain a number corresponding to the vertex at which $R \circ S$ is at time t.

In chapter 3, we present a fundamental improvement for determining times when the walker is close to the origin. Usually one takes the scenery to be i.i.d.. So, for reconstructing the scenery ξ in the finite interval I, the observations made whilst S is

outside I, are not helpful. In other words, to stand a chance to reconstruct the restriction of ξ to I, we need to determine times when S is in I. A simple approach to solving this problem is presented in [1]. But that approach fails with less than five colors. We boost the proof to make it work with only 4 colors. The approach should also be useful in the future for the reconstruction of sceneries with low entropy. (So, far the reconstruction methods work only with sceneries which contain a lot of entropy (i.i.d.)). It is an important open question if by decreasing the entropy, we reach a point where scenery reconstruction becomes impossible. Let us explain a little more of the method in chapter 3. Assume an i.i.d. scenery ξ with C equiprobable colors. Assume that S is a simple symmetric random walk on $\mathbb{Z}$. Assume further more that x is an integer number such that $|x| > 2n$ and let w^n be the string $w^n = \xi_0\xi_1 \dots \xi_{n-1}$. Let R be a nearest neighbor path on $\mathbb{Z}$ of length n and starting in x. Since R takes only steps of length 1, it can not enter the interval $[0, n-1]$. So, the observations made by R are independent of w^n. This leads to

$$P(\xi \circ R = w^n) = \left(\frac{1}{C}\right)^n.$$

There are 2^n nearest neighbor paths starting in x. Thus, the probability that there exists a nearest neighbor path starting in x and generating w^n is bounded from above by

$$\left(\frac{2}{C}\right)^n. \tag{1.0.4}$$

The last expression above is an exponentially small quantity in n, as soon as $C > 2$. The main contribution of chapter 3, is to realize that the bound 1.0.4 can be very much improved. For this it is noticed that the path R which should generate the word $w^n = \xi_0\xi_1 \dots \xi_{n-1}$, should produce an observation-string which reflects the distribution of $w^n = \xi_0\xi_1 \dots \xi_{n-1}$. Most paths will generate observations having a very different distribution from ξ and can thus be excluded. In most situations reconstructing a scenery becomes way more difficult when the entropy of the scenery is reduced. The present method seems however still possible with low-entropy sceneries: when the entropy of $\xi_0\xi_1 \dots \xi_{n-1}$ is low, we will also be able to reduce the number of nearest neighbor paths susceptible of generating w^n. This gives us new hope for the low-entropy case.

1.1 History

The scenery reconstruction problem is part of the research area known as "Random walk in random sceneries" (RWRS), which studies the distribution of the observations of a random media by a random walk. Let us explain the (RWRS) problem.

Let $(X_n)_{n\in\mathbb{Z}}$ be a sequence of $i.i.d$ random variables taking values in a possible infinite set $F \in \mathbb{Z}^d$ according to a common distribution μ_F, with full support on F. Let $(S_n)_{n\in\mathbb{Z}}$ be a two-side simple random walk on $\mathbb{Z}^d$, i.e,

$$S_0 = 0 \text{ and } S_n - S_{n-1} = X_n,\ n \in \mathbb{Z}.$$

The simple random walk is the case where $F = \{e \in \mathbb{Z}^d; |e| = 1\}$ and $\mu_F(e) = \frac{1}{2^d}$, for $e \in F$.

Define another sequence of *i.i.d* variables, but this time taking values in a finite set G according to a common distribution μ_G, with full support on G. Denote them by $(\xi_z)_{z\in\mathbb{Z}^d}$ and call G a "set of colors" and the path realization ξ a "random scenery". Observe that ξ_x represent the scenery value at the site x. Take S and ξ independent of each other.

Let $(\chi_n)_{n\in\mathbb{Z}}$ with $\chi_n = (\xi \circ S)_n$ be the scenery observed along the random walk. This sequence is called "scenery record" or simply "observations" .

Random walk in random scenery is the joint process $(Z_n)_{n\in\mathbb{Z}}$, where $Z_n = (X_n, \chi_n)$. This process register at the same time two things: the step taken by the random walk and the scenery value at the site it visits.

Scenery reconstruction is the problem of recovering ξ given only $\chi_+ = (\chi_n)_{n\geq 0}$: with a single realization of the scenery seen trough a random walk at nonnegative times, is it possible to reconstruct the full scenery without knowing the walk?

The scenery reconstruction comes from the T, T^{-1} problem. The T, T^{-1}-problem is a problem from ergodic theory. The origin of this problem is a famous conjecture by Kolmogorov. He demonstrated that every Bernoulli shift T has a trivial tail-field (let us call the class of all transformations having a trivial tail-field $\mathcal{K}$) and conjectured that also the converse is true. This was proved to be wrong by Ornstein, who presented an example of a transformation which is $\mathcal{K}$ but not Bernoulli. Evidently his transformation was constructed for the particular purpose to resolve Kolmogorov's conjecture. In 1971 Ornstein, Adler, and Weiss came up with a very natural example which is $\mathcal{K}$ but appeared not to be Bernoulli. This was the T, T^{-1}-transformation, and the T, T^{-1}-problem was to show that it was not Bernoulli. In a famous paper Kalikow [7] showed that the T, T^{-1}-transformation is not even loosely Bernoulli and therefore solved the T, T^{-1}-problem.

The T, T^{-1}-transformation gives rise to a random process of pairs. The first coordinate of these pairs can be regarded as the position of a realization of simple random walk on the integers at time i. The second coordinate tells which color the walker would read at time i, if the integers were colored by an i.i.d. process with black and white in advance. In other words, that process of pairs is just the above described "Random walk in random scenery" process $(Z_n)_{n\in\mathbb{Z}}$, where $Z_n = (X_n, \chi_n)$.

This is the original setup of the scenery distinguishing and scenery reconstruction problems. They are related to the T, T^{-1}-problem, but actually we also consider them interesting in their own rights.

At the end of the 1970's and during the 1980's, several results about the ergodic properties of the "scenery record" got published. In the mid 1980's, Keane and den Hollander in [9] and also Benjamini and Kesten in [3] asked themselves if a pair of non-equivalent sceneries could be distinguished. For the "scenery distinguishing problem", we assume a set of two sceneries $\{\xi^a, \xi^b\}$ known to us. We also assume that the observed scenery ξ is either equal to ξ^a or to ξ^b, but we don't know which one. From the color record $\xi \circ S$, one tries to infer whether $\xi = \xi^a$ or $\xi = \xi^b$. Two sceneries are called "distinguishable sceneries" if with only one sequence of observations χ_+ it is possible to determine $a.s.$ which of them was observed.

In [14] Lindenstrauss constructed a countable infinite collection of one dimensional non-equivalent sceneries that can not be distinguished by a simple random walk. The sceneries in the collection have measure zero under an $i.i.d$ scenery process.

In [3], Benjamini and Kesten show that almost all pairs of sceneries are distinguishable, even when the dimension is two and only two colors in the sceneries. More recently in [6], Howard shows that any pair of periodic non-equivalent sceneries are distinguishable. Same thing for periodic with a single defect.

The problem of distinguishing two sceneries different only in one site is called "detecting a single defect in a scenery". In [10], Kesten shows that $a.s.$ it is possible to recognize a single defect in the case with at least five colors.

The question from Kesten about whether a single defect can be detected even if there are only two color in the scenery, is the origin of the scenery reconstruction problem. In [18], Matzinger answered the question proving that typical 2-color sceneries can be reconstructed almost surely up to equivalence.

In [20], it is proven that almost every 2-color scenery can be reconstructed when seen along the path of a simple symmetric random walk. The scenery is taken i.i.d. with equiprobable symbols. In [19], the same result is shown for a 3-color scenery.

Later Kesten noticed that Matzinger's method in [20] depends heavily on the one dimensionality of the problem and the skip-freeness of the random walk. This observation produced intense research on the matter. During the next years Lember, Matzinger, Merkl and Rolles worked on these questions. The techniques for solving the reconstruction problem when the random walk is not skeep-free [17], or two dimensional [16] are very different to the approach in [20].

Scenery reconstruction problem become more difficult when the number of colors decrease (except in the trivial case, when there is only one color). In [12], Lember and Matzinger consider a random walk with jumps on a 2-color scenery. Den Hollander asked

whether the scenery reconstruction problem is possible if the jumps were unbounded. In [13], Lenstra and Matzinger answer positively this question. In [15], Löwe and Matzinger undertake the reconstruction problem when the scenery is not i.i.d. but contain correlations between the different sites.

Some others developments are in [2], where Matzinger and Hart make a new version of the problem working on distinguishing sceneries with error-corrupted observations. In [28], Matzinger and Popov solve a continuous analog and in [27], Pachon and Popov get a result on $d-$dimension considering a random walk with branches.

Chapter 2

DNA-APPROACH TO SCENERY RECONSTRUCTION

Modern DNA sequencing methods work by reconstructing small micro pieces at first and then assembling them. In this chapter we use the same approach for scenery reconstruction and present a practical algorithm for the reconstruction of a finite piece of scenery around the origin. It just works by reconstructing small pieces with length order $\log n$ and then assembling them in order to retrieve a finite piece of length n. This allows us to reconstruct a finite piece in polynomial time. (The polynomial time is take in the length of the reconstructed piece).

This is an important break-through, since until now the existing algorithms were more of theoretical interest since they seemed too difficult to implement in practice.

In [24], [21], [22] [24] , Matzinger and Rolles prove that in certain cases finite pieces of a 3-color sceneries can be reconstructed in polynomial time close to the origin. However, it might be very difficult to implement their method for a computer program, hence these are a more theoretical result.

We will prove that a simple combinatorial approach allows in the 3-color case to reconstruct a finite piece in polynomial time and show how to implement it in practice.

We restrict ourselves to 3-color i.i.d. sceneries. Thus, we consider a coloring ("scenery") ξ of the integers with 3 colors. This means that ξ is a map from $\mathbb{Z}$ to $\{0, 1, 2\}$.

We show that with high probability we can reconstruct a piece of length order n in finite time. High probability here means one minus a term which goes to zero as n tends to infinity. Let us formulate our main result more precisely:

2.1 Formulation of the Problem and Theorem

Assume that the scenery $\{\xi_i\}_{i\in\mathbb{Z}}$ is a double infinite *i.i.d.* sequence of random variables with state space $\{0,1,2\}$ for which:

$$P(\xi_i = 0) = P(\xi_i = 1) = P(\xi_i = 2) = 1/3.$$

Then, ξ will designate a path realization of $\{\xi_i\}_{i\in\mathbb{Z}}$. Let $\{S_t\}_{t\in\mathbb{N}}$ be a simple symmetric random walk starting at the origin and let χ_t be the observation of the random walk at time t, i.e.

$$\chi_t := \xi(S_t).$$

Theorem 2.1.1 *For every $n \in \mathbb{N}$ large enough, there exists a map*

$$\mathcal{A} : \{0,1,2\}^{\mathcal{T}^n} \to \cup_{m\in[2n,8n]}\{0,1,2\}^m$$

such that

$$P\left(\begin{array}{l}\exists i_1, i_2 \text{ such that}\\ i_1 \in [-4n,-n]; i_2 \in [n,4n] \text{ and}\\ (\mathcal{A}(\chi_1\chi_2\dots\chi_{\mathcal{T}^n}) = \xi_{i_1}\xi_{i_1+1}\dots\xi_{i_2} \text{ or}\\ \mathcal{A}(\chi_1\chi_2\dots\chi_{\mathcal{T}}) = \xi_{i_2}\xi_{i_2-1}\dots\xi_{i_1})\end{array}\right) \geq 1 - n^{-\beta}, \tag{2.1.1}$$

where, $\mathcal{T}^n := n^6 + n^{9k_3+9}$ whilst $k_3 > 0$ and $\beta > 0$ are constants not depending on n.

In the theorem above the map $\mathcal{A}(.)$ represents the output of an algorithm which takes as input the first $\mathcal{T}^n$ observations of χ and spits out a piece of scenery. With high probability, the piece of scenery is a piece containing the restriction of ξ to $[-n,n]$. The reconstructed "piece of ξ" is contained with high probability in $[-4n,4n]$. The algorithm is presented in section 2.3. The sections 2.4 and 2.5 are there to prove that the algorithm works with high probability. This then guaranties that theorem 2.1.1 holds true.

2.2 Main Ideas

Let us mention several ideas which are used for the reconstruction.

2.2.1 Reconstructing from pieces

Definition 2.2.1 *Let $s = s_1s_2\dots s_i$ and let $r = r_1r_2\dots r_j$ be two strings such that $i < j$. Let s^* denote the transpose $s^* := s_is_{i-1}\dots s_1$.*
We say that s appears in more than one location in r iff there exists $x+i-1, y+i-1 \leq j$ with $x \neq y$, such that at least one of the following three conditions hold:

1. *$s = r_xr_{x+1}\dots r_{x+i-1}$ and $s = r_yr_{y+1}\dots r_{y+i-1}$*

2. $s = r_x r_{x+1} \dots r_{x+i-1}$ *and* $s^* = r_y r_{y+1} \dots r_{y+i-1}$

3. $s^* = r_x r_{x+1} \dots r_{x+i-1}$ *and* $s^* = r_y r_{y+1} \dots r_{y+i-1}$.

The idea of reconstruction from pieces is enough to reduce the problem of reconstructing a piece of scenery of length order n, to reconstructing substrings with shorter length. This is important since short strings can be reconstructed much quicker.

Let $\mathcal{S}$ be a string of length order l. The question is: How can we reconstruct $\mathcal{S}$ using substrings? The answer is: one first needs to show that with high probability in an *i.i.d.* 3-equiprobable-color string of length order l every substring of length $k \ln l$ appears only in one location, with $k > 0$ a constant large enough but not depending on l. (This is shown in the next section). The algorithm works as follows:

1. Assume we have a collection W of substrings of the string $\mathcal{S}$, and that for each substring s of $\mathcal{S}$ of length $K \ln l + 1$ there exists $w \in W$ such that s is a substring of w.

2. Then assemble the strings in W by asking if they fit on at least $K \ln l$ consecutive letters.

Consider the following example.

Example 2.2.1 *Assume we are given the words*

$$w_1 = 22321, w_2 = 3212, w_3 = 1212.$$

Assume that these three words are all substrings of a string $\mathcal{S}$ in which every 3-letter substring appears at most in one location. (By substring we mean that it appears in the string when we read it from left to right or right to left). Then we assemble w_1, w_2 and w_3 in order to get a bigger substring of $\mathcal{S}$.

First take w_1 and w_2 and see on which three letter group they coincide.

w_1	2	2	3	2	1	
w_2			3	2	1	2
w_4	2	2	3	2	1	2

Now puzzle together w_4 and the transpose $w_3^t = 2121$.

w_4	2	2	3	2	1	2	
w_3^t				2	1	2	1
w_5	2	2	3	2	1	2	1

Hence the string $w_5 = 2232121$ must be a substring of $\mathcal{S}$.

2.2.2 Reconstructing a word

For the reconstruction from pieces it is assumed that a collection of substrings of the string which we want to reconstruct is given. Of course one first needs to produce these short substrings which later get assembled. Stopping times are used for the production of these short words:

Assume x and y are two non-random integer numbers such that $x < y$. Assume that we want to reconstruct the "word" written between x and y, i.e. we would like to reconstruct $\xi_x\xi_{x+1}\dots\xi_y$. For this assume that we have the observations $\chi_1\chi_2\dots$ and whenever the random walk S visits x or y.

Let ν_i be the i-th visit of the random walk to x and let τ_i be the i-th visit to y. Then, when the random walk crosses in the shortest period of time from x to y we have that the random walk takes only steps to the right. Hence, during such a minimal time in the observations we are seeing a copy of the word $\xi_x\xi_{x+1}\dots\xi_y$.

Given that our random walk is recurrent, it will cross in the shortest period of time from x to y infinitely often. Hence, to reconstruct the word between x and y take

$$\chi_{\nu_i}\chi_{\nu_i+1}\chi_{\nu_i+2}\cdots\chi_{\tau_j}$$

where ν_i and τ_j satisfy

$$\tau_j - \nu_i = \min_{k,l}\{|\tau_k - \nu_l|\}.$$

Now, apriori the stopping times τ_j and ν_i are not observable. In the next subsection we explain how often we can figure them out based solely on the observations χ.

