Københavns Universitet

Niels Bohr Instituttet

In close cooperation with UC Davis Complexity Science Center

Irreversibility in Complex Processes

Author:
Gorm Gruner Jensen
qgf524@ku.alumni.dk

Supervisor:
Namiko Mitarai

June 11, 2014

Abstract

The goal of this project is to develop a theory to describe irreversibility of one-dimensional, complex processes. In particular the focus will be on stationary processes in discreet time over a finite alphabet. It is shown how such a process can be described by an ε-machine - a unique, minimal, unifilar, generating Hidden Markov Model. The question of irreversibility is then recast in terms of ε-machines. To get to an answer, an algorithmic approach to finding ε-machines in general, and reverse ε-machines in particular, is required. To meet this purpose the Mixed State Algorithm is presented, and carefully studied. Finally a small handful of examples are analysed to get an idea of the diversity of irreversibilities that a process can have, and an exhaustive survey of irreversibility is performed of all processes whose ε-machines have six states or less.

Contents

1 Motivation

How much can we learn from measurements? In physics - in all of science - we take pride in claiming that a theory is only as good as the data that supports it. A theory can only be 'scientific' if there is an experiment to test it. However, many important physical theories are expressed in terms of continuous variables. In theoretical classical mechanics, electro dynamics, or even quantum mechanics, systems are assumed to have continuous space and change in continuous time. We celebrate these theory both for their beauty and for the amazing accuracy with which they predict the world around us. But, we do not design continuous experiments. No one have ever taken a continuous series of data points, or even written down a single irrational number. In fact we are almost always performing our experiments using digital, binary computers. When we measure the intensity of light from a super nova or the current running through a neuron, what we get is not a smooth curve, but a large list of 0's and 1's. Between us and the world is a measurement. Maybe what is out there is beautiful and continuous, but all we get are lists of numbers. Therefore the question naturally arises:

What can we really learn from a long list of 0's and 1's?.

In an attempt to approach an answer to this question we can try to study *Processes*. A process could for example be a time sequence of measurements. If there is one thing we think we know about time, we know it's running - but only one way. What happened yesterday can neither be undone today nor tomorrow, or in other words, time seem to be irreversible.

2 Processes and generators

By process we will understand an ordered sequence of random variables $\mathcal{S} = \{X_t\}_{t\in T}$ taking values in the alphabet $\mathcal{A}$. In terms of physics, one can think of an experimental set-up, including all of the measure apparatus and the computer collecting the data, as one process and the index t, can be thought of as time[1]. This project will only consider processes in discreet time[2] over a finite alphabet. While this is indeed a very simple subset of all possible processes these restrictions are very well motivated by the dominant use of digital computers in all branches of science and every other activity based on collecting data.

Notation and lingo Given a process $\mathbb{P} = ...X_{t-1}X_tX_{t+1}X_{t+2}...$ we will often have specify a subset of consecutive random variables. To do so we will use the notation

$$X_{t:t'} = X_tX_{t+1}...X_{t'-2}X_{t'-1}$$

to denote the random variables from and including X_t and up to, but not including, $X_{t'}$. When we call the index *time* provides us with some suggestive lingo. When we fix some t we will call the $X_{t-L:t}$ the 'length L past' and $X_{t:t+L}$ the 'length L future' of t. We will also talk about the entire past (future) of t meaning the entire set of random variables with index lower that (higher than) t. We will denote the entire *past* by $X_{:t} = ...X_{t-2}X_{t-1}$ and the entire *future* by $X_{t:} = X_tX_{t+1}....$

[1]This is good picture to gain intuition, but in the general theory there is nothing to stop us from letting T describe the spacial dimension of a one-dimensional crystal, or any other dimension suitable for a given problem.

[2]The natural index set of a discreet bi-infinite sequence is $\mathbb{Z}$, and time will indeed be integers throughout the project.