2.2.3 The Combinatorial Tree

The idea here is to translate the problem of reconstructing a sequence of colors on the integers to retrieve a path on a 3-regular tree. Formally the idea is as follows:

Let $T = (E_T, V_T)$ be a 3-regular tree with root v_0, and let $\psi : V_T \to \{0,1,2\}$ be a (random) coloring on T such that every vertex $v \in V_T$ has its 3 adjacent vertices colored in the three different colors 0, 1 and 2, i.e.

$$\forall v \in V_T, \{\psi(w) | w \in \{v_1, v_2, v_3\}\} = \{0,1,2\}, \tag{2.2.1}$$

where v_1, v_2, v_3 are the three vertexes adjacent to v.

Let ψ^0, ψ^1 and ψ^2 be three non-random colorations such that each one satisfies the condition (2.2.1) and $\psi^i(v_0) = i$ for $i = 0,1,2$. We assume that ψ is always equal to either ψ^0, ψ^1 or ψ^2. When $\xi(0) = 0$, then $\psi = \psi^0$, whilst when $\xi(0) = 1$, then $\psi = \psi^1$ and finally $\xi(0) = 2$ implies $\psi = \psi^2$.

So the color at the origin of ψ is the same color at the origin of ξ. Also, ψ "is only random as far as $\xi(0)$ is".

We call the map $R : I \cap \mathbb{Z} \to V_T$ (where I is an interval) on T, a nearest neighbor path on T, iff for all $z \in I \cap \mathbb{Z}$ we have that: $R(z)$ and $R(z+1)$ are adjacent vertexs, i.e.

$$\forall z \in I \cap \mathbb{Z}, \{R(z), R(z+1)\} \in E_T.$$

Let $\zeta : I \cap \mathbb{Z} \to \{0, 1, 2\}$ be a 3-color scenery on $I \cap \mathbb{Z}$, then we say that R generates ζ on ψ iff $\zeta = \psi \circ R$.

In order to represent the double infinite sequence ξ as a nearest neighbor walk R on a colored tree ψ we need a condition of uniqueness.

Proposition 2.2.1 *Let $\mathcal{S} = s_1 s_2 \dots s_j \in \{0, 1, 2\}^j$ be a string, and let v be a vertex in V_T such that $\psi(v) = s_1$. Then, there is a unique nearest neighbor path $\{R_t\}_{t\in[1,j]}$ on T such that,*

$$\psi(R_1) \dots \psi(R_j) = s_1 s_2 \dots s_j,$$

with $R_1 = v$. Thus, R generates $\mathcal{S}$.

Proof. Suppose that $\{U_t\}_{t\in[1,j]}$ and $\{W_t\}_{t\in[1,j]}$ are two nearest neighbor paths such that

$$\psi(U_1) \dots \psi(U_j) = s_1 s_2 \dots s_j = \psi(W_1) \dots \psi(W_j),$$

with $U_1 = W_1 = v$. Then, at time 2, U_2 and W_2 will be over an adjacent vertex from v, and also $\psi(U_2) = s_2 = \psi(W_2)$, so by (2.2.1) $U_2 = W_2$. Suppose now that at time $k < j$, $U_k = W_k = v_k$, then at time $k+1$, U_{k+1} and W_{k+1} will be over an adjacent vertex from v_k, and also $\psi(U_{k+1}) = s_{k+1} = \psi(W_{k+1})$, so, one more time by (2.2.1) $U_{k+1} = W_{k+1}$. Thus by the induction argument we have that $\{U_t\}_{t\in[2,j]} = \{W_t\}_{t\in[2,j]}$. ■

What this preposition says is that, given any sequence of colors $\mathcal{S}$, there is a unique nearest neighbor walk that generates $\mathcal{S}$ once we know where it starts.

Note that the representation of ξ as a nearest neighbor path R defines a simple random walk on the graph (E_T, V_T):

More precisely, for $z \geq 0$, we have $P(\{R(z+1) = v_1 | R(z) = v\}) = 1/3$, $P(\{R(z+1) = v_2 | R(z) = v\}) = 1/3$ and $P(\{R(z+1) = v_3 | R(z) = v\}) = 1/3$, with $\{v_1, v_2, v_3\}$ designating the 3-adjacent vertexes of v. Thus $\{R_z\}_{z\in\mathbb{N}}$ is a random walk on our tree (E_T, V_T) starting at the origin. Same thing for $\{R_{-z}\}_{z\in\mathbb{N}}$. Thus then let R designate the unique nearest neighbor path R on T with $R(0) = v_0$ and such that

$$\psi(R(z)) = \xi(z),$$

$\forall z \in \mathbb{Z}$. We call $\{R_z\}_{z\in\mathbb{Z}}$ the representation of the scenery $\{\xi_i\}_{i\in\mathbb{Z}}$ as nearest neighbor walk on T.

Using this representation, our problem of reconstructing ξ is translated to reconstructing R using the observations χ. Even though, we do not know R, we can easily figure out $R \circ S$ just with the observations χ.

By definition we have that $\psi(R(z)) = \xi(z)$, then, $\forall t \in \mathbb{N}$

$$\psi(R(S_t)) = \xi(S_t) = \chi_t$$

hence,

$$\psi \circ (R \circ S) = \chi.$$

Note that $R \circ S$ is also a nearest neighbor walk on T, and it is the only one who generates χ on ψ.

In this order of ideas, if we knew R, we would also know ξ, so the problem of reconstructing ξ is equivalent to reconstructing R given $R \circ S$. Let us look at an example:

Example 2.2.2 *Suppose that the scenery is*

$$\begin{aligned} \xi_z &= 0201001.... \\ z &= 0123456... \end{aligned}$$

and over it a random walk S_t starts to move producing χ_t, then we have

$$\begin{aligned} S(0), \ldots, S(10) &= 0, 1, 2, 1, 2, 3, 4, 3, 4, 5, 6 \text{ and} \\ \chi(0), \ldots, \chi(10) &= 0, 2, 0, 2, 0, 1, 0, 1, 0, 0, 1. \end{aligned}$$

Now observe R and $R \circ S$ over the tree (Figure 2.1). They are respectively the representation of ξ and χ.

2.2.4 Using the representation of ξ by R

Let v and w be two different vertexes of V_T such that v, w are visited by R_z, and let $\{x_j, x_{j-1}, \ldots x_1\}$ and $\{y_1, y_2, \ldots y_i\}$ be respectively the set of all times when R_z visits v and the set of all times when R_z visits w. Since R_z is transient (see Lemma 5 in [19]), then *a.s.* (almost sure) every vertex of V_T is visited only a finite number of times, so if v, w are far enough the next holds *a.s.*

$$x_j < x_{j-1} < \ldots < x_2 < x_1 < y_1 < y_2 < \ldots < y_i.$$

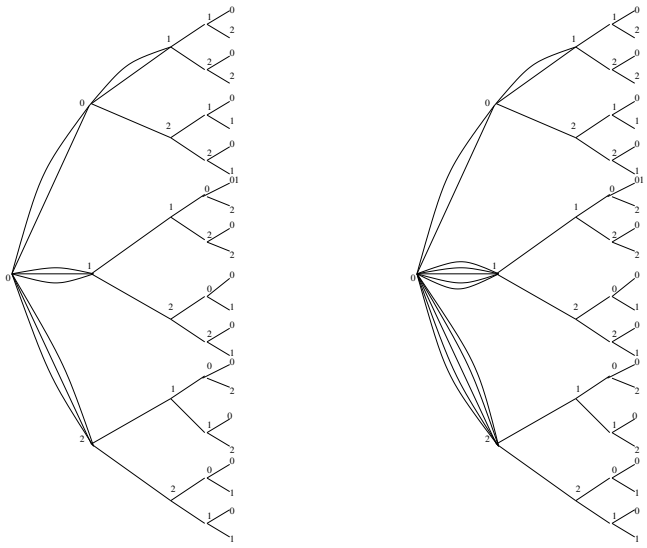


Figure 2.1: Left and right side, respectively they are R and $R \circ S$.

Thus, by the method of "reconstructing a word", there is a simple way to reconstruct the scenery between x_1 and y_1.

Take the pair of times (t, s) *minimizing* $(s - t)$ *under the constrains*

$$R(S_t) = v \text{ and } R(S_s) = w, \;\; s > t.$$

Then, $a.s.$ we have that the observations during the interval of time $[t, s]$ are equal to the scenery between x_1 and y_1, i.e.

$$\chi_t \chi_{t+1} \cdots \chi_s = \xi_{x_1} \xi_{x_1+1} \ldots \xi_{y_1}.$$

The reason why this works is because the random walk, when going in shortest time from x_1 to y_1 goes in a straight way. While doing so reveals the piece of scenery ξ between x_1 and y_1. Also, we know that the recurrent random walk will cross from x_1 to y_1 in the shortest period of time infinitely often.

2.3 The Algorithm

Let us describe the algorithm to reconstruct a piece of ξ which contains the string $\xi(-n)\xi(-n+1)\xi(-n+2)\ldots\xi(n)$ and is contained in the string $\xi(-4n)\xi(-4n+1)\xi(-4n+2)\ldots\xi(4n)$.

Let $\mathcal{T}^n := n^6 + n^{9k_3+9}$, where $k_3 > 0$ is a constant not depending on n and which will get defined subsequently. Our algorithm only takes as input the observation up to time $\mathcal{T}^n$. Hence, the algorithm only uses a polynomial number of observations in the length of the piece of scenery to be reconstructed. Let V^n denote the subset of V_T containing those vertexes which have been visited by $R \circ S$ up to time $\mathcal{T}$ and are not further away from the root than n, i.e.

$$V^n := \{\ R(S(t)) \mid d(R(S(t)), v_0) \leq n, t \in [0, \mathcal{T}]\ \}.$$

1. Determine V^n.

2. Build a "lexica" W of words which can be obtained by shortest passage: for any pair $(v_1, v_2) \in (V^n) \times (V^n)$ such that up to time $\mathcal{T}$, the nearest neighbor walk $R \circ S$ goes at least once from v_1 to v_2 in less or equal $(k + 2k_1) \ln n + 1$ steps:
   take (t, s) minimizing $s - t$ under the constraints

   $$R(S_t) = v_1 \text{ and } R(S_s) = v_2, s > t\ ;\ s, t \leq \mathcal{T}.$$

   The string $\chi_t \chi_{t+1} \dots \chi_s$ is one of the reconstructed words. Keep now only those words with length at least $k \ln n + 1$. The set of words obtained in this way is denoted by W.

3. Assemble the words from W. For this use the following assembling rule: in order to "puzzle two words together", the words or their transposes must coincide on at least $k \ln n$ contiguous letters. (Check out subsection 2.2.1 to see how this is done). Start with a word which was obtained using the vertex v_0. Assemble one word after the other to the already assembled word. Produce in this manner one piece of scenery. (Some words might not get used since they might occur in another interval which is not connected to the reconstructed interval.)

2.4 Combinatorics

Let $\mathcal{A}lg$ be the event that our reconstruction algorithm works. More precisely,

$$\mathcal{A}lg := \left\{ \begin{array}{l} \text{the reconstructed piece contains } \xi(-n)\xi(-n+1)\xi(-n+2)\dots\xi(n), \\ \text{and is contained in the string } \xi(-4n)\xi(-4n+1)\xi(-4n+2)\dots\xi(4n) \end{array} \right\}.$$

Now define:

$$
\begin{aligned}
B_1 &:= \left\{ \{\forall z \notin [-4n, 4n], d(R_z, v_0) > n\} \right\}, \\
B_2 &:= \left\{ \begin{array}{l} \text{in the string } \xi_{-4n}\xi_{-4n+1}\ldots\xi_{4n}, \\ \text{there is no word of length } k\log n \text{ appearing} \\ \text{in two different places} \end{array} \right\}, \\
B_3 &:= \left\{ \begin{array}{l} \text{for every } z \in [-4n, 4n], \\ R(z - k_1 \ln n) \notin R[z, +\infty] \end{array} \right\}, \\
B_4 &:= \left\{ \begin{array}{l} \text{for every } z \in [-4n, 4n], \\ R(z + (k + k_1)\ln n + 1) \notin R[-\infty, z + k\ln n + 1] \end{array} \right\}, \\
B_5 &:= \left\{ \begin{array}{l} \text{every subinterval of } [-4n, 4n] \text{ of length } (k + 2k_1)\log n + 1, \\ \text{gets crossed in a straight manner by } S \text{ before time } n^6 + n^{9k_3+9} \end{array} \right\}, \\
B_6 &:= \left\{ \{R(z) \neq v_0, \forall z \notin [n/2, n/2]\} \right\},
\end{aligned}
$$

Let B_7 be the event that for all $z \in [n - k_1 \ln n, n + k_1 \ln n]$ we have

$$d(R(z), v_0) \leq n.$$

The main combinatorial lemma is:

Lemma 2.4.1 *We have that*

$$B_1 \cap B_2 \cap B_3 \cap B_4 \cap B_5 \cap B_6 \cap B_7 \subset \mathcal{A}lg.$$

Proof. Let (v_1, v_2) be a pair of vertices of V^n which gets selected by our algorithm and which leads to a reconstructed word w for W. Let (t, s) be the time pair minimizing $s - t$ under the constrain $R(S_t) = v_1$ and $R(S_s) = v_2$ whilst $s > t$ and $s, t < \mathcal{T}$. Hence, the reconstructed word w is equal the observations χ during the time interval $[t, s]$, that is:

$$w = \chi_t\chi_{t+1}\ldots\chi_s.$$

Because of B_1, we have that $R(z)$ can only be in V^n, when $z \in [-4n, 4n]$. It follows that if B_1 holds, then, since $v_1, v_2 \in V^n$, we must have that $S_s, S_t \in [-4n, 4n]$. Denote S_t by z_1 and S_s by z_2. The algorithm chooses only pairs (v_1, v_2) for which the nearest neighbor walk $R \circ S$ goes in less than $(k + 2k_1)\ln n + 1$ steps from one to the other. It follows, that

$$|z_1 - z_2| \leq (k + 2k_1)\ln n + 1.$$

We have already seen that $z_1, z_2 \in [-4n, 4n]$. But according to the event B_5, every interval of $[-4n, 4n]$ of length less or equal to $(k + 2k_1)\ln n + 1$ gets crossed in a straight manner by S before time $\mathcal{T}$. This implies that before time $\mathcal{T}$, the nearest neighbor walk

S will walk in a straight manner from z_1 to z_2.

Hence, if (t, s) is to minimize $s - t$ under the constrain $R(S_t) = v_1$ and $R(S_s) = v_2$ and $s - t > 0$ with $s, t \leq \mathcal{T}$, then, necessarily during the time (t, s) the random walk S must walk in a straight way from z_1 to z_2. (Otherwise, we would not have a minimum, since the straight walk would be shorter).

Since, during the time interval (t, s) the random walk makes steps only in one direction, we have that the observations during that time are a copy of the scenery between the points z_1 and z_2. More precisely, assume that the random walk S makes only steps to the right during the time interval (t, s). Then the observations $\chi_t \chi_{t+1} \dots \chi_s$ are equal to $\xi_{z_1} \xi_{z_1+1} \xi_{z_1+2} \dots \xi_{z_2}$.