Word distributions When we 'perform a measurement' of one of the random variables it gives a *realization*. Realizations are symbols from the symbol alphabet, and we will generally denote them with lower case x's. If we measure a consecutive set of random variables $\{ X_t \mid t \in t_m, ..., n \}$ each of them gives a realization x_t. When we combine these symbols we get a word. $w = x_m ... x_n$. We say that the word w is the realisation of $X_{m:n}$.
In the simple case of discreet time and finite alphabet, we can define the probability of a words without worrying about measure theory (we will simply use the counting measure). The probability of a specific realization will be denoted by $Pr(X_{m:n} = w)$.

Stationary Processes A process is called stationary if it is invariant under time-translations. In terms of word-distributions this mean that

$$Pr(X_{t_0:t_0+L} = w) = Pr(X_{t_0+\Delta t:t_0+\Delta t+L} = w)$$

for any $\Delta t \in \mathbb{Z}$.[3] When a process is stationary we can therefore talk about the probability of a word without specifying what specific random variables the word is a realization of. When convenient we will simplify the notation to $Pr(w)$ meaning that given some realization of any length L sequence of random variables from the process, what is the probability that it will be exactly w.

Past, future and the present 'state' Now imagine that an observer have seen a specific realization $w = x_{-L}x_{-L+1}...x_{-1}$ of the length L past[4] $X_{-L:0}$ of a stationary process. Then we talk about our prediction of the future as the conditional probability distribution of the future given the past $Pr(X_{0:L'} = w'|X_{-L:0} = w)$. We will say that pasts are causally equivalent if they lead to the same prediction of the future. That is given two pasts w_1 and w_2 of length $L_1, L_2 \in \mathbb{N} \cup \{\infty\}$ respectively, the probability of any finite length future is the same:

$$w_1 \sim w_2 \Leftrightarrow Pr(X_{0:L} = w|X_{-L_1:0} = w_1) = Pr(X_{0:L} = w|X_{-L_2:0} = w_2), \forall L \in \mathbb{N}$$

That this is indeed an equivalence relation is obvious since it is define by an equality. This relation induces a partition of the set of all possible pasts into what we will call the causal states. The causal state corresponding to a given past can be thought of a the observers 'present state of knowledge' about the process.

3 Markov Models

Markov Chains A Markov chain is a process obeying the Markov condition:

$$Pr(X_{t+1:}|X_{:t}) = Pr(X_{t+1:}|X_t)$$

This means all the information about the future stored in the past is is available in the present. Or in other words, predictive states are the set of pasts ending on the same symbol.

Markov chains have been the subject of intense studying for about a century and the amount of results and literature is well beyond what can be done justice to in this project. For our purposes, however, it will suffice to restrict ourselves to the simplest of cases: Assuming discreet time, a finite alphabet, and stationarity. In this simple setting the dynamical system is fully

[3] since the time is discreet there is a bijection between T and $\mathbb{Z}$. So I will generally assume that the time is just integers.

[4] When processes are stationary the 'spitting point' between past an future can be chosen freely. For convenience we usually choose $t = 0$.

described by one Transition Matrix T in which the (i,j)'th entry is the probability of seeing symbol j directly after i.

$$Pr\big(X_{t+1} = x_j | X_t = x_i\big) = Pr\big(X_1 = x_j | X_0 = x_i\big) = T_{ij} = \langle e_i | \, T \, | e_j \rangle$$

Here $\langle e_i|$ (and $|e_j\rangle$) are row (column) "vectors" with zero in all entries except for a one in the i'th (j'th). The intuition we should try to get by this notation is, that the row "vector" $\langle e_i| \, T$ is the probability-distribution of the random variable X_{t+1} conditioned by knowing the realization of X_t.
From this we can calculate the probability of a length-2 word $w = x_j x_k$ following the symbol x_i by using:

$$\begin{aligned} Pr\big(X_1 = x_j \wedge X_2 = x_k | X_0 = x_i\big) &= Pr\big(X_1 = x_j | X_0 = x_i\big) Pr\big(X_2 = x_k | X_1 = x_j \wedge X_0 = x_i\big) \\ &= Pr\big(X_1 = x_j | X_0 = x_i\big) Pr\big(X_2 = x_k | X_1 = x_j\big) \\ &= T_{ij} T_{jk} = \langle e_i | \, T \, | e_j \rangle \, \langle e_j | \, T \, | e_k \rangle \end{aligned}$$

And should we be interested in the value of X_2 without caring about X_1 all we have to is to sum over all the possible intermediate x_j's:

$$\begin{aligned} Pr\big(X_2 = x_k | X_0 = x_i\big) &= \sum_{j \in \mathcal{A}} Pr\big(X_1 = x_j \wedge X_2 = x_k | X_0 = x_i\big) \\ &= \sum_{j \in \mathcal{A}} T_{ij} T_{jk} = \langle e_j | \, T^2 \, | e_k \rangle = \big(\langle e_j | \, T \big) T \, | e_k \rangle \end{aligned}$$

This result is suggestive because it shows how T can be thought of as a generator of time-translation in the following sense: Given a probability distribution over the symbols at time t, the probability distribution over the symbols at time $t+1$ is found simply by right multiplication with T. To get to time $t + \Delta t$ we just repeat this Δt times (or equivalently multiply from the right by $T^{\Delta t}$).

Markov Machines (the graph representation) The transition matrix representation is a very powerful tool to perform computations with Markov chains.However, to get a good intuition about what is going on, it will help us to introduce a more graphical way of presenting the same information.

We will think of a Markov Machine as weighted and directed graph. Each node (or state) corresponds to a symbol in the process alphabet, and the edge from node i to node j specifies the probability that x_i is followed by x_j. The machine is inhabited by a random walker. At a given time t the random walker is standing on a node (we also say that the machine 'is in one of the states'). At every time-step, the random walker looks at the edge labels (transition probabilities) of the edges leaving its current state, rolls a dice, and walks along one of the edges to a new state.

Example: Let's consider the four state Markov Machine with the alphabet $\{ A, B, C, D \}$ given by the transition matrix:

$$T = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 \\ 0.5 & 0 & 0 & 0.5 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

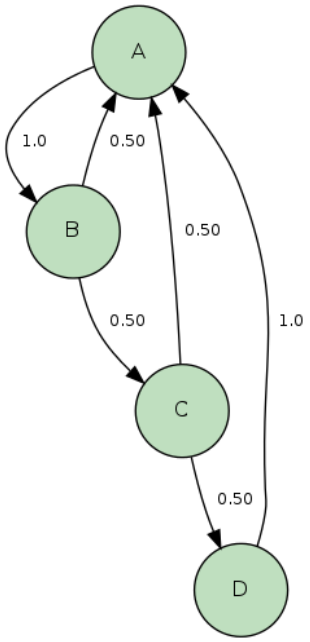


Figure 1: Example of a graph-representation of a four state Markov Machine

where state A corresponds to $\langle e_1| = (1,0,0,0)$, and so on.
The graph of this machine is shown in Figure 1.
The graph representation is nice because it gives us a much more intuitive feeling of 'flow' between the nodes than we get from looking at the transition matrix. It is for example easy to get the idea that the random walker 'spends more time' in state A than in D. If we come back from a long coffee break, we will be less surprised to find it in state A than in state D.

Invariant distribution Now let's imagine we are performing an experiment perfectly described by a known Markov Process. Once we have measured one symbol we can start making predictions about the future by multiplying away by T. But, what can we say before we have done a singe measurement, and how far into the future does it make sense to keep on calculating?
Keeping in mind the example it seems fairly intuitive that for each time-step the probability distribution becomes more 'washed out' across the states. The further we look into the future, the less it helps us that we know the present. In the limiting case of trying to predict a symbol infinitely far into the future we should expect that the answer is completely independent of the measurements we have made in the present.
Because of the stationarity of the process this is equivalent to estimating the present given that our latest measurement took place infinitely long time ago. So, if the correlation between measurements really goes to zero as the time between them goes to infinity, the question of guessing the present without knowing anything about the past is really the same as the question of prediction the far future.