Hence the reconstructed word w, is equal to $\xi_{z_1} \xi_{z_1+1} \xi_{z_1+2} \dots \xi_{z_2}$ and is part of the scenery ξ restricted to $[-4n, 4n]$. The same conclusion holds true if the steps taken during the time (s, t) are all to the left, but then the reconstructed word is equal to $\xi_{z_1} \xi_{z_1-1} \xi_{z_1-2} \dots \xi_{z_2}$. We have just proven that if B_1 and B_5 both hold, then the collection of words W reconstructed by our algorithm contains only words contained in the part of the scenery:

$$\xi_{-4n} \xi_{-4n+1} \xi_{-4n+2} \dots \xi_{4n-1} \xi_{4n}.$$

We have so far that there are no "wrong words" in W. For the algorithm to work, that is for the "puzzling together" of the words to lead to the desired reconstructed piece, we also need to make sure that we have enough good words in W. This is what we are going to check next:

Let z and $z + k \ln n + 1$ both be in $[-n, n]$. Let

$$v_1 := R(z - k_1 \ln n)$$

and let

$$v_2 := R(z + k \ln n + 1 + k_1 \ln n).$$

Let z_1 be the largest $z \in \mathbb{Z}$ for which $R(z) = v_1$. Then because of the event B_3, we have that $z_1 < z$ and hence

$$z_1 \in [z - k_1 \ln n, z].$$

Let z_2 be the smallest $z \in \mathbb{Z}$ such that $R(z) = v_2$. Because of B_4, we find that $z_2 > z + k \ln n + 1$ and hence

$$z_2 \in [z + k \ln n + 1, z + k \ln n + 1 + k_1 \ln n].$$

So, the couple (z_1, z_2) minimizes $|z_1 - z_2|$ under the constrain

$$R(z_1) = v_1, R(z_2) = v_2.$$

By the event B_5 we have that up to time $\mathcal{T}$, the random walk crosses at least once from v_1 to v_2 in a straight way. Let the times of such a straight crossing be denoted by (s_1, s_2), hence $S_{s_1} = v_1$ and $S_{s_2} = v_2$ and during the time interval (s_1, s_2), the random walk S takes steps only in one direction and $s_1, s_2 \leq \mathcal{T}$. Since z_1, z_2 minimizes $|z_1 - z_2|$ under $R(z_1) = v_1$ and $R(z_2) = v_2$, we have that the shortest way for the nearest neighbor walk $R \circ S$ to go from v_1 to v_2 is in $|z_1 - z_2|$ steps, and this can only occur when S walks straight from z_1 to z_2. Hence, the time (s_1, s_2) corresponds to a straight crossing of the random walk S from z_1 to z_2, so that:

$$\chi_{s_1}\chi_{s_1+1}\cdots\chi_{s_2} = \xi_{z_1}\xi_{z_1+1}\ldots\xi_{z_2}.$$

Again, note that up to time $\mathcal{T}$, the time pair (s_1, s_2) minimizes $|s_2 - s_1|$ under the constrain $R(S_{s_1}) = v_1$ and $R(S_{s_2}) = v_2$. So, as soon (v_1, v_2) get picked by our algorithm, then

$$\xi_{z_1}\xi_{z_1+1}\ldots\xi_{z_2}$$

will be a reconstructed word put by our algorithm into W. Now, we know that $|s_1 - s_2|$ are less apart then $(k + 2k_1)\ln n + 1$. This implies that the nearest neighbor walk $R \circ S$ goes from v_1 to v_2 in less or equal to $(k + 2k_1)\ln n + 1$ steps. This is the first criteria for the pair of vertices (v_1, v_2) to get selected. The second criteria is that $v_1, v_2 \in V^n$, this is guarantied by the event B_7. We have just proven that if all the events B_3, B_4, B_5, B_7 holds, then the substring

$$\xi_{z_1}\xi_{z_1+1}\ldots\xi_{z_2} \tag{2.4.1}$$

is obtained by our reconstruction algorithm and added to W. The piece of scenery 2.4.1, contains the piece

$$w_z := \xi_z\xi_{z+1}\ldots\xi_{z+k\ln n+1}.$$

So, we have proven that for every interval

$$[z, z + k\ln n + 1] \subset [n, -n]$$

there is in the set of words W at least one word w containing the piece w_z.
We had proven that if B_1 and B_5 both hold, then the collection of words W reconstructed by our algorithm contains only words contained in the part of the scenery:

$$\xi_{-4n}\xi_{-4n+1}\xi_{-4n+2}\ldots\xi_{4n-1}\xi_{4n}.$$

The event B_2 guaranties that the algorithm "puzzles" words of W together correctly, that is the result is again a piece of

$$\xi_{-4n}\xi_{-4n+1}\xi_{-4n+2}\ldots\xi_{4n-1}\xi_{4n}.$$

The algorithm starts puzzling with a word w_0 which was obtained using the vertex v_0. By B_6, we have that the word w_0 is part of the restriction of ξ to $[-n, n]$. For every $[z, z + k\ln n + 1]$ in $[-n, n]$, we have at least one word in W containing $\xi_z\xi_{z+1}\ldots\xi_{z+k\ln n+1}$. This implies, that the final reconstructed piece by our algorithm, must contain the restriction

of ξ to $[-n, n]$. It must also be contained in the restriction of ξ to $[-4n, 4n]$ since all the words in W are. This finishes proving that if all the events B_1, B_2, B_3, B_4, B_5, B_6 and B_7 holds, then our algorithm manages to reconstruct a piece the way we want it to. This means that the reconstructed piece is contained in the restriction of ξ to $[-4n, 4n]$, but contains the restriction of ξ to $[-n, n]$. In other words, the event $\mathcal{A}lg$ holds. ■

Proof of the main theorem Here we want to prove our main theorem, that is theorem 2.1.1. This means that we want to prove that our reconstruction algorithm has a probability to fail bounded by a negative power in n. Note that according to our combinatorial lemma (that is lemma 2.4.1), we have that the reconstruction algorithm works correctly as soon as all the events $B_1, B_2, \ldots, B_7$ all hold. Thus the probability that the algorithm does not work is bounded from above by the sum:

$$P(B_1^c) + P(B_2^c) + P(B_3^c) + P(B_4^c) + P(B_5^c) + P(B_6^c) + P(B_7^c). \tag{2.4.2}$$

But in the next section we provide for each of the probabilities:

$$P(B_1^c), P(B_2^c), P(B_3^c), P(B_4^c), P(B_5^c), P(B_6^c) \text{ and } P(B_7^c)$$

upper bounds. None is larger than a negative power in n. Hence, it follows that 2.4.2 can also be bounded by a negative power in n.

2.5 Events with high probability

Lemma 2.5.1 *We have that*

$$P(B_1^c) \leq c_1 e^{-c_2 n},$$

where c_1 and c_2 are positive constants not depending on n.

Proof. Let $D_z = d(R_z, v_0)$ be the distance between v_0 and the vertex corresponding to R_z. Then, $\{D_z\}_{z\geq 0}$ is a simple random walk reflected at the origin, for which $P(D_z - D_{z-1} = 1 | D_{z-1} \neq 0) = 2/3$ and $P(D_z - D_{z-1} = -1 | D_{z-1} \neq 0) = 1/3$. Hence, $\{D_z\}_{z\geq 0}$ is a random walk with positive drift reflected at the origin. We can use this to write

$$\begin{aligned} B_1 &= \Big[\cap_{z>4n} \{D_z > n\}\Big] \cap \Big[\cap_{s<-4n} \{D_{|s|} > n\}\Big], \text{ so} \\ P(B_1^c) &\leq 2 \sum_{i>4n} P(D_i \leq n). \end{aligned}$$

Let $\{T_z\}_{z\geq 0}$ be a random walk with the same transition probabilities as $\{D_z\}_{z\geq 0}$ and starting at the origin. Then

$$P(D_z \leq n) \leq P(T_z \leq n),$$

thus,

$$P(B_1^c) \leq 2 \sum_{i>4n} P(T_i \leq n).$$

Writing $T_i = X_1 + \cdots + X_i$, where $X_1, \ldots, X_i$ are $i.i.d$ random variables with $P(X_1 = 1) = 1 - P(X_1 = -1) = \frac{2}{3}$, we have that

$$\begin{aligned} P(B_1^c) &\leq 2 \sum_{i>4n} P(X_1 + \cdots + X_i \leq n) \\ &= 2 \sum_{i>4n} P\Big((\frac{1}{4} - X_1) + \cdots + (\frac{1}{4} - X_i) \geq \frac{i - 4n}{4}\Big) \\ &\leq 2 \sum_{i>4n} P\Big((\frac{1}{4} - X_1) + \cdots + (\frac{1}{4} - X_i) \geq 0\Big). \end{aligned}$$

Let Y_i be equal to

$$Y_i := \frac{1}{4} - X_i.$$

Then the right side of the very last inequality above is equal to

$$\sum_{i>4n}^{\infty} P(Y_1 + Y_1 + \ldots + Y_i \geq 0). \tag{2.5.1}$$

Note that $E[Y_i] = 0.25 - 0.\bar{3} = -0.08\bar{3}$ and hence by Large Deviation Theory, expression 2.5.1 must be exponentially small in n. Let us check out the details:

Recall that $P(Z \geq 0) \leq E[e^{Zt}]$ for any $t \geq 0$. Taking Z equal to $Y_1 + Y_2 + \ldots + Y_n$, we obtain

$$P(Y_1 + \cdots + Y_n \geq 0) \leq E[e^{Y_1 t}]^n, \tag{2.5.2}$$

for any $t \geq 0$.

Let $f(t)$ be the function $f(t) = E[e^{Y_1 t}]$, then if,

1. there exists an open interval I around 0 such that $E[e^{Y_1 t}]$ is finite for all $t \in I$, and
2. the expectation of Y_1 is negative, i.e., $E[Y_1] < 0$,

there exists a small $t \geq 0$ such that $E[e^{Y_1 t}] \leq 1$.

The best possible exponential upper bound for (2.5.2) is the positive value for t which minimizes $E[e^{Y_1 t}]$.

For our definition of Y_1, i.e. $\left(Y_1 = \frac{1}{4} - X_1\right)$, the two above conditions hold, and $E[e^{Y_1 t}]$ reaches the minimum value for $t_0 = 0.091$. And $E[e^{Y_1 t_0}] = 0.99618$. It follows that

$$\begin{aligned} P(B_1^c) &\leq 2\sum_{i=4n}^{\infty}(0.99618)^i \\ &= 2\frac{(0.99618)^{4n}}{(1-0.99618)} \\ &= c_1 e^{-c_2 n}, \end{aligned}$$

with $c_1 = 523.56$ and $c_2 = 0.0153$. ■

Lemma 2.5.2 *We have*

$$P(B_2^c) \leq 128 n^{(2-0.5k\log 3)}.$$

Proof. Let w_z denote the word:

$$w_z := \xi_z \xi_{z+1} \xi_{z+2} \ldots \xi_{z+k\log n}$$

and let $\bar{w}_z$ be the word

$$\bar{w}_z := \xi_z \xi_{z-1} \xi_{z-2} \ldots \xi_{z-k\log n}.$$

Let B_{2,z_1,z_2} be the event that w_{z_1} is not equal to w_{z_2}. Let $\bar{B}_{2,z_1,z_2}$ be the event that w_{z_1} is not equal to $\bar{w}_{z_2}$. Clearly:

$$B_2 = (\cap_{z_1 \neq z_2} B_{2,z_1,z_2}) \cap \left(\cap_{z_1,z_2} \bar{B}_{2,z_1,z_2}\right)$$

where the intersections above are taken over z_1, z_2 in $[-4n, 4n]$. The last equation above leads to:

$$P(B_2^c) \leq \left(\sum_{z_1 \neq z_2} P(B_{2,z_1,z_2}^c)\right) + \left(\sum_{z_1,z_2} P(\bar{B}_{2,z_1,z_2}^c)\right) \tag{2.5.3}$$

where the sums above are taken with z_1, z_2 ranging over $[-4n, 4n]$. Assume to simplify notations that n is an even number. For $z_1 \neq z_2$, we can always find at least $k\log n/2$ letters which are "all independent of each other in w_{z_1} and w_{z_2}". More precisely, since $z_1 \neq z_2$, there exists an integer subset $I \subset [0, k\log n]$ with at least $k\log n/2$ elements, so that

$$(z_1 + I) \cap (z_2 + I) = \emptyset.$$

Hence, using the fact that the scenery ξ is i.i.d. with 3 equiprobale colors, we find that if $z_1 \neq z_2$, then

$$P(B_{2,z_1,z_2}^c) = P(w_{z_1} = w_{z_2}) \leq \left(\frac{1}{3}\right)^{k\log n/2}. \tag{2.5.4}$$

A similar argument yields:

$$P(\bar{B}_{2,z_1,z_2}^c) = P(w_{z_1} = \bar{w}_{z_2}) \leq \left(\frac{1}{3}\right)^{k\log n/2}. \tag{2.5.5}$$

Applying inequalities 2.5.4 and 2.5.5 to inequality 3.2.15 yields:

$$P(B_2^c) \leq 128n^2 \left(\frac{1}{3}\right)^{k \log n/2} = 128n^{2-0.5k \log 3}.$$

The bound on the right side of the above inequality is a negative power of n as soon as $2 - 0.5k \log 3$ is strictly negative. Hence, we just have to take $k > 0$ strictly larger than $4/\log 3$ to have a negative-power-in-n upper bound for $P(B_2^c)$. ■

Lemma 2.5.3 *We have that*

$$P({B_3}^c) \leq c_1 n^{1-c_2 k_1},$$

where c_1 is a positive constant not depending on n.

Proof. Let B_{3z} be the event that $\{R(z - k_1 \log n) \notin R(\ [z, +\infty)\ )\}$. Then

$$B_3 = \cap_{z=-4n}^{4n} B_{3z}. \tag{2.5.6}$$

Note that the probability of B_{3z} does not depend on z. So equation 2.5.6 implies:

$$P({B_3}^c) \leq \sum_{z=-4n}^{4n} P(B_{3z}^c) \leq 9nP(B_{3z}^c). \tag{2.5.7}$$

Taking thus z equal to $k_1 \log n$, we obtain

$$P(B_{3z}) = P(R(0) \notin R(\ [k_1 \log n, +\infty)\ ) = P(v_0 \notin R(\ [k_1 \log n, +\infty)\ ).$$

In other words, the probability of the event B_{3z} is equal to the probability that after time $k_1 \log n$ the nearest neighbor walk $R(.)$ does not return to v_0. As in the prove of lemma 2.5.1, let the distance between $R(z)$ and v_0 be denoted by D_z so that

$$D_z := d(v_0, R(z)).$$

Again, as in 2.5.1, we have that D_z is a simple random walk on $\mathbb{N}$ reflected at the origin. Let $\{T_z\}_{z \geq 0}$ be a random walk with the same transition probabilities as $\{D_z\}_{z \geq 0}$ and starting at the origin. We have that

$$P(B_{3z}^c) \leq \sum_{z \geq k_1 \log n} P(T_z \leq 0) \tag{2.5.8}$$

Let $X_i := T_i - T_{i-1}$. Hence, we can use large deviations to bound

$$P(T_z \leq 0) = P(X_1 + X_2 + \ldots + X_z \leq 0).$$

With the same argument as in 2.5.1, we find

$$P(T_z \leq 0) \leq E(e^{-X_1 t})^z,$$

for any $t \geq 0$ and X_1 is the random variable such that

$$P(X_1 = 1) = 1 - P(X_1 = -1) = 2/3,$$

then we have $E[X_i] = +1/3$.

Minimizing $t \to E(e^{-X_1 t})$ with respect to t, we get

$$\min_{t \geq 0} E[e^{-X_1 t}] = 0.94281$$

so that with inequality 2.5.8, we obtain:

$$P(B_{3z}^c) = \sum_{z \geq k_1 \log n} P(T_z \leq 0) \leq \sum_{z \geq k_1 \log n} 0.94281^z.$$

The last inequality above with inequality 2.5.7 together imply that $P(B_3^c)$ is less than:

$$\begin{aligned} &= \frac{9n(0.94281)^{k_1 \log n}}{1 - 0.94281} \\ &= c_1 n e^{-c_2 k_1 \log n}, \text{ or} \\ &= c_1 n^{1 - c_2 k_1}, \end{aligned}$$

where $c_1 = 157.37$ and $c_2 = 0.0589 = \log 0.94281$. If we take the constant k_1 large so that $k_1 > \frac{1}{c_2}$, then our bound:

$$P(B_3^c) \leq c_1 n^{1 - c_2 k_1},$$

is a negative power in n and hence goes to 0 as n goes to infinity. ■

Lemma 2.5.4 *We have that*

$$P(B_4{}^c) \leq c_1 e^{\log n(1 - c_2 k_1)},$$

where $c_1 > 0$ is a positive constant not depending on n.

Proof. By symmetry follows the same steps as for the proof of Lemma 2.5.3. ■

Lemma 2.5.5 *We have that B_5 holds with high probability:*

$$P(B_5^c) \leq \frac{c_5}{n}$$

where $c_5 > 0$ is a constant not depending on n.