It can be shown that[5] for any Markov Chain which is irreducible (that is: there is a path from any state to any other state), there exists one unique *Invariant Distribution* $\langle \pi|$ such that $\langle \pi| = \langle \pi| T$, and that if there is at least one transition with probability strictly between zero and one then

$$\langle p|\, T^t \to \langle \pi| \ \text{ for } t \to \infty$$

for any initial probability distribution $\langle p|$.

Notice that the definition of $\langle \pi|$ is just the definition of a left-eigenvector of T with eigenvalue one. Thus the invariant distribution can be found by the standard method of finding eigenvectors which essentially comes down to solving a set of linear equations.

Due to the uniqueness of the invariant distribution, $\langle \pi|$ is the answer to the question about predicting without any preceding measurements. And to evaluate the value of predictions about the future, one can always compare them to the invariant distribution to see if the prediction is substantially different than that of someone who has not made any measurements at all.

To follow up on the example above, Figure 1, a calculation of it's invariant distribution yields:

$$\langle \pi| = \left(\frac{4}{11}, \frac{4}{11}, \frac{2}{11}, \frac{1}{11}\right)$$

Markov chains can be interesting, but we want to discus a more general set of processes and to do so we need a more sophisticated model class than the Markov Machines, namely:

Edge-labelled Hidden Markov Machines [HHM's] These are essentially Markov machines with a symbol attached to each edge. The random walker behaves just like before with the additional property that every time it chooses a path it yells out the label attached to it.

[5] [1] Information Symmetries in Irreversible Processes - page 4

These machines *hidden* because we imagine them being inside a closed box. The box prevents us from directly observing which state the random walker is in, but the random walker shouts loud enough, that we can keep track of the sequence of edge symbols. If there is a distinct label for each edge, we are just as well of as we were before we closed the box, but this is not one of our assumptions. In general each symbol can be attached to any number of edges so reading off emitted symbols only gives us partial information about the internal state.

To formalize this idea, we now have a set of internal states $\mathcal{S}$, an output alphabet $\mathcal{A}$, and a set of labelled transitions with known probabilities. This can be summed up in a set of labelled transition-matrices $\left\{ T^{(x)} \;\middle|\; x \in \mathcal{A} \right\}$ with entries:

$$T^{(x)}_{ij} = Pr(x, s_j | s_i)$$

the probability of transitioning from state s_i to state s_j while emitting the symbol x.

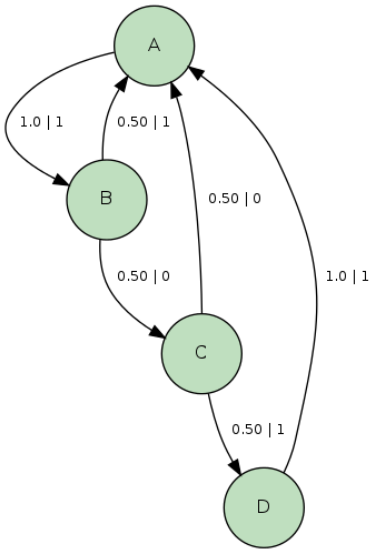


Figure 2: Example of a graph-representation of a Hidden Markov Machine. The internal machine is the same as the Markov Machine in Figure 1

Example: Consider a Hidden Markov Machine Figure 2 with the same internal state structure as the Markov Machine in Figure 1], and an output alphabet $\mathcal{A} = \{ 0, 1 \}$. The corresponding labelled transition matrices are:

$$T^{(0)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, T^{(1)} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

A simple, but very important observation is that the sum of the labelled transition-matrices adds up to the transition matrix of the internal Markov machine. Not listening to the output symbols and guessing the current state is the same as predicting the state in advance, before the symbols were emitted.

HMMs as Generators of Processes Let us first specify some notation. We keep track of time by using sub-scripts. S_t is a random variable describing the internal state at time t. It takes values from the set of states $\mathcal{S}$ who's elements we will denote with lower-case s'es. X_t is the random variable corresponding to the emitted symbol leaving state S_t. It takes values in the symbol alphabet $\mathcal{A}$ who's elements will be denoted by lower-case x'es. To visualize the time-indexation the following table might be useful:

Internal states	...	S_0	S_1	S_2	S_3	...
Observed symbols	...		X_0	X_1	X_2	...