Proof. Let $k_3 > 0$ be the constant:

$$k_3 := k + 2k_1.$$

Let B_{51} be the event that the random walk S visits the points $-4n$ and $4n$ before time n^6.
Let B_{52z} be the event that the random walk S visits the point z at least

$$2^{3k_3 \log(n)} = n^{3k_3}$$

times within n^{9k_3+9} time unit from the first visit to z. More precisely, let τ_{zi} be the i-th visit by S to the point z. Hence:

$$\tau_{z1} := \min\{t | S_t = z\}$$

and by induction on i:

$$\tau_{z(i+1)} := \min\{t > \tau_{zi} | S_t = z\}.$$

The event B_{52z} can now be described as the event that the difference

$$\tau_{zj} - \tau_{z1}$$

is less or equal to n^{9k_3+9} for all $j \leq n^{3k_3}$. Let B_{53z} be the event that within the first n^{3k_3} visits to z there is at least one followed immediately by a straight crossing of length $k_3 \log n + 1$. More, precisely, let B_{53z} be the event that there exists i (random) such that $i \leq n^{3k_3}$ and

$$S_{t+1} - S_t = +1$$

for all $t \in [\tau_i, \tau_i + n^{3k_3}]$.
We have the following inclusion:

$$B_{51} \cap \left(\cap_{z\in[-4n,4n]} B_{52z}\right) \cap \left(\cap_{z\in[-4n,4n]} B_{53z}\right) \subset B_5 \tag{2.5.9}$$

The above inclusion can be explained as follows: for any $z \in [-4n, 4n]$ by the event B_{51} we have that the first visit to z by S takes place before time n^6. Then by B_{52z}, we get n^{3k_3} visits to z before an additional time n^{9k_3+9}. Hence, before time

$$\mathcal{T}^n := n^6 + n^{9k_3+9} \tag{2.5.10}$$

we have n^{3k_3} visits to z. According to the event B_{53z}, during those n^{3k_3} visits to z, there is at least one followed directly by a straight crossing of length $k_3 \log n + 1$. These crossings take place before time given in 2.5.10. In other words, we have just shown that when B_{51}, B_{52z} and B_{53z} all hold, then before time 2.5.10 there is a straight crossing by the random walk S of the interval $[z, z + k_3 \log n + 1]$. when there is such a straight crossing for each $z \in [-4n, 4n]$, then the event B_5 holds. This finishes proving the inclusion 2.5.9. From inclusion 2.5.9, we obtain:

$$P(B_5^c) \leq P(B_{51}^c) + \sum_{z\in[-4n,4n]} P(B_{52}^c) + \sum_{z\in[-4n,4n]} P(B_{53z}^c) \tag{2.5.11}$$

We can now apply the probability-bounds found in the next three lemmas to inequality 2.5.11 to find

$$P(B_5^c) \leq \frac{c_{51}}{n} + 8n\frac{c_{52}}{n^3} + 8ne^{-0.25n^{k_3}}.$$

In the sum, on the right side of last inequality above, the term with largest order is c_{51}/n. It follows, that there exists a constant $c_5 > 0$ not depending on n such that for all $n \in \mathbb{N}$, we have

$$P(B_5^c) \leq \frac{c_5}{n}.$$

■

Lemma 2.5.6 *We have that*

$$P(B_{51}^c) \leq \frac{c_{51}}{n}$$

where $c_{51} > 0$ is a constant not depending on n.

Proof. Let ν_i designate the first visit of the random walk S to the point i:

$$\nu_i := \min\{t | S_t = i\}.$$

Let $\tau_i := \nu_i - \nu_{i-1}$. By the strong Markov property of the random walk, the sequence

$$\tau_1, \tau_2, \tau_3, \ldots$$

is a sequence of i.i.d. variables. We have that the random walk reaches the point $4n$ before time n^6 iff we have

$$\tau_{4n} \leq n^6.$$

This and a symmetric argument for $-4n$ leads to

$$P(B_{51}^c) \leq 2P(\tau_1 + \tau_2 + \ldots + \tau_{4n} > n^6)$$

and hence,

$$P(B_{51}^c) \leq 2P\left((\tau_1 + \tau_2 + \ldots + \tau_{4n})^{1/3} > n^2 \right). \tag{2.5.12}$$

For positive numbers, the third power of the sum is always more than the sum of the third powers. Hence,

$$\tau_1 + \ldots + \tau_{4n} \leq (\tau_1^{1/3} + \ldots + \tau_{4n}^{1/3})^3$$

from which it follows that

$$(\tau_1 + \ldots + \tau_{4n})^{1/3} \leq \tau_1^{1/3} + \ldots + \tau_{4n}^{1/3}.$$

Applying the last inequality above to 2.5.12, we obtain

$$P(B_{51}^c) \leq 2P(\tau_1^{1/3} + \ldots + \tau_{4n}^{1/3} > n^2)$$

which with the help to the Markov inequality yields

$$P(B_{51}^c) \leq 8\frac{E[\tau_1^{1/3}]}{n}.$$

The above bound is useful because the (1/3)-th moment of τ_i is known to be finite.

■

Lemma 2.5.7 *We have that*

$$P(B_{52z}^c) \leq \frac{c_{52}}{n^3}$$

where $c_{52} > 0$ is a constant not depending on n or z.

Proof. Note that $P(B_{52z}^c)$ does not depend on z. Hence, we can find a bound for $P(B_{520}^c)$ nd this bound will be valid for all $P(B_{52z}^c)$. Let T_i be the i-th visit to the origin by S. We have

$$P(B_{520}^c) = P(T_1 + T_2 + \ldots + T_{n^{3k_3}} > n^{9k_3+9}).$$

Now, the expression on the right side of the equality above is equal to

$$P((T_1 + T_2 + \ldots + T_{n^{3k_3}})^{1/3} > n^{3k_3+3}). \tag{2.5.13}$$

Note that for non-negative terms, the third power of the sum is always larger than the sum of the third powers. Hence, in our case, taking the terms $T_i^{1/3}$ we find:

$$T_1 + T_2 + \ldots + T_{n^{3k_3}} \leq \left(T_1^{1/3} + T_2^{1/3} + \ldots + T_{n^{3k_3}}^{1/3}\right)^3$$

Taking the third root of the last inequality above we obtain:

$$(T_1 + T_2 + \ldots + T_{n^{3k_3}})^{1/3} \leq T_1^{1/3} + T_2^{1/3} + \ldots + T_{n^{3k_3}}^{1/3}. \tag{2.5.14}$$

Because of inequality 2.5.14, we find that the probability in expression 2.5.13 is less or equal to

$$P(T_1^{1/3} + T_2^{1/3} + \ldots + T_{n^{3k_3}}^{1/3} > n^{3k_3+3})$$

By the Markov inequality, we obtain

$$P(T_1^{1/3} + T_2^{1/3} + \ldots + T_{n^{3k_3}}^{1/3} > n^{3k_3+3}) \leq \frac{E[T_1^{1/3}]}{n^3}$$

and hence

$$P(B_{52z}^c) \leq \frac{E[T_1^{1/3}]}{n^3}.$$

The bound on the last inequality above is useful because $E[T_1^{1/3}]$ is know to be a finite number. ■

Lemma 2.5.8 *We have that*

$$P(B_{53z}^c) \leq e^{-0.25n^{k_3}}.$$

Proof. Let Y_i be the Bernoulli variable which is equal to one iff we have a straight crossing of length $k_3 \log n + 1$ right after the stopping time τ_{zj}, where $j = i(k_3 \log n + 1)$. Since, we take the stopping times $\tau_{z\cdot}$ apart by at least $k_3 \log n + 1$, we get that $Y_1, Y_2, \ldots$ are i.i.d. Also, the probability of a straight crossing is:

$$P(Y_i = 1) = \left(\frac{1}{2}\right)^{k_3 \log n + 1} = \frac{1}{2n^{k_3}}.$$

The event B_{53z} holds, as soon as at least one of the Y_i's is equal to 1 for $i = 1, 2, \ldots, n^{2k_3}$. Hence,

$$P(B_{53z}^c) \leq P(\sum_{i=1}^{n^{2k_3}} Y_i = 0) = (1-q)^{n^{2k_3}}, \tag{2.5.15}$$

where

$$q = \frac{1}{2n^{k_3}}.$$

Note that

$$\left(1 - \frac{1}{2n^{k_3}}\right)^{2n^{k_3}}$$

converges to e^{-1} as $n \to \infty$. Applying this to 2.5.15, yields for n large enough the bound

$$P(B_{53z}^c) \leq e^{-0.25n^{k_3}}.$$

■

Lemma 2.5.9 *We have that*

$$P({B_6}^c) \leq c_1 e^{-c_2 n},$$

where c_1 and c_2 are positive constants not depending on n.

Proof. Let $D_z = d(R_z, v_0)$ be the same as in the proof of Lemma 2.5.1. Recall that D_z is a simple random walk reflected at the origin with bias $+1/3$. We can write B_6 as

$$B_6 = \{\cap_{z>n/2} D_z > 0\} \cap \{\cap_{s<-n/2} D_s > 0\}.$$

Hence,

$$P({B_6}^c) \leq 2 \sum_{i=n/2}^{\infty} P(D_i = 0).$$

Let $\{T_z\}_{z\geq 0}$ be a random walk with the same transition probabilities as $\{D_z\}_{z\geq 0}$, and let $X_1, \ldots, X_i$ be independent and identically distributed random variables with $P(X_1 = 1) = 1 - P(X_1 = -1) = 2/3$. Then once again:

$$\begin{aligned} P({B_6}^c) &< 2\sum_{i=n/2}^{\infty} P(T_i \leq 0) \\ &= 2\sum_{i=n/2}^{\infty} P(X_1+, \ldots, +X_i \leq 0) \\ &= 2\sum_{i=n/2}^{\infty} P(-X_1-, \ldots, -X_i \geq 0) \\ &\leq 2\sum_{i=n/2} (0.94281)^i, \\ &< 2\frac{2(0.94281)^n}{(1-0.94281)} \\ &= c_1 e^{-c_2 n}, \end{aligned}$$

where $c_1 = 34.971$ and $c_2 = 0.0589$ ■

Chapter 3

RECOGNITION OF TIMES A WALKER IS CLOSE TO THE ORIGIN

In this chapter we consider a simple random walker moving on a random media. Whilst doing so, the random walker observes at each point of time the "color" of the location he is at. This process creates a sequence of observations.

We consider the problem of determining when the walker is close to the origin. For this we are only given, the observations made by the walker as well as a small portion of the media close to the origin. With that information alone, we show that we can typically construct an exponential number of stopping times, which all occur whilst the walker is on the small piece of media available to us. The number is exponential in the size of that small piece of media.

So far this problem could only be solved when the media contained 5 colors. In the present chapter, we use a subtle entropy argument on the set of possible observations given the point where the walker starts and given the media in that neighborhood. This allows us to achieve our goal when the media contains 4 equiprobable colors.

Our present result, implies that the Scenery Reconstruction result in [1] also applies with 4 colors as opposed to just 5 colors.

3.1 Formulation of the Problem and Theorem

Let S_t denote the position of a random walker at time t. We assume that S_t is a simple symmetric random walk starting at the origin. Let $\xi : \mathbb{Z} \to \{0, 1, 2, 3\}$ denote a coloring of the integers with 4-colors. We call the landscape ξ a scenery. We assume that at every time $t \geq 0$, the random walker sees the color of the point he is at. This implies that at

time t the random walker observes the color $\chi_t := \xi(S_t)$.

The problem we consider in this chapter is to figure out times when the random walker is in a close vicinity of the origin. For this we are given only the observations

$$\chi := \chi_0\chi_1\chi_2\cdots$$

and the restriction of ξ to $[0, n-1]$.

The way we try to guess when the random walk is close to the origin, is by searching in the observations for the word

$$w^n := \xi_0\xi_1\xi_2\ldots\xi_{n-1}.$$

More precisely, consider the stopping times $\tau_1, \tau_2, \ldots$ defined as follows:

$$\tau_1 := \min\{t \geq n | w^n = \chi_{t-n+1}\chi_{t-n+2}\cdots\chi_t\}.$$

By induction on i, τ_{i+1} is the next time the pattern w^n appears in the observations:

$$\tau_{i+1} := \min\{t > \tau_i | w^n = \chi_{t-n+1}\chi_{t-n+2}\cdots\chi_t\}.$$

We take the scenery ξ to be *i.i.d.* The pattern w^n will appear infinitely often in the scenery ξ. So there is no hope that at all the times τ_i the walker is close to the origin, because he will also observe the pattern w^n in other locations. Instead, we will prove that an exponential number of the times τ_i tell us that the walker is close to the origin.

Let B^n be the event that the walker is close to 0 for all τ_i with $i \leq (v_2)^n$. Here v_2 is a constant not depending on n satisfying

$$\frac{2}{2^{H_2(0.25)}} > v_2 > 1, \tag{3.1.1}$$

where $H_2(x)$ is the entropy function:

$$H_2(x) := x\log_2(1/x) + (1-x)\log_2(1/(1-x)).$$

(Note that $2/2^{H_2(0.25)} > 1$, so that a constant v_2 satisfying 3.1.1 really exists!). In the event B^n, "close to the origin" is defined as in the interval $[0, n-1]$, so that

$$B^n := \{S_{\tau_i} \in [0, n-1], \forall i \leq v_2^n\}.$$

Our main theorem states that when the scenery ξ is taken *i.i.d.* with four equiprobable colors then the probability that B^n does not hold is exponentially small in n. (This is true for any constant v_2 satisfying 3.1.1 but not depending on n). Here comes the theorem:

Theorem 3.1.1 *Assume that ξ_z with $z \in \mathbb{Z}$ is a collection of i.i.d. variables independent of the simple symmetric random walk S_t starting at the origin. We also assume that the variables ξ_z are equally likely to be equal to* 0,1, 2 *or* 3*:*

$$P(\xi_z = 0) = P(\xi = 1) = P(\xi = 2) = P(\xi = 3) = 1/4.$$

Then, all the stopping times τ_i up to $i = v_2^n$ stop the random walk S with high probability in $[0, n-1]$:

$$P(B^n) \geq 1 - e^{-c_B n}$$

for all $n \in \mathbb{N}$, where $c_B > 0$ is a constant not depending on n.

In [1], it is proven that an exponential number of times τ_i stop the random walk close to the origin in the context of a 5-color scenery. However the proof in [1] fails with less than 5 colors. We introduce a subtle entropy argument for the class of observations generated by a walker, which allows this improvement. The technique we develop here is important and we expect it to be useful in many other situations.

The stopping time problem considered here is an essential step for scenery reconstruction. Once many stopping times are constructed, it is relatively easy to reconstruct a large portion of the scenery around the origin. Once the stopping times are available, the scenery reconstruction can be performed exactly as in [1]. The present result implies that the scenery reconstruction result proven in [1] for 5 color sceneries also holds with 4-color sceneries. We explain more details on this in the next subsection.

3.1.1 Implication of present result for scenery reconstruction

The research in this area started with people investigating the ergodic properties of the observations made by a random walker of a random media. Kesten [10] proved that with five colors, if one knows the scenery in every point except in one, then, it becomes possible to reconstruct the missing color in that one location. For this purpose, the observations ξ are "observable". But, at that time, specialists believed that it might not be possible to distinguish single defects with less than 5 colors in the scenery. Hence, the result in [18] came as a surprise. Non-the-less, the general question remains open: when does it become impossible to reconstruct a scenery? When does reducing the entropy in the scenery whilst increasing it for the walker lead to a critical phenomena where the scenery becomes unreconstructable?

The article [1] is the first and only, where scenery reconstruction is shown to be possible despite the increment of the random walk having a non-bounded support. In [1], a symmetric random walk has its distribution close to a symmetric random walk, but with small probability $\delta > 0$ the steps can be larger than 1 unit. The conditions in [1] for the random walk can be written as

$$P(|S_{t+1} - S_t| \neq 1) \leq \delta \tag{3.1.2}$$

and

$$P(|S_{t+1} - S_t| = m \mid \ |S_{t+1} - S_t| \neq 1) \leq e^{-cm}, \ \forall m \in \mathbb{N} \tag{3.1.3}$$

for a constant $c > 0$ not depending on m. Hence, the step length has an exponentially decaying tail, but non-bounded support. Even when $\delta > 0$ is taken very small and $c > 0$ very large, all other reconstruction methods fail.

The algorithm in [1] achieves reconstruction for $\delta > 0$ small enough and $c > 0$ large enough. It reconstructs the scenery on larger and larger intervals. Once it has reconstructed the restriction of ξ to $[-n, n]$, it proceeds in determining ξ_{n+1} and ξ_{-n-1}. For this, it first obtains an exponential number of stopping times. It uses the observations χ and the already reconstructed piece $\xi_{-n}\xi_{-n+1}\dots\xi_{n-1}\xi_n$. These stopping times are shown to typically all occur whilst the walker is in $[-n, n]$. With the availability of these stopping times, the reconstruction of ξ_{n+1} and ξ_{-n-1} is relatively easy.

We can use the stopping times here constructed in the same way as in [1] to obtain ξ_{n+1} and ξ_{-n-1}. The stopping times are even defined in the same way here and in [1]. The only difference is that here we prove them to work with only 4 colors instead of 5. The fact that in [1], we do not only consider a simple random walk but also a slightly disturbed version of a simple random walk does not matter: the proof and methods provided here does also carry over to that case. This implies that scenery reconstruction is possible with a slightly disturbed random walk (i.e. taking $\delta > 0$ small enough and $c > 0$ large and assuming that 3.1.2 and 3.1.3 holds for a symmetric random walk S), even if there are only 4 equiprobable colors in the scenery.

The new idea used here gives us hope for sceneries with lower entropy. When entropy is low, scenery reconstruction becomes way more difficult. However, the present technique offers a new approach: If the string

$$w^n := \xi_0\xi_1\dots\xi_{n-1}$$

has low entropy, then we should also be able to very much restrict the collection of strings generated by a walker starting in a given point x on the scenery ξ and which might lead to w^n. (See below the argument restricting path which might generate w^n).

3.2 Proof of Theorem 3.1.1

3.2.1 Definition of events and combinatorics

Recall that w^n designates the word obtained by restricting the scenery ξ to $[0, n-1]$:

$$w^n := \xi_0\xi_1\xi_2\dots\xi_{n-1}.$$

Note that

$$\frac{4}{2^{H_2(0.25)}} > 2.$$

In the introductory section we defined the constant $v_2 > 1$ to be any constant not depending on n and satisfying 3.1.1. We will also need the constants r and v_1 which shall not depend on n, but satisfy the equation

$$\frac{4}{2^{H_2(0.25)}} > r > v_1 > 2v_2 > 2. \tag{3.2.1}$$

Let $q_0 > 0.25$ denote a constant not depending on n such that

$$\frac{4}{2^{H_2(0.25)}} > \frac{4}{2^{H_2(q_0)}} > r. \tag{3.2.2}$$

Note that we can always find such a constant q_0 because the entropy function $H_2(.)$ is strictly increasing in the interval $[0, 0.5]$.

Let us quickly give a sneaky preview of where the constants r, v_1 and v_2 make their appearance:

- With high likelihood, within a radius r^n of the origin there is no place where the word w^n can be read except at the origin.
- Typically at least $(v_1)^n$ visits to the origin occur before the random walk S leaves the interval $[-r^n, r^n]$.
- Typically, a number $(v_2)^n$ of stopping times all stop the random walk in $[0, n-1]$.

Let R be a map from the integer interval $[0, n-1]$ into $\mathbb{Z}$, i.e, $R : [0, n-1] \to \mathbb{Z}$, and such that

$$|R(i+1) - R(i)| = 1$$

for all $i \in [0, n-2]$. We call R a *nearest neighbor walk path* of length n and say R starts in x if $R(0) = x$.

Let $\mathcal{P}_x^n$ denote the set of all nearest neighbor paths of length n starting at x and such that the percentage of back-forth steps is less than $q_0 > 1/4$. Hence, $R : [0, n-1] \to \mathbb{Z}$ is in $\mathcal{P}_x^n$ iff both of the following conditions hold

1. $R(0) = x$, and
2. $|\{\ i \in [1, n-2] \mid (R(i) - R(i-1))(R(i+1) - R(i)) = -1\}| \leq q_0(n-2)$.

Let T denote the first visit by the random walk S to the set of two points $\{-r^n, r^n\}$:

$$T := \min \{\ t \mid |S_t| = r^n\ \}.$$

Next we define the events which we will use:

- Let ν_i denote the i-th visit after 0 by S to the origin:

$$\nu_{i+1} := \{t > \nu_i | S_t = 0\},$$

whilst $\nu_0 = 0$. (The random walk starts at the origin). Let C^n be the event that the random walk visits the origin at least $(v_1)^n$ times before time T:

$$C^n := \left\{ \nu_i \leq T, \forall i \leq (v_1)^n \right\}.$$

- Let D_1^n be the event that there is no path R starting in $[-r^n, r^n] - [-2n, 2n]$ with less than q_0-percentage of back-forth steps and generating the word w^n. In other words, the event D_1^n means that if

$$x \in [-r^n, r^n] \text{ and } x \notin [-2n, 2n]$$

and $R \in \mathcal{P}_x^n$ then

$$\xi(R_0)\xi(R_1)\xi(R_2)\ldots\xi(R_{(n-1)}) \neq w^n.$$

- Let D_2^n be the event that no path starting in $[-2n, 2n]$ and ending outside $[0, n-1]$ whilst having less than q_0-percentage of back-forth steps can generate the word w^n. More precisely, the event D_2^n means that $\forall x \in [-2n, 2n]$ and all $R \in \mathcal{P}_x^n$, we have that

$$\xi(R_0)\xi(R_1)\xi(R_2)\ldots\xi(R_{(n-1)}) \neq w^n,$$

if $R(n-1) \notin [0, n-1]$.

- Let E^n be the event that the random walk crosses the interval $[0, n-1]$ in a straight way at least $(v_2)^n$ times among the first $(v_1)^n$ visits to the origin. More precisely, let E^n be the event the (random) set

$$\left\{ \nu_i | i \leq (v_1)^n \ ; \ S_{j+1} - S_j = +1, \forall j \in [\nu_i, \nu_i + n - 1] \right\}$$

contains more than $(v_2)^n$ elements.

- Finally let F^n denote the event that the word w^n has a proportion less or equal to q_0 of letters w_i such that $w_i = w_{i+2}$. Hence, F^n is the event that

$$\text{Cardinality}\{i \in [1, n-1] | w_{i+1} = w_{i-1}\} \ \leq \ q_0(n-2).$$

Recall that B^n stands for the event that the first $(v_2)^n$ stopping times τ_i all occur whilst the random walk S is in the interval $[0, n-1]$.

Lemma 3.2.1 *We have that*

$$C^n \cap D_1^n \cap D_2^n \cap E^n \cap F^n \subset B^n.$$

Proof. With the event C^n we know that before time T, there are at least $(v_1)^n$ visits to the origin before time T. The event E^n guaranties that among the first $(v_1)^n$ visits to the origin, there are at least $(v_2)^n$ followed by a direct crossing of the interval $[0, n-1]$. When, the random walk S crosses the interval $[0, n-1]$ in a straight way, then during that time we see the pattern $w^n = \xi_0\xi_1 \dots \xi_{n-1}$ appearing in the observations. Thus, when C^n and E^n both hold, we see the pattern w^n appear at least $(v_2)^n$ times in the observations χ before time T. So, there will be at least $(v_2)^n$ stopping times τ_i before time T:

$$\tau_i \leq T \ , \forall i \leq (v_2)^n.$$

The next question is if those stopping times really stop the random walk in the interval $[0, n-1]$. With the event F^n, in the word w^n there are less than $q_0(n-2)$, letters w_i such that $w_i = w_{i+1}$. So, any nearest neighbor walk path with more than $q_0(n-2)$ "back and forth" steps can not generate w^n on the scenery ξ. In other words, any nearest neighbor walk path $R : [0, n-1] \to \mathbb{Z}$ starting in x but not in $\mathcal{P}_x^n$ can not generate w^n:

$$\xi(R_0)\xi(R_1) \dots \xi(R_{n-1}) \neq w^n.$$

So, when F^n holds, for a nearest neighbor walk path $R : [0, n-1] \to \mathbb{Z}$ to generate w^n, we need to have $R \in \mathcal{P}_x^n$ where $x := R(0)$. By the event $D_1^n \cap D_2^n$, for all $x \in [-r^n, r^n]$, and all $R \in \mathcal{P}_x^n$, R can generate w^n only if it ends in $[0, n-1]$. That means that with the event $D_1^n \cap D_2^n$, for all $x \in [-r^n, r^n]$ and all $R \in \mathcal{P}_x^n$, we have

$$\xi(R_0)\xi(R_1) \dots \xi(R_{n-1}) = w^n$$

implies $R(n-1) \in [0, n-1]$. Summarizing: when F^n and $D_1^n \cap D_2^n$ both hold, then the only way a nearest neighbor walk $R : [0, n-1] \to \mathbb{Z}$ can start in $[-r^n, r^n]$ and generate w^n on ξ, is when it ends in $[0, n-1]$, i.e. when $R(n-1) \in [0, n-1]$. Recall that by definition, up to time T the random walk S remains in $[-r^n, r^n]$. So, up to time T, when F^n and $D_1^n \cap D_2^n$ both hold, we can "only observe w^n when the random walk S follows a nearest neighbor walk path of length $n-1$ ending in $[0, n-1]$". In other words, for all $\tau_i \leq T$, we have that

$$S_{\tau_i} \in [0.n-1].$$

We have seen in the beginning of this proof, that when C^n and E^n both hold, then before time T we see the pattern w^n appear at least $(v_2)^n$ times in the observations χ. So, there at least $(v_2)^n$ stopping times τ_i before time T. With F^n and $D_1^n \cap D_2^n$ holding all these stopping times occur when S_{τ_i} is in $[0, n-1]$. Hence, all this together implies that the first $(v_2)^n$ stopping times τ_i happen whilst S_{τ_i} in is in $[0, n-1]$. Formally, we have proven that when all the events

$$C^n, E^n, F^n, D_1^n, D_2^n$$

hold then $S_{\tau_i} \in [0, n-1]$ for all $i \leq (v_2)^n$. This is the definition of the event B^n, so we have that

$$C^n \cap E^n \cap F^n \cap D_1^n \cap D_2^n \subset B^n.$$

■

3.2.2 High probability of events

Lemma 3.2.2 *We have that*

$$P(C^n) \geq 1 - \left(\frac{v_1}{r}\right)^n.$$

Proof. Let S_t^1 be a simple random walk such that $S_0^1 = 1$. Define the stopping time

$$\tau^1 = \min_t \Big\{ S_t^1 = 0 \text{ or } S_t^1 = r^n \Big\}.$$

We know that for a stopping time thus defined

$$E(S_{\tau_1}^1) = E(S_0^1),$$

then

$$r^n P\Big(S_{\tau_1}^1 = r^n) = 1$$

and

$$P\Big(S_{\tau_1}^1 = r^n) = \frac{1}{r^n},$$

but it is just the probability of S_t^1 visits r^n before visits the origin.

For the case of a simple random walk starting at minus one, S_t^{-1}, the probability of it visits $-r^n$ before visits the origin is also $\frac{1}{r^n}$. So the probability of a simple random walk to hitting r^n or $-r^n$ before hitting the origin is $p = \frac{1}{r^n}$.

Let C_i^n be the event that after the i-visit to the origin, the random walk first gets back to the origin before visiting the set $\{-r^n, r^n\}$.
(Recall that ν_i denotes the i-th visit after 0 by S to the origin:

$$\nu_{i+1} := \min\{t > \nu_i | S_t = 0\},$$

whilst $\nu_0 = 0$.) So, C_i^n is the event that

$$\min\{t > \nu_i || S_t| = r^n\} > \min\{t > \nu_i | S_t = 0\}.$$

by the strong Markov property of S, we have that

$$P(C_i^{nc}) = \frac{1}{r^n}. \tag{3.2.3}$$

But we have that

$$C^n = \bigcap_{i=0}^{(v_1)^n} C_i^n$$

and hence

$$P(C^{nc}) \leq \sum_{i=0}^{(v_1)^n} P(C_i^{nc})$$

The last inequality together with 3.2.3, yields

$$P(C^{nc}) \leq \sum_{i=0}^{(v_1)^n} \frac{1}{r^n} = \left(\frac{v_1}{r}\right)^n.$$

Note that the constants r and v_1 where defined so that $(v_1/r) < 1$, which implies that the last bound above is exponentially small in n. ■

Lemma 3.2.3 *We have that*

$$P(D_1^n) \geq 1 - 4\left(\frac{r \cdot 2^{H_2(q_0)}}{4}\right)^n.$$

Proof. Let D_{1x}^n denote the event that there is no nearest neaigborwalk path R in $\mathcal{P}_x^n$ and generating w^n on ξ. In other words, D_{1x}^n is the event that there is no nearest neighbor walk path $R : [0, n-1] \to \mathbb{Z}$, starting in x with less than q_0-percentage of back-and-forth steps and such that

$$\xi(R(0))\xi(R(1))\xi(R(2))\ldots\xi(R(n-1)) = w^n.$$

We have that

$$D_1^n = \bigcap_{x \in [-r^n, r^n] - [-2n, 2n]} D_{1x}^n$$

so that

$$P(D_1^{nc}) \leq \sum_{x \in [-r^n, r^n] - [-2n, 2n]} P(D_{1x}^{nc}). \tag{3.2.4}$$

Then, if R starts in x (that is $R(0) = x$), with $x \notin [-2n, 2n]$ and since the nearest neighbor walk path moving at most one unit by step, it follows that R can not enter the interval $[0, n-1]$. Assuming that R is non-random, we then obtain that the observation generated by R, that is the string

$$\xi(R_0)\xi(R_1)\ldots\xi(R_{n-1})$$

is independent of ξ restricted to $[0, n-1]$. This is because the scenery ξ is i.i.d. In other words, we obtain that w^n is independent of $\xi(R_0)\ldots\xi(R_{n-1})$. Since, we have 4 equiprobable colors in w^n, this leads to

$$P\left(w^n \ = \ \xi(R_0)\xi(R_1)\ldots\xi(R_{n-1})\ \right) = \left(\frac{1}{4}\right)^n, \tag{3.2.5}$$

for any non-random $R \in \mathcal{P}_x^n$ as soon as $x \notin [-2n, 2n]$. Now,

$$D_{1x}^n = \cap_{R \in \mathcal{P}_x^n} \{w^n \neq \xi(R_0)\xi(R_1)\ldots\xi(R_{n-1})\}$$

so that

$$P(D_{1x}^{nc}) \leq \sum_{R \in \mathcal{P}_x^n} P\left(w^n = \xi(R_0) \dots \xi(R_{n-1}) \right).$$

Applying now 3.2.5 to the last inequality above yields

$$P(D_{1x}^{nc}) \leq \sum_{R \in \mathcal{P}_x^n} \left(\frac{1}{4}\right)^n, \tag{3.2.6}$$

when $x \notin [-2n, 2n]$.