Given a Hidden Markov Model, and assuming that we know it's internal state at time $t = 0$, that is $S_0 = s_i$, we can find the probability that the next emitted symbol will be $x \in \mathcal{A}$, by summing over all the transition-probabilities of leaving s_i while emitting x.

$$\begin{aligned} Pr(X_0 = x | S_0 = s_i) &= \sum_{s_j \in \mathcal{S}} Pr(X_0 = x \wedge S_1 = s_j | S_0 = s_i) \\ &= \sum_{s_j \in \mathcal{S}} T^{(x)}_{ij} = \langle e_i | \, T^{(x)} \, | \mathbf{1} \rangle \end{aligned}$$

Where $|\mathbf{1}\rangle$ is a column of all ones. In the case where the exact internal state at time zero is unknown, but we do have a probability-distribution, we must do a weighted sum over the different start-states:

$$Pr\big(X_0 = x | Pr(S_0 = s_i) = p_i\big) = \sum_{s_i \in \mathcal{S}} \sum_{s_j \in \mathcal{S}} p_i T^{(x)}_{ij} = \langle p| \, T^{(x)} \, |\mathbf{1}\rangle \, .$$

To update our probability distribution once we see a symbol, we use:

$$Pr(S_1 = s_j | X_0 = x \wedge S_0 = s_i) = \frac{Pr(S_1 = s_j \wedge X_0 = x | S_0 = s_i)}{Pr(X_0 = x | S_0 = s_i)}$$

If we denote the probability distribution at time t by $\langle p_t|$ we have:

$$\langle p_1| = \frac{\langle p_0| \, T^{(x)}}{\langle p_0| \, T^{(x)} \, |\mathbf{1}\rangle}$$

We can think of the denominator as a scalar re-normalization constant to insure that the probability keeps summing to one.
With these equations at hand we can start answering questions about word probabilities. Since the internal machine behaves exactly like a Markov Machine we can find it's invariant distribution and use it as our optimal guess given no previous data. The probability of a word $w = x_0 x_1 ... x_n$ can then be calculated by repeatingly finding the probability of the next symbol and updating the probability distribution given that symbol

$$\begin{aligned}
Pr(w) &= Pr(x_0 x_1 ... x_n) = Pr(x_0) Pr(x_1 ... x_n | x_0) \\
&= Pr(x_0) Pr(x_1 | x_0) Pr(x_2 | x_0 x_1) ... Pr(x_n | x_0 ... x_{n-1}) \\
&= \langle p_0| \, T^{(x_0)} \, |\mathbf{1}\rangle \, \langle p_1| \, T^{(x_1)} \, |\mathbf{1}\rangle \, ... \, \langle p_n| \, T^{(x_n)} \, |\mathbf{1}\rangle \\
&= \langle p_0| \, T^{(x_0)} \, |\mathbf{1}\rangle \, \frac{\langle p_0| \, T^{(x_0)}}{\langle p_0| \, T^{(x_0)} \, |\mathbf{1}\rangle} T^{(x_1)} \, |\mathbf{1}\rangle \, ... \frac{\langle p_0| \, T^{(x_0)} T^{(x_1)} ... T^{(x_{n-1})}}{\langle p_0| \, T^{(x_0)} T^{(x_1)} ... T^{(x_{n-1})} \, |\mathbf{1}\rangle} T^{(x_n)} \, |\mathbf{1}\rangle \\
&= \langle p_0| \, T^{(x_0)} T^{(x_1)} ... T^{(x_{n-1})} T^{(x_n)} \, |\mathbf{1}\rangle \\
&= \langle p_0| \, T^{(w)} \, |\mathbf{1}\rangle
\end{aligned}$$

where $T^{(w)} = T^{(x_0)} T^{(x_1)} ... T^{(x_{n-1})} T^{(x_n)}$.

This gives us an explicit formula for calculating word-probabilities given a Hidden Markov Model. Since a process is defined by it's word-probabilities there is a process corresponding to every HMM. We say that the Hidden Markov Machine generates (or is a generator of) its process. There is, however, not a one-to-one correspondence here. Different HMMs may generate the same process. Two answer this ambiguity we will introduce the concept of a process' ε-machine.

Recurrent states In a Markov machine (hidden or not), a state s is called recurrent if there exists a finite probability path from s to itself. If a state is not recurrent it's called transient. As mentioned earlier, a machine is called irreducible if given any two states s and s', there is a finite probability path from s to s'. It is clear that in an irreducible machine every state is recurrent.
This project is only considering irreducible machines. This is because machines being irreducible is closely related to their generated processes being stationary, in the sense that time translation can be thought of as changing the 'start state' of the generating machine.

Unifilarity A hidden Markov model is said to be unifilar if all the edges leaving a given node have different symbols attached to them. A more mathematical way to state the same thing is that given the state at time t and the following symbol the state at time $t+1$ is uniquely determined.

$$Pr(S_1 = s_j | X_0 = x_0 \wedge S_0 = s_i) = \delta_{j,k(i,x)}$$

Figure 2 is an example of a unifilar machine. It is easy to see that a HMM is unifilar if and only if each row in each labelled transition matrix have at most one non-zero entry.
If a HMM is unifilar and we know the state at at time t and the following n emitted symbols, then we also know the internal state at time $t+n$. This follows easily by induction since at each time-step we know both the state and the following symbol. Thus given the start state, calculating the probability of a give word w reduces to multiplying the probabilities of the symbol at each time step given the current state

$$Pr(x_0 x_1 x ... x_n | s_0) = Pr(x_0, s_1 | s_0) Pr(x_1, s_2 | s_1) ... Pr(x_{n-1}, s_n | s_{n-1})$$

A very useful way to think about unifilarity is that once an observer has 'synchronized' to the process (come to know the internal state specifically), she will not loose synchronization as long as she keeps track of the emitted symbols. Later we will see how unifilarity is also closely related to the definition of states by partitioning possible pasts.

4 ε-Machines

We started by discussing processes and introduced the idea of a causal state as subset of the set of all possible pasts which are equivalent for predicting the future. Then we introduced Markov Chains, a certain sub-class of processes, and saw how their predictive states were determined only by the most recent symbol of the past. This led us to introduce Markov machines as an efficient and powerful tool for describing Markov chains. Inspired by the Markov machines we proceeded by introducing Hidden Markov Models and saw how finite HMMs can generate a strictly larger class of processes than finite Markov machines. And now we will see how it all ties up.

Given a process $\mathbb{P}$, we can partition all realizations into a set of predictive states $\mathcal{S}$. An observer is keeping track of the emitted symbols, and at time $t = 0$, he has seen the word $w =$'$x_{-L} ... x_{-2} x_{-1}$', for $x_i \in \mathcal{A}$. The causal state corresponding to the past w is $s_i \in \mathcal{S}$. We say that the process is 'in state s_i. Since pasts in s_i all give the same prediction of the entire future, they do in particular give the same probability-distribution over the next symbol, governed by the random variable X_0. After one time-step the observer sees that $X_0 = x_0$, which leads to a new past $w' =$'$x_{-L} ... x_{-2} x_{-1} x_0$' which corresponds to a predictive state $s_j \in \mathcal{S}$. We can think of this as a transition from state s_i to state s_j on emitting the symbol x_0.
Say we are working with a process that has only finitely many predictive states $|\mathcal{S}| = n$ for some $n \in \mathbb{N}$. Then we can list all of these transition probabilities in a set of labelled transition matrices, and so we have found a HMM which generates the process.
Since the predictive states and their 'predictions' are uniquely determined by the process $\mathbb{P}$, this specific HMM is also uniquely determined. We call it the ε-machine of the process.

ε-Machines are Unifilar This follows from the construction of the transition dynamics, since every transition probability is specified by the causal state it is coming from and the symbol which is emitted during the transition.