Since the number of different sequences with length n and proportion q_0 of back-and-forth steps is

$$\begin{pmatrix} n-2 \\ q_0(n-2) \end{pmatrix},$$

then using Stirling approximation for $n!$ we get that there are less than $2^{H_2(q_0)(n-2)}$ elements in the set $\mathcal{P}_x^n$, thus 3.2.6 can be written as:

$$P(D_{1x}^{nc}) \leq \frac{2^{H_2(q_0)(n-2)}}{4^n}$$

for all $x \in [r^n, r^n] - [-2n, 2n]$. Applying the last equation above to inequality 3.2.4, we obtain

$$P(D_1^{nc}) \leq \sum_{x \in [-r^n, r^n] - [-2n, 2n]} \frac{2^{H_2(q_0)(n-2)}}{4^n}.$$

Since in the set $[-r^n, r^n] - [-2n, 2n]$ there are less than $2r^n$ elements, we find

$$P(D_1^{nc}) \leq 2\left(\frac{r}{4}\right)^n 2^{H_2(q_0)(n-2)}.$$

The expression on the right side of the last equation above is an exponential negative bound, since by inequality 3.2.1, we have

$$\frac{r \cdot 2^{H_2(q_0)}}{4} \; < \; 1.$$

■

Lemma 3.2.4 *We have that*

$$P(D_2^n) \geq 1 - \frac{n}{4}\left(\frac{2^{H_2(q_0)}}{4}\right)^{(n-2)}.$$

Proof. Let $R : [0, n-1] \mapsto \mathbb{Z}$ be a (non-random) nearest neighbor path ending outside $[0, n-1]$. Assume first that $R(n-1) > n-1$. Then, since which each step R travels no

more than one unit, we get that $n-1-i < R(n-1-i)$ for all $i \in [0, n-1]$. Hence, since the scenery ξ is i.i.d., we find that $\xi_{n-1-i} = w_{n-1-i}$ is independent of

$$\xi(R_{n-1-i})\xi(R_{n-1-i+1})\dots\xi(R_{n-1}) \tag{3.2.7}$$

Let Z_i be the Bernoulli variable which is equal to 1 if $\xi(R_{n-1-i}) = w_{n-1-i}$ and $Z_i = 0$ otherwise. Because of the independence of expression 3.2.7, we get

$$P(Z_i = 1 | Z_{i-1}Z_{i-2}\dots Z_0) = 1/4.$$

It follows that the variables $Z_0Z_1\dots Z_n$ are i.i.d. so that

$$P(Z_0 = 1, Z_1 = 1, \dots, Z_n = 1) \;=\; \left(\frac{1}{4}\right)^n$$

But having all the Z_i's equal to 1 for $i = 1, 2, \dots, n$ is the same as saying that R generates the word w^n on the scenery ξ. Hence,

$$P(w^n = \xi(R_0)\xi(R_1)\dots\xi(R_n)) = \left(\frac{1}{4}\right)^n. \tag{3.2.8}$$

The last inequality was obtained assuming $R(n-1) > n-1$. The same inequality can be obtained for when $R(n-1) < 0$ and so inequality 3.2.8 holds for all nearest neighbor path not ending in $[0, n-1]$. Now, the event B_2^n is the event that there exits no nearest neighbor walk path $R \in \mathcal{P}_x^n$, with $x \in [-2n, 2n]$ and generating w^n on ξ whilst ending outside $[0, n-1]$. Hence

$$B_2^n = \bigcap_R \{\; w^n \neq \xi(R_0)\xi(R_1)\dots\xi(R_n) \;\}, \tag{3.2.9}$$

where the intersection is taken over all R in

$$\bigcup_{x\in[-2n,2n]} \mathcal{P}_x^n \tag{3.2.10}$$

ending outside $[0, n-1]$, i.e. such that $R(n-1) \notin [0, n-1]$. For those paths ending outside $[0, n-1]$, equation 3.2.8 applies. We can use this in conjunction with equation 3.2.9, (since in equation 3.2.9 all paths considered end outside $[0, n-1]$). We obtain:

$$P(B_2^{nc}) \leq \sum_R P(w^n = \xi(R_0)\xi(R_1)\dots\xi(R_n)) = \sum_R \left(\frac{1}{4}\right)^n. \tag{3.2.11}$$

where in the last sums above, R is taken over the set 3.2.10 and such that $R(n-1) \notin [0, n-1]$. The set $\mathcal{P}_x^n$ for given x contains less than $2^{(n-2)H_2(q_0)}$ elements. So the set 3.2.10 contains less than $4n2^{(n-2)H_2(q_0)}$ elements. Applying this to inequality 3.2.11, we get:

$$P(B_2^{nc}) \leq 4n\frac{2^{(n-2)H_2(q_0)}}{4^n}.$$

Note that the bound on the last inequality above is exponentially small in n since by 3.2.1, we have $(2^{H_2(q_0)}/4) < 0.5$ ■

Lemma 3.2.5 *We have that*

$$P(E^{nc}) \leq \frac{4n}{(v_1/2)^n},$$

for all n large enough.

Proof. As before, let ν_i denote the i-th visit by the random walk to the origin and define the following sequence. Let

$$k_1 = \nu_1$$

and, for $i \geq 1$, let

$$k_{i+1} := \min\{\nu_j \geq k_i + n : j \in \mathbb{N}\},$$

The sequence of k_i's denotes a set of visits by S to the origin, such that two consecutive visits are separated by at least n steps.

Let Y_i be a Bernoulli variable, where $Y_i = 1$ if after time k_i, S takes n steps to the right. and $Y_i = 0$ otherwise. Hence $Y_i = 1$ implies that

$$S_{j+1} - S_j = 1 \ , \ \forall j \in [k_i, k_i + n - 1].$$

The variables $Y_1, Y_2, \ldots$ are $i.i.d$ with

$$p = P(Y_i = 1) = \left(\frac{1}{2}\right)^n.$$

Note that among the first $(v_1)^n$ visits ν_i to the origin, there are at least $(v_1)^n/n$ visits k_i, and hence:

$$\Big\{ \sum_{i=1}^{v_1^n/n} Y_i \geq (v_2)^n \Big\} \subseteq E^n.$$

From the last inequality above it follows that

$$P(E^{nc}) \leq P\Big(\sum_{i=1}^{v_1^n/n} Y_i < (v_2)^n \Big) \tag{3.2.12}$$

At this point we simply use the Chebycheff inequality. Put

$$Z := \sum_{i=1}^{v_1^n/n} Y_i$$

so that

$$E[Z] = \frac{(v_1)^n}{n} E[Y_1] = \frac{(v_1)^n}{n} \left(\frac{1}{2}\right)^n = (1/n) \left(\frac{v_1}{2}\right)^n$$

and

$$VAR[Z] = \frac{(v_1)^n}{n} VAR[Y_1] = \frac{v_1^n}{n} \left(\frac{1}{2}\right)^n \left(1 - \frac{1}{2^n}\right) \leq \frac{(v_1)^n}{n} \left(\frac{1}{2}\right)^n. \tag{3.2.13}$$

The constants v_1 and v_2 do not depend on n and satisfy inequality 3.2.1, so that $v_1/2 > v_2$. It follows that for n large enough, $E[Z] = (v_1/2)^n/n$ is much larger than $(v_2)^n$. So for n large enough, we have

$$\left|(1/n)\left(\frac{v_1}{2}\right)^n - (v_2)^n\right| \geq (1/2n)\left(\frac{v_1}{2}\right)^n$$

and hence

$$|E[Z] - (v_2)^n| \geq (1/2n)\left(\frac{v_1}{2}\right)^n. \tag{3.2.14}$$

Applying now Chebycheff inequality to 3.2.12, we obtain

$$P(E^{nc}) \leq \frac{VAR[Z]}{(E[Z] - v_2^n)^2}.$$

Applying equation 3.2.13 and inequality 3.2.14 to the last inequality above we find

$$P(E^{nc}) \leq \frac{(v_1/2)^n}{(v_1/2)^{2n}}\frac{4n^2}{n} = \frac{4n}{(v_1/2)^n}. \tag{3.2.15}$$

Since by inequality 3.2.1 we have $v_1/2 > 1$ it follows that the bound on the right side of inequality 3.2.15, is an exponentially small quantity in n. ■

Lemma 3.2.6 *We have that*

$$P(F^{nc}) \leq (c_F)^n,$$

where $0 < c_F < 1$ *does not depend on* n.

Proof. For any integer $z \in [1, n-1]$, define the event

$$A_z = \{\xi(z+1) = \xi(z-1)\}.$$

Since the scenery-process $\{\xi\}_{z\in\mathbb{Z}}$ is a sequence of *i.i.d* random variables with uniform probability on a set of 4 colors, we get: $P(A_z) = 4\left(\frac{1}{16}\right) = \frac{1}{4}$.

Let X_z be the Bernoulli variable, such that $X_z = 1$ iff A_z holds. Note that any sequence of colors with size n has exactly $(n-2)$ possible pairs of positions for to A_z occurs, so that

$$\begin{aligned} P(F^{nc}) &= P(X_1 + X_2 + \cdots + X_{n-2} > (n-2)q_0) \\ &\leq P((X_1 - q_0) + \cdots + (X_{n-2} - q_0) \geq 0) \\ &\leq E(e^{Y_1 t})^{n-2}, \end{aligned}$$

where $Y_1 = (X_1 - q_0)$. Here we use the same argument as in the proof of lemma 3.2.5. That is we use that any random variable Z and for any $t > 0$, $P(Z \geq 0) \leq E[e^{Zt}]$. Since $q_0 > 0.25$, it follows that

$$E(Y_1) = E[X_1] - q_0 = 0.25 - q_0 < 0.$$

Hence, there exist a $t_0 \geq 0$ such that $E(e^{Y_1 t_0}) < 1$. Call this value c_F:

$$c_F := E(e^{Y_1 t_0}) < 1.$$

Thus we have an upper bound for $P(F^{nc})$ which decrease exponentially fast to zero:

$$P(F^{nc}) \leq c_F^n.$$

■

3.2.3 $P(B^{nc})$ exponentially small in n

In lemma 3.2.1, we prove that

$$C^n \cap D_1^n \cap D_2^n \cap E^n \cap F^n \subset B^n$$

It follows that

$$P(B^{nc}) \leq P(C^{nc}) + P(D_1^{nc}) + P(D_2^{nc}) + P(E^{nc}) + P(F^{nc}). \tag{3.2.16}$$

In the subsection 3.2.2, we get a negatively exponentially small in n upper bounds for each of the probabilities:

$$P(C^{nc}), P(D_1^{nc}), P(D_2^{nc}), P(E^{nc}), P(F^{nc}).$$

Hence, together with inequality 3.2.16, this implies that $P(B^{nc})$ is also exponentially small in n. Hence, there exists a constant $c_B > 0$ not depending on n such that for all n, we have

$$P(B^{nc}) \leq e^{-c_B n}.$$

Chapter 4

COMPUTER IMPLEMENTATION AND SIMULATIONS

In this section, we explain how to implement practically the algorithm and present the results of our computer simulations. It might be easy to draw a three colored tree with color pencils and then represent by hand the path R representing the scenery. But how should the computer represent the tree and the path R?

Once one starts programming a scenery reconstruction algorithm, things become more difficult in practice than they are in theory. Remember that our reconstruction algorithm first obtains words of length order $O(\ln n)$ and then assembles them. The piece of scenery one tries to reconstruct is of length order n.

In the last chapter we take the observations χ up to time $\mathcal{T}$, where $\mathcal{T}$ is of polynomial order in the size of the piece we want to reconstruct, *i.e.* in n. The polynomial bound for $\mathcal{T}$ is still large.

To see where the difficulty lies check the following: to reconstruct the words, one needs the random walk S to cross them in a straight way at least once up to time $\mathcal{T}$. Assume for example that the words have length 10. The probability that the random walk crosses a word of length 10 in a straight way, when it is located at the border of the word is 0.5^{10} and this is $1/1024$. So, for this event to have a good chance to happen, we need the random walk to come back to the border of that word about 1024 times. But for the random walk $\{S_t\}_{t\geq 0}$ to come back to a given point m times, we need to wait approximately order m^2 time. In our case, this leads to a time frame of order 1000000 for a straight crossing of a piece of length 10 to become likely. (Probably that million would need to be multiplied by a constant larger than one, in order to get the probability very close to one...)

Here we present an algorithm, which if the random walk walks only once through the piece it reconstructs something already close to the original scenery. With more colors it works better. Probably with 5 colors it should already be pretty good.

4.1 Algorithm: Reconstruction with Only One Visit

We describe different levels of precision for our reconstruction. All the reconstruction methods enumerated below do not reconstruct a piece of scenery exactly, but only up to a small number of missing letters. But the algebraic formalism presented here can be directly applied to program the algorithm presented in chapter 2.

Before describing how to implement the algorithm, we need to identify the vertexes on the tree in a way which our program can understand.

Vertexes as numbers We identify each vertexes v on the tree with a number. That number is equal to the observations made by the shortest path which joins the origin to v. See the Figure 4.1.

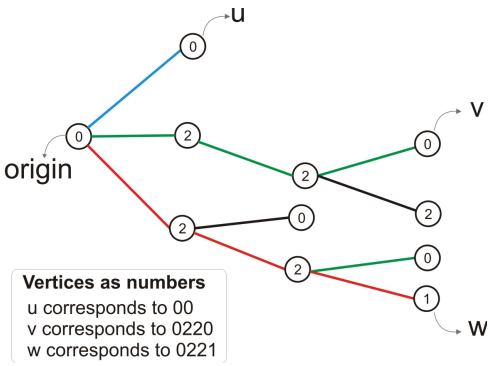


Figure 4.1: Vertexes as numbers.

Note that if the vertex v is visited at time s by $R \circ S$, then the color record which we observe when we move on the tree from the origin to v in a straight manner can be obtained from the observations χ up to time s. More precisely, take $\chi_0\chi_1 \dots \chi_s$ modulo $aba = a$. By this we mean that we need to replace any sub-string of the form aba by a, until the string can not be simplified any further. The order does not matter, we always will get the same result in the end. This "maximally simplified string" is equal to the color record seen when moving from v_0 to v in a straight manner.

Note that the vertexes visited by $R \circ S$ are also the vertexes visited by R.

Let χ_0^t be the observations from 0 to t, so that: $\chi_0^t = \chi_0\chi_1\chi_2 \dots \chi_t$, and denote by $\bar{\chi}_0^t$ the string χ_0^t to which we applied the transformation $aba = a$ as many times as possible until no more. We call this transformation the *rule for maximal simplification*. Let us show how the transformation works with a simple example:

Example 4.1.1 *Suppose* $\chi_0^t = 131210$. *Here with our rule for maximal simplification we apply the "simplification* $(aba = a)$*" twice on* χ_0^t. *Hence:*

$$\chi_0^t = 131210 \overset{(131=1)}{\longrightarrow} 1210 \overset{(121=1)}{\longrightarrow} 10 = \bar{\chi}_0^t.$$

The algorithm uses the *rule for maximal simplification* in the following way: When the rule is applied on $\chi_t = \psi(R(S(0)))\psi(R(S(1)))\dots\psi(R(S(t)))$, then it produces the skeleton of $R \circ S$ on the tree without branches. So using the rule $aba = a$ until $s < t$, the simplified observations χ_0^{-s}, gives the number corresponding to the vertex where $R \circ S$ is at time s.

Hence to get all the sequence of vertexes visited by $R \circ S$ up to time T, we computer for every $t \leq T$ the string χ_0^{-t}, and every time we get a new string which did not appear before, we record it. Let us show it with the next example.

Example 4.1.2 *Suppose we have the sequence of observations:*

$$\chi = 010121222020...$$

so that:

$\bar{\chi}_0^t$	0	01	0	01	012	01	012	0122	012	0120
t	0	1	2	3	4	5	6	7	8	9

thus

$$v_1 = 0, v_2 = 01, v_3 = 012, v_4 = 0122, v_5 = 0120, ...$$

Look Figure 4.2 following the blue tube. Also observe that the vertexes will be find in order of the first visit and besides the j*-th vertex visited by* $R \circ S$ *during time* $[0, T]$ *is equal to* v_j*, the* j*-th vertex visited by* R *after* 0. *Check out Figure 4.2 following the red tube.*

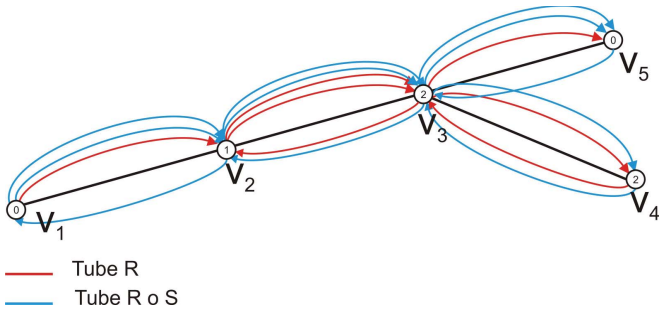


Figure 4.2: Representation of the vertexes as numbers on R and $R \circ S$.

We use first only the observations χ_0^T as input for our reconstruction algorithm.