Infinite Pasts and Recurrent Causal States Say an observer have recorded the last L symbols emitted by some process, and they happens to form the word w which has the corresponding causal state s. We can think of w as *the* past, but we can also choose to think of it as just the last L symbols of a longer past of say length L'. If we do so we can say: *The processes' current state is actually the state corresponding to a past of length L' whose last L symbols form the word w. The observer is just not sure which one of them.* By taking this idea to the limit $L' \to \infty$ we can think of any past as being infinitely long.
States corresponding to infinite pasts are either recurrent or have probability zero. To give a formal proof of this requires going into a measure-theoretical description of processes, in which we can talk about the probability distribution over infinite length words. That is well beyond the scope of this project so I will point to stationarity and talk to the intuition of a physicist when I say: If the past is infinitely long, and a state has finite probability of being reached only once, then it have already happened a long time ago, and we can safely assume that it won't happen again.
Effectively this means that for any ε-machine of a recurrent process, all the transient causal states can be thought of as superpositions of the recurrent states. We can say that the process is always in one of the recurrent states of the ε-machine, the observer just might not know which one.

In the literature the name ε-machine seems to be used in two meanings. Both as the full set of causal states corresponding to words of any length, and as only the recurrent part of this machine. In the rest of this project I will focus on the recurrent part of ε-machines, and since it's too cumbersome to keep writing 'the recurrent part' I will use ε-machines in the second meaning, unless the context clearly indicates differently.

5 Mixed state machine

So far we have discussed the existence of ε-machines based on the theoretical existence of a causal equivalence relation which partition infinite pasts into causal states. In this section we will develop an algorithmic approach to actually finding the ε-machine of a process, starting with some generating HMM.
We have already described a probability-distribution over internal states by some row-matrix $\langle p|$ where $\langle p \,|\, e_i \rangle$ is the probability of the machine being in state s_i. We have also seen how the labelled transition matrices can be used to 'update' the probability distribution when observing symbols. The new idea here is to consider every possible probability distribution as an individual 'mixed-state'. The dynamics over these states is induced by the dynamics over M, simply by using the labelled transition matrices to advance the probability-distributions. Given a mixed state $\langle p_0|$ at time $t = 0$ we know to calculate the probability of the following symbol by the formula

$$Pr(x|\,\langle p_0|) = \langle p_0|\, T^{(x)}\,|\mathbf{1}\rangle$$

And given the following symbol x_0 we know how to calculate the next probability distribution $\langle p_1|$, which is also a mixed state:

$$\langle p_1| = \frac{\langle p_0|\, T^{(x_0)}}{\langle p_0|\, T^{(x_0)}\,|\mathbf{1}\rangle}$$

From this description we can see the main advantage of the mixed-state representation, namely that it is unifilar. However, this has come at the cost of going from a finite number of states to an uncountable infinity (since the state-probabilities can be varied continuously). The solution to this is to realize that not all of these states have to be taken into account. While every past has a corresponding state (due to the unifilariy), there might be states which do not have

corresponding pasts. In other words, a state may be the empty subset of pasts.
One state which does definitely correspond to a past is the invariant distribution $\langle \pi |$ which corresponds to the past 'not having seen any symbols yet'. This will be the start state of our construction. From this we can calculate the state corresponding to any finite length past $w = x_{-n}...x_{-1}$ by the formula:

$$\langle p_w | = \frac{\langle \pi | T^{(w)}}{\langle \pi | T^{(w)} | \mathbf{1} \rangle}$$

If it happens that two words w and w' lead to the same mixed-state/probability-distribution ($\langle p_w | = \langle p_{w'} |$) then we know that the two pasts are equivalent for prediction. This is true since the probability of any future word w^* depends only on the present state.

$$\begin{aligned} Pr(X_{0:n} = \vec{w} | X_{-m:0} = w) &= \frac{Pr(X_{-m:n} = ww^*)}{Pr(X_{-m:0} = w)} = \frac{\langle \pi | T^{(ww^*)} | \mathbf{1} \rangle}{\langle \pi | T^{(w)} | \mathbf{1} \rangle} \\ &= \frac{\langle \pi | T^{(w)}}{\langle \pi | T^{(w)} | \mathbf{1} \rangle} T^{(w^*)} | \mathbf{1} \rangle = \langle p_w | T^{(w^*)} | \mathbf{1} \rangle \\ &= \langle p_{w'} | T^{(w^*)} | \mathbf{1} \rangle = \frac{\langle \pi | T^{(w'w^*)} | \mathbf{1} \rangle}{\langle \pi | T^{(w')} | \mathbf{1} \rangle} \\ &= Pr(X_{0:n} = w^* | X_{-m':0} = w') \end{aligned}$$