4.1.1 First approximation

The first and worse level of the algorithm is simply apply the transformation $aba = a$ to the observations χ_0^T. For this case, the output of the algorithm will be χ_0^{-T}. We obtain a finite piece of the scenery with gaps, because χ_0^{-T} equal to the simplification modulo $aba = a$ of the string $\xi_0\xi_1 \dots \xi_y$. Here we assume that $S_T = y$.

To see this, observe that since χ_0^T is equal to $\xi(S_t)$ for $0 \leq t \leq T$, then

$$\{\xi_0^y\} \subseteq \{\chi_0^T\},$$

thus every time that any portion like aba is in $\{\xi_0^y\}$, it should be also in $\{\chi_0^T\}$. On the other hand, the part of the sequence in $\{\chi_0^T\}$ which is not in $\{\xi_0^y\}$ are subsequences of the form aba too, that is because each of these subsequences was just produced when the random walk made a "back forth step". Note that the rule $aba = a$ applied on each of these subsequences give only the first letter from each of them respectively, which corresponds to a letter from the original sequence. So $aba = a$ applied on this part gives just $\{\xi_0^y\}$, the original sequence.

The information "χ_0^{-T}" is called "fingerprint of ξ_0^y" and does not depend on the random walk path.

Let us look this with a numerical example.

Example 4.1.3 *Take the scenery ξ between 0 and 7 be equal to:*

$\xi(z)$	...	0	1	2	1	2	2	2	0...
z	:::	0	1	2	3	4	5	6	7...

Suppose the random walk walks twelve steps producing the sequence of observations

$$\chi_0^{12} = 010121222020...$$

Let us process the piece of scenery ξ_0^7 and the sequence the observations χ_0^{12} by replacing stings aba by a, i.e,

$$\xi_0^7 = 01212220 \overset{(aba=a)}{\longrightarrow} 012220 \overset{(aba=a)}{\longrightarrow} 0120$$

$$\chi_0^{12} = 010121222020 \overset{(aba=a)}{\longrightarrow} 0121222020 \overset{(aba=a)}{\longrightarrow} 01222020 \overset{(aba=a)}{\longrightarrow} 012020 \overset{(aba=a)}{\longrightarrow} 0120.$$

We observe how the rule for maximal simplification gives the same result whether applied to the original sequence and the sequence of observations.

We assemble the sequences of vertexes as number in the same order as they were appearing. This leads to the partial reconstruction $\hat{\xi} = 0120$, which has several "wholes".

4.1.2 Algorithm 1

The second level is recovering two back-bones of $R \circ S$ on the tree. For this we not only need to check the vertexs on the tree which get be visited by $R \circ S$, but also select the side of the tree where they come from.

In [19] Matzinger and Roles show by Lemma 5.3 that there exist two infinite connected components from the set of vertex visited by R. We use this fact in order to make a decision criterion to separate the vertexes in practice. The criterion is as follow:

Fix a subsequence of length h from the sequence corresponding to the color record from each vertex, call this subsequence as "prefix". Then separate the vertexes in groups such that, all the vertexes from the same group have the same prefix. Finally we choose the two biggest groups, one of both will correspond to the right backbone and the another to the left backbone.

Look the Figure4.3. Now assume we already sorted the vertexs and obtained two se-

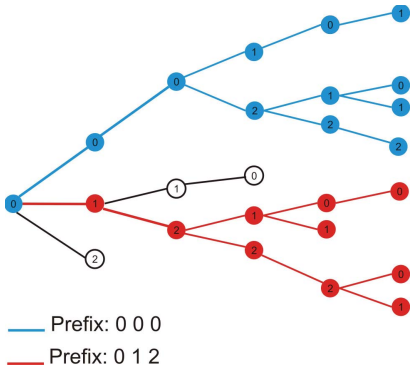


Figure 4.3: Right and left side, respectively represented by the blue and red group.

quences of vertexs $v_{1r}, v_{2r}, \ldots, v_{jr}$ and $v_{1l}, v_{2l}, \ldots, v_{jl}$, one corresponds to the right part of the scenery and the another to the left side. The increasing order corresponds to the order in which the vertexes were visited by $R \circ S$.

Let t_{ir} denote the first time that $R \circ S$ visits the $i-$th vertex on the right subtree, that is the time when v_{ir} appears, and let s_{ir} be the last time that $R \circ S$ visits the $i-$th vertex before time $t_{(i+1)r}$. Similarly define the times for the left subtree, i.e. t_{il} and s_{il}.

Denote by $\chi_{s_{ir}}^{t_{(i+1)r}}$ the observations between s_{ir} and $t_{(i+1)r}$, that is

$$\chi_{s_{ir}} \chi_{s_{ir}+1} \chi_{s_{ir}+2} \cdots \chi_{t_{(i+1)r}},$$

then for $ir = 1r, \ldots, (j-1)r$ apply once more time the rule for maximal simplification to each from these subsequences.

So our attempt to reconstruct the right part of the scenery is the concatenation

$$\chi_{s_{1r}}^{t_{2r}} \chi_{s_{2r}}^{t_{3r}} \cdots \chi_{s_{(j-1)r}}^{t_{jr}}.$$

Similarly, for $il = 1l, \ldots, (j-1)l$ the attempt to reconstruct the left part will be

$$\chi_{s_{1l}}^{t_{2l}} \chi_{s_{2l}}^{t_{3l}} \cdots \chi_{s_{(j-1)l}}^{t_{jl}}.$$

Of course we won't know which one is right or left. We only put the two pieces together. They could have some missing parts because when we apply the rule $aba = a$, the final sequence could be too compressed.

4.1.3 Algorithm 2

The third possibility is similar to the second one, however here we try to solve the problem of the missing parts.

Once more we use the representation of the vertexes as numbers and classify the vertexes according to the prefix criterion to find the two groups which contain most vertexes.

The new idea here is to choose the shortest path made by $R \circ S$ up to time T between two consecutive vertexes from the same side. That is we compute the shortest paths between each of the following couples: $(v_{1r}, v_{2r}), (v_{2r}, v_{3r}), \ldots$ and $(v_{(j-1)r}, v_{jr})$.

Finally we concatenate these shortest paths. This can be done with both collection of vertexes: those on the left side of the tube and those on the right side.

Note that both algorithms need to be improved at the origin because to use the prefix idea to separated the vertexes in two groups and hence in most cases the vertexes near to the origin can not be classified. Near to the origin, we do not have enough information. That is, the sequences which identify the vertexes close to the origin are smaller than the prefixes we consider.

4.2 Alignment between $\hat{\xi}$ and ξ

Having obtained by the reconstruction algorithm a sequence which we call $\hat{\xi}$. We hope it to be a good approximation of a portion of the scenery close to the origin. We use sequence analysis in order to compare $\hat{\xi}$ and ξ.

Sequence analysis is used to identify regions of similarity between two sequences, (see

Page. 12 in [4]). More explicitly, they are methods and algorithms to solve the problem of deciding if the sequences are either structurally or evolutionary related and if yes how similar they are. Sequence analysis is used in Bio-informatics for comparing biological sequences for example DNA, RNA or protein. Also these methods are used for non-biological sequences as financial data, natural language processing or in social sciences. Here, we use it to compare $\hat{\xi}$ with ξ

There are two general categories, *global alignments* and *local alignments.* The first one is used to produce a global optimization: it produces an alignment in the entire length of the two sequences. By contrast, the second one identifies regions of similarity between subsequences. So, global alignments are more useful when the sequences are roughly of equal size. On the other hand, local alignments are used when the sequences are dissimilar but it is suspected they contain regions of similarity.

We will use local alignments to compare $\hat{\xi}$ and ξ. We need to compare the original sequence with the finite sequence obtained by the algorithm, which is typically only an approximation of a portion from the first one. In [30], Smith and Waterman present an algorithm to find a pair of segments, one from each of two long sequences producing the best local alignment score. We explain the algorithm in the Appendix at the end of this document.

4.2.1 Results: Alignment between ξ and $\hat{\xi}$

We simulated ten random sceneries ξ's and ten simple random walks of size 5000. We applied Algorithm 1 and Algorithm 2 to the sequences of observations $\chi = \xi \circ S$ in order to produce the sequences $\hat{\xi}$'s, i.e, the estimations of ξ's around the origin.

We use the program "JAligner", which is an open source Java implementation of Smith-Waterman algorithm [30] with Gotoh's improvement [5]. The author of the program is Ahmed Moustafa and it is avalaible in [25].

This tool is used to compare two sequences when the goal is trying to find the best region of similarity between them.

The score matrix used is the *match-matrix*. It gives the value equal to one when a match holds and menus one when is a mitmatch. For the case with three colors the score matrix is as follow:

	$-$	c_1	c_2	c_3
$-$	0	0	0	0
c_1	0	$+1$	-1	-1
c_2	0	-1	$+1$	-1
c_3	0	-1	-1	$+1$

The gap scoring system is an affine gap cost structure like in A.1.2, whit *gap-open* penalty

$o = 1$ and *gap-extension* penalty $e = 0.5$.

For the biggest sequence estimated of the two sides, right and left, we present on the next tables the proportion of matches or similarity, the proportion of gaps and the score of the longest common subsequence LCS for each of one of the ten simulations.

Length(ξ)	Length($\hat{\xi}$)	Similarity(%)	Gaps(%)	Score
53	77	(68.83)	(31.17)	38.50
34	43	(79.07)	(20.93)	28.00
70	91	(76.92)	(23.08)	55.50
42	48	(87.50)	(12.50)	37.50
96	130	(73.85)	(26.15)	74.50
34	46	(73.91)	(26.09)	26.50
73	103	(70.87)	(29.13)	53.00
112	138	(81.16)	(18.84)	94.00
57	83	(68.67)	(31.33)	41.00
72	90	(80.00)	(20.00)	60.00

Table 4.1: Results obtained by Algorithm 1

Length(ξ)	Length($\hat{\xi}$)	Similarity(%)	Gaps(%)	Score
57	77	(74.03)	(25.97)	44.50
38	43	(88.37)	(11.63)	34.50
79	92	(85.87)	(14.13)	69.00
42	48	(87.50)	(12.50)	37.50
100	132	(75.76)	(24.24)	79.00
36	46	(78.26)	(21.74)	30.00
118	172	(68.60)	(25.00)	72.50
123	143	(86.01)	(13.99)	108.5
59	83	(71.08)	(28.92)	44.50
78	98	(79.59)	(20.41)	65.00

Table 4.2: Results obtained by Algorithm 2

In general we can observe according with the proportion of matches (similarity), the Algorithm 2 obtain better results than the Algorithm 1. Only in two of the ten cases does Algorithm 1 show higher similarities between $\hat{\xi}$ and a portion of ξ close to the origin. However they don't overpass two percentage points, 70.87 *vs* 68.60 and 80.00 *vs* 79.59, (see *similarity* for the cases 7 and 10 in 4.1 and 4.2) .

The Algorithm 2 also presents better results to see the proportion of gaps. Only in simulation number 10 does Algorithm 1 obtain a lower proportion, but it is also 20.00 *vs*

20.41, (see *Gaps* for the cases 10 in 4.1 and 4.2).

The length of ξ is computed up to the first match with $\hat{\xi}$, thus the part around the origin which can not be estimate by these algorithms is not taking into account the calculation of the proportion of matches and gaps between the two sequences.

An improvement

With both algorithms we are observing paths between consecutive vertexes, consecutive in the order they appear, i.e, paths between pairs of vertexes (v_i, v_{i+1}). When they are consecutive on the tree, that is when the shortest path by R between them has length one, then the estimated sequence using any of these algorithms will simply be $\psi(v_i), \psi(v_i+1)$. These are the colors assigned by ψ to these pairs of vertexes. But, what happen if after R has visited v_i, it visits others vertexes before visiting v_{i+1}?

For this problem the above algorithms are not equipped. We propose the follow improvement:

Take all the path by $R \circ S$ between v_{ir}, v_{ir+3}, for $ir = 1, \ldots, jr-3$, i.e,

$$\{s < t < \mathcal{T}; R \circ S(s) = v_{ir}, R \circ S(t) = v_{ir+3}, R \circ S(u) \neq v_{ir} \text{ for } u \in [s+1, t-1]\} \text{ or}$$

$$\{s < t < \mathcal{T}; R \circ S(s) = v_{ir+3}, R \circ S(t) = v_{ir}, R \circ S(u) \neq v_{ir+3} \text{ for } u \in [s+1, t-1]\},$$

then between them choose the shortest path according with the length between the first and last time that $R \circ S$ visits v_{ir+2}, i.e,

$$\min\{|l-f|; R \circ S(f) = v_{ir+2}, R \circ S(l) = v_{ir+2}, s < f < l < t\},$$

to obtain the word $\psi(R \circ S(f)), \psi(R \circ S(f+1)), \ldots, \psi(R \circ S(l))$, which is the part of R after the first visit to v_{ir+2} and before the first visit to v_{ir+3}. This is not captured by Algorithm 1 and Algorithm 2.

We add these subsequences between the sequences obtained by any of the algorithms above between (v_{ir+1}, v_{ir+2}) and (v_{ir+2}, v_{ir+3}), for $ir = 1, \ldots, jr-3$. The same is made to the left side.

The idea of the above process come from the expectation of the number of additional steps that a simple random walk takes when it first enters state i until it enters state $i+1$.

Consider $\{S_t\}_{t \geq 0}$ a Markov chain with space state $\{0, 1, 2 \ldots\}$ having the transition probabilities $p_{0,1} = 1$, $p_{i,i+1} = 2/3$, $p_{i.i-1} = 1/3$, $i \geq 1$. Let N_i denote the number of additional transitions that the Markov chain takes when it first enters state i until it enters state $i+1$. By the Markov property, it follows that these random variables N_i, $i \geq 0$ are independent.

Let $\mu_i = E(N_i)$, then upon conditioning on the next transition after the chain enters state i, $i \geq 1$,

$$\mu_i = 1 + \frac{1}{3}E(N_{i-1} + N_i).$$

Hence,

$$\begin{aligned} \mu_i &= 1 + \frac{1}{3}(\mu_{i-1} + \mu_i) \text{ or} \\ \mu_i &= \frac{3}{2} + \frac{1}{2}\mu_{i-1},\ i \geq 1. \end{aligned}$$

Starting with $\mu_0 = 1$, we obtain from the preceding recursion that

$$\mu_i = 3 - 2\Big(\frac{1}{2}\Big)^i.$$

So $\mu_1 = 2, \mu_2 = 2.5, \mu_3 = 2.75$ and μ_n goes to 3 when n goes to infinity. For example on the state n, i.e, $\{S_t\}_{t\geq 0} = n$, we expect 3 steps before $\{S_t\}_{t\geq 0} = n+1$, for n large enough.

Suppose that at time l $S_l = n$, then 3 steps until S enters state $n+1$ are only possible if $S_{l+1} = n-1, S_{l+2} = n, S_{l+3} = n+1$. Let $W_i = \{v; |v_0 - v| = i\}$ be the subset of vertex from V_T such that, the distance from them to the origin is equal to i, then at time l $R(l) = w_n$ with $w_n \in W_n$. So 3 steps until S enters state $n+1$ are only possible if $R(l+1) = w_{n-1}, R(l+2) = w_n, R(l+3) = w_{n+1}$, with $w_{n-1} \in W_{n-1}$ and $w_{n+1} \in W_{n+1}$. Thus for any pair of consecutive vertexes on the tree, (v_i, v_{i+1}) and $R = v_i$, we expect at most only one back-forward step before R reaches v_{i+1}.

That is the reason why we are choosing the paths by $R \circ S$ between v_i, v_{i+3} and between them, to choose the shortest path according with the length between the first and last time that $R \circ S$ visits v_{i+2}.

Results: We used once more the program "JAligner" with the same score-matrix and gap scoring system. The Tables 4.3 and 4.4 show the results.

We observe percentages of similarity higher and a lower percentage of gaps in comparison with what was obtained by the algorithms without improvement. However. we note that some cases like (5,8,10) in Table 4.4 exist, where the similarity is lower and the gaps higher. This is because we have made the improvement for the paths between (v_{ir}, v_{ir+3}) for $ir = 1, \ldots, jr-3$. What is happening here is that when the vertexes are not consecutive on the tree and the distance between (v_{ir}, v_{ir+3}) is too large, the added word $\psi(R \circ S(f)), \psi(R \circ S(f+1)), \ldots, \psi(R \circ S(l))$ could be also too large, in this situation the improvement does not work as we had hoped.

Length(ξ)	Length($\hat{\xi}$)	Similarity(%)	Gaps(%)	Score
65	82	(79.27)	(14.63)	50.00
40	49	(81.63)	(18.37)	34.00
77	97	(79.38)	(19.59)	62.50
46	48	(95.83)	(4.17)	44.50
127	177	(71.75)	(25.42)	93.50
36	46	(78.26)	(21.74)	30.00
140	194	(72.16)	(19.59)	93.50
155	213	(72.77)	(19.25)	105.0
69	95	(72.63)	(27.37)	53.00
106	150	(70.67)	(20.00)	68.00

Table 4.3: Results obtained by the improvement of Algorithm 1

Length(ξ)	Length($\hat{\xi}$)	Similarity(%)	Gaps(%)	Score
68	82	(82.93)	(9.76)	54.50
44	49	(89.80)	(10.20)	40.50
	86	(87.76)	(11.22)	76.50
46	48	(95.83)	(4.17)	44.50
130	181	(71.82)	(25.97)	95.50
38	46	(82.61)	(17.39)	33.50
199	333	(59.76)	(31.53)	90.50
164	233	(70.39)	(27.90)	115.5
71	95	(74.74)	(25.26)	56.50
109	161	(67.70)	(26.09)	69.00

Table 4.4: Results obtained by the improvement of Algorithm 2

Appendix A

LOCAL ALIGNMENT

The Smith-Waterman algorithm compares segments of all possible lengths and optimizes the similarity measure. To find the optimal local alignment it uses a scoring system, which is composed by a substitution matrix and a gap-scoring scheme.

A.1 Scoring Model

In Biology, the basic idea of comparing sequences is looking for evidence that they come from a common ancestor by a process of mutation and selection. We are not explain in detail what this process mean in genetic, but what it is in terms of sequences is represented by mismatches produced by a "substitution" process, which changes letters in a sequence and by gaps produced by either an "insertion" or "deletion" process, which add or remove letters.

Before to explain the alignment algorithm is necessary to define a scoring model. A scoring model is a function that assign a score to each possible pair of letters. Usually a measure of the relative likelihood that the sequences are related as opposed to being unrelated is used. In other words, a probability to the alignment in each of the two cases, related and unrelated, is founded and the reason of the probabilities is considered.

Let x and y a pair of sequences of length m and n, respectively. Let x_i the iTh symbol in x and y_j the jth symbol of y. These symbols come from some alphabet A, in our case this alphabet is just the set of colors $C = \{0, 1, \ldots, c-1\}$.

The probability for the unrelated or random case R assumes that the letter $a \in A$ occurs independently with probability q_a, and hence the probability of the alignment is just the product of the probabilities of each sequence

$$P(x, y|R) = \Pi_i q_{x_i} \Pi_j q_{y_j}.$$

The model M which assume that the sequences are related, assign join probabilities to aligned pairs, i.e. p_{ab}, Then suppose that the letters a and b have each independently been derived from some unknowing original letter c in their common ancestor, where c might be the same as a or b. Thus, the probability for the whole alignment is

$$P(x, y|M) = \Pi_i p_{x_i y_i}.$$

The ratio of these two likelihoods is called *odds ratio* and is just

$$\frac{P(x, y|M)}{P(x, y|R)} = \frac{\Pi_i p_{x_i y_i}}{\Pi_i q_{x_i} \Pi_j q_{y_j}} = \Pi_i \frac{p_{x_i y_i}}{q_{x_i} q_{y_i}}.$$

To obtain an additive scoring system, the logarithm of this ratio is taken, it is call *log-odds ratio*, and then produce the sore

$$S = \sum_i s(x_i, y_i),$$

where

$$s(a, b) = log\Big(\frac{p_{ab}}{q_a q_b}\Big),$$

is the log likelihood ratio of the pair (a, b) occurs as an aligned pair, as opposed to an unaligned pair.

The $s(a, b)$ scores can be arranged in a matrix for each aligned pair of letters, then at the position k, l of this matrix will correspond to the score $s(a_k, a_l)$, where a_k, a_l are the kth and lth letter of the alphabet A. This matrix is known as a *score matrix* or a *substitution matrix*

To penalize the cost associated with a gap of length g, a linear scores can be used and it is given by

$$\gamma(g) = -gd, \tag{A.1.1}$$

where d is a penalty per unit length of gap. In some cases, when is more likely to have a large gap, rather than many small gaps, that is when is much more likely to have one big gap of length 10 than to have 10 small gaps of length 1, a gap opening penalty, o, and a gap extension penalty, e are used. The score for a gap of length l is

$$\gamma(g) = -o - (l - 1)e, \tag{A.1.2}$$

where o is the *gap-open* penalty and e the *gap-extension* penalty.

The total score assigned to an alignment is the sum of terms for each aligned pair, plus terms for each gap. This additive scoring scheme suppose that the mutations or gaps at different sites in a sequence occurs independently.

If the interest is only to count the amount of aligned pairs, that is the longest common subsequence (LCS), then the gap score is equal to zero and the score matrix will be the identical matrix.

A.2 Local Alignment: Smith-Waterman Algorithm

The algorithm build recursively a matrix F, it is known by substitution matrix, where $F(i.j)$ is the score of the best alignment between the subsequence of x up to i, i.e. $x_{1\ldots i}$ and the subsequence of y up to j, i.e. $y_{1\ldots j}$. Let $F(0,0) = 0$, then if $F(i-1,j-1), F(i-1,j)$ and $F(i,j-1)$ are known, it is possible to calculate $F(i,j)$. Note that there are three ways to get the best score $F(i,j)$, they are:

1. x_i could be aligned to y_j, in which case $F(i,j) = F(i-1,j-1) + s(x_i, y_j)$
2. x_i is aligned to a gap, in which case using the simplest case (A.1.1), $F(i,j) = F(i-1,j) - d$, or
3. y_j is aligned to a gap, in which case $F(i,j) = F(i,j-1) - d$.

The best score up to (i,j) will be the largest of these three options and zero, that is:

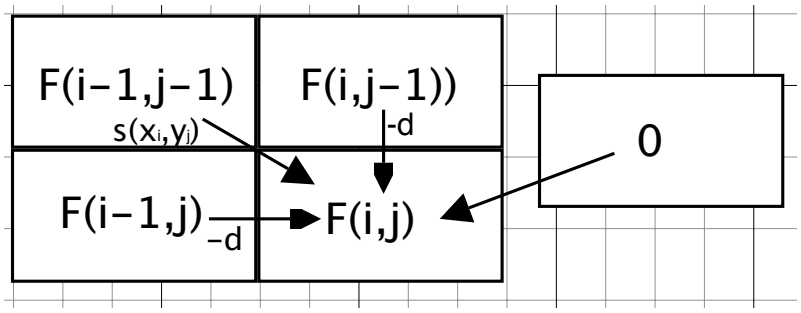
$$F(i,j) = max \left\{ \begin{array}{l} 0, \\ F(i-1,j-1) + s(x_i, y_j), \\ F(i-1,j) - d, \\ F(i,j-1) - d. \end{array} \right\}$$

Note that $F(i,j)$ takes the value 0 if all other options have values less than 0, thus if the best alignment up to some point has a negative score, then is better to start a new alignment. For this reason, the values $F(i,0)$ and $F(0,j)$ which represent alignments with gaps in y and x respectively, take the values $F(i,0) = F(0,j) = 0$, i.e, the top row and the left column in F will be filled with 0s.

Thus, the matrix F is filling from top left to bottom right following the next process: For each square of four cells, $F(i,j)$, $F(i-1,j-1), F(i-1,j)$ and $F(i,j-1)$, $F(i,j)$ is obtained by the maximizing process (A.2). Observe the next diagram.

In the case of a gap structure like (A.1.2), the recurrence relation at this case is:

$$F(i,j) = max \left\{ \begin{array}{l} 0, \\ F(i-1,j-1) + s(x_i, y_j), \\ F(k,j) + \gamma(i-k), \ k = 0, \ldots, i-1 \\ F(i,k) + \gamma(j-k), \ k = 0, \ldots, j-1. \end{array} \right\}$$



A dynamic programming for a gap structure (A.1.2) is implemented in order to reduce the time of the operation, (see pag. 29 in [4]). It is called Alignment with affine gap.

A.2.1 The traceback process

The alignment find the pair of segments with maximum similarity. For that a process known as *traceback* is used. The dynamic of such a process is as follow: It starts from a fixed cell of F and follow in reverse way the pointers to build the matrix F, i.e, at each step the traceback process, it moves back from the current cell (i, j) to the cell from it was derived, that is to one of the cells $(i-1, j-1), (i-1, j)$ or $(i, j-1)$. Then if the step was to $(i-1, j-1)$ the current alignment is between x_i and y_j, but if the step was to $(i-1, j)$ the alignment is between x_i and a gap ' $-$, and finally if the step was to $(i, j-1)$ the alignment is between a gap ' $-$ and y_j . If at any point two of the derivations are equal, an arbitrary choice is made.

Now we are already to explain the local alignment algorithm.

1. Find the highest value of $F(i, j)$ over the whole matrix.

2. Start a traceback process from there, ending with an element of$F$ equal to zero.

The next example is showed in ([30]): In this example the parameters $s(aibj)$ and deletions of length k with a weight W_k were chosen on an a priori statistical basis. A match has a value of unity while a mismatch produced a minus one-third value. The local dynamic programming matrix for the example sequences is as follow.

Table A.1: The underlined elements indicate the trackback path from the maximal element 3.3

	g	C	A	G	C	C	U	C	G	C	U	U	A	G
g	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
A	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
A	0.0	0.0	1.0	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.7
U	0.0	0.0	0.0	0.7	0.3	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.7
G	0.0	0.0	0.0	<u>1.0</u>	0.3	0.0	0.0	0.7	1.0	0.0	0.0	0.7	0.7	1.0
C	0.0	1.0	0.0	0.0	<u>2.0</u>	1.3	0.3	1.0	0.3	2.0	0.7	0.3	0.3	0.3
C	0.0	1.0	0.7	0.0	1.0	<u>3.0</u>	1.7	1.3	1.0	1.3	1.7	0.3	0.0	0.0
A	0.0	0.0	2.0	0.7	0.3	<u>1.7</u>	2.7	1.3	1.0	0.7	1.0	1.3	1.3	0.0
U	0.0	0.0	0.7	1.7	0.3	1.3	<u>2.7</u>	2.3	1.0	0.7	1.7	2.0	1.0	1.0
U	0.0	0.0	0.3	0.3	1.3	1.0	2.3	<u>2.3</u>	2.0	0.7	1.7	2.7	1.7	1.0
G	0.0	0.0	0.0	1.3	0.0	1.0	1.0	2.0	<u>3.3</u>	2.0	1.7	1.3	2.3	2.7
A	0.0	0.0	1.0	0.0	1.0	0.3	0.7	0.7	2.0	3.0	1.7	1.3	2.3	2.0
C	0.0	1.0	0.0	0.7	1.0	2.0	0.7	1.7	1.7	3.0	2.7	1.3	1.0	2.0
G	0.0	0.0	0.7	1.0	0.3	0.7	1.7	0.3	2.7	1.7	2.7	2.3	1.0	2.0
G	0.0	0.0	0.0	1.7	0.7	0.3	0.3	1.3	1.3	2.3	1.3	2.3	2.0	2.0

Bibliography

[1] A. Hart, F. Machado, H. Matzinger(2008) Information recovery from observations by a random walk having jump distribution with exponential tails. in review

[2] A. Hart, H. Matzinger (2006). Markers for error-corrupted observations. *Stochastic Process. Appl.* **116** 807–829

[3] I. Benjamini, H Kesten (1996) Distinghuishing sceneries by observing the scenery along a random walk path. *J Anal Math* **69**, 97–135.

[4] I. Durbin, S. Eddy, A. Krogh, G. Mitchison (1998) Biological sequence analysis. *Cambridge University Press.*

[5] O. Gotoh (1982) An improved algorithm for matching biological sequences. *Journal of Molecular Biology.* **162(3)** 705–708.

[6] C. Howard (1997). Distinguishing certain random sceneries on $\mathbb{Z}$ via random walks. *Statist. Probab. Lett.* **34** 123–132.

[7] Kalikow (1982) $T, T^{-}1$ transformation is not loosely Bernoulli. *Annals of Mathematics. Second Series.***115** 393–409.

[8] I. Karatzas, S.E. Shreve (1991) Brownian Motion and Stochastic Calculus. *Springer. Second Edition.*

[9] M. Keane, W. den Hollander (1986). Ergodic properties of color records. *Phys. A* **138** 183–193.

[10] H. Kesten (1996). Detecting a single defect in a scenery by observing the scenery al a random walk path. *Itô's Stochastic Calculus and Probability Theory* 171–183.Springer, Tokyo.

[11] J. Lember. and H. Matzinger (2008) Information recovery from a randomly mixed up message-text. *Electronic Journal of Probability* **13** 396–466.

[12] J. Lember. and H. Matzinger (2003) Reconstructing a 2-color scenery by observing it along a recurrent random walk path with bounded jumps. *Eurandom. In preparation.*

[13] A. Lenstra, H. Matzinger (2001). Reconstructing a 4-color scenery by observing it along a recurrent random walk path with unbounded jumps. Eurandom.

[14] Lindenstrauss (1999) Indistinguishable sceneries. *Random Structures Algorithms.* **14** 71–86.

[15] M. Löwe, H. Matzinger(2003) Reconstruction of sceneries with correlated colors. *Stochastic Process. Appl.* **105** 175–210.

[16] M. Löwe, H. Matzinger (2002) Scenery reconstruction in two dimensions with many colors. *Ann. Appl. Probab.* **12(4)** 1322–1347.

[17] M. Löwe, H. Matzinger, F. Merkl (2004) Reconstructing a multicolor random scenery seen along a random walk path with bounded jumps. *Electronic Journal of Probability.* **15** 436–507.

[18] H. Matzinger (1999) *Reconstructing a 2-color scenery by observing it along a simple random walk path with holding. Ph.D. thesis, Cornell University.*

[19] H. Matzinger (1999) Reconstructing a three-color scenery by observing it along a simple random walk path. *Random Structures Algorithms* **15(2)**, 196–207.

[20] H. Matzinger (2005) Reconstructing a 2-color scenery by observing it along a simple random walk path. *Ann. Appl. Probab.* **15**, 778–819.

[21] H. Matzinger, S.W.W. Rolles (2006) Retrieving random media. *Probab. Theory Related Fields.* **136**, 469–507.

[22] H. Matzinger, S.W.W. Rolles (2006) Finding blocks and other patterns in a random coloring of $\mathbb{Z}$. *Random Structures Algorithms.* **28**, 37–75.

[23] H. Matzinger, S.W.W. Rolles (2003) Reconstructing a random scenery observed with random errors along a random walk path. *Probab. Theory Related Fields.* **125(4)**, 539–577.

[24] H. Matzinger, S.W.W. Rolles (2003) Reconstructing a piece of scenery with polynomially many observations. *Stochastic Process. Appl.* **107(2)**, 289–300.

[25] A. Moustafa (1981) JAligner. *http://jaligner.sourceforge.ne*

[26] S.V. Nagaev (1979) Large Deviations of Sums of Independent Random Variables. *The Annals of Probability* **7**, 745–789.

[27] A. Pachon, S. Popov (2008). Scenery reconstruction with branching random walk *Submited..*

[28] S. Popov, H. Matzinger (2008). Detecting a local perturbation in a continuous scenery *Electronic Journal of Probability.* **12** 1103–1120.

[29] I.S. Shiganov (1986) A refinement of the upper bound of the constant in the remainder term of the Central Limit Theorem. *Stability Problems for Stochastic Models, Moscow, 1982*, 109–115, (in Russian); *English translation, J.Soviet Math* **35**, N 3

[30] TF. Smith, MS. Waterman (1981) Identification of Common Molecular Subsequences. *Journal of Molecular Biology.* **147(1)** 195–197.