While this proves that pasts which lead to the same mixed-state are equivalent predictors, it is not necessarily true that equivalent predicting words will lead to the same mixed-state. So when we are finished with the mixed-state algorithm (which is potentially infinite) we must find equivalent predictive states and 'collapse' them to get the ε-machine.

Mixed state algorithm The algorithm assumes that we start with a process $\mathbb{P}$, over an alphabet $\mathcal{A}$, generated by some finite Hidden Markov Model M consisting of a set of internal states $\mathcal{S} = \{ s_1, ..., s_n \}$ with dynamics described by a set of labelled transition-matrices $\left\{ T^{(x)} \;\middle|\; x \in \mathcal{A} \right\}$.
The algorithm works with a set of reachable mixed states *MixedStates* and a set of transitions between the reachable mixed states *Transitions*. When we start the algorithm both sets are empty, and our goal is to fill them up.

1. Calculate the invariant distribution of the generating HMM, and add it to the (currently empty) set *MixedStates* marked as 'unchecked'.

2. Pick an unchecked state s from *MixedStates* and for all $x \in \mathcal{A}$:

   - Calculate the probability of the next symbol being x: $\Pr(x|s) = \langle s | T^{(x)} | \mathbf{1} \rangle$.
   - If $Pr(x|s) > 0$ calculate what state this transition leads to: $\langle s' | = \frac{\langle s | T^{(x)}}{\langle s | T^{(x)} | \mathbf{1} \rangle}$
   - Add s' as an element to *MixedStates* (if it's not already there) and then add a link to *Transitions* specified by: from s, to s', on symbol x, with probability $Pr(x|s)$.
   - Then mark s as checked.

3. When there are no more 'unchecked' states in *MixedStates* we are left with a unifilar generator of the process MSM. We minimize this machine by 'collapsing' any states which are causally equivalent, which will leave us with the ε-machine including it's transient states.

4. Finally, we can choose to isolate the recurrent part of the machine and throw away the transients, if we don't need these for computations.

One important thing to notice is that if we keep on adding new states to the set *MixedStates* we will never run out of 'unchecked' states to check, and the algorithm will never finish. Luckily for us, it so happens that the algorithm does actually terminate for most of the cases we are examining in this project.

Golden Mean Process The mixed state algorithm plays a central role in the rest of this project, so I will give an simple of explicitly calculating the mixed state machine. For simplicity we take a process, know as the Golden Mean Process generated by an HMM with just two states $\mathcal{S} = \{ A = (1,0), B = (0,1) \}$, and an output alphabet $\mathcal{A} = \{ 0, 1 \}$. The Machine has labelled transition matrices:

$$T^{(0)} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \text{ and } T^{(1)} = \begin{pmatrix} p & 1-p \\ 0 & 0 \end{pmatrix}, \text{ for } p \in (0,1)$$

This process generator is not unifilar, so there is no chance it's the ε-machine.

1. Invariant distribution: $\langle \pi | = (\frac{1}{2-p}, \frac{1-p}{2-p})$.

2.

$s = (\frac{1}{2-p}, \frac{1-p}{2-p})$	x	$P(x\|s)$	s'	new?
	0	$\frac{1-p}{2-p}$	$(1,0)$	yes
	1	$\frac{1}{2-p}$	$(p, 1-p)$	yes

$s = (1,0)$	x	$P(x\|s)$	s'	new?
	0	0	N/A	no
	1	1	$(p, 1-p)$	no

$s = (p, 1-p)$	x	$P(x\|s)$	s'	new?
	0	$1-p$	$(1,0)$	no
	1	p	$(p, 1-p)$	no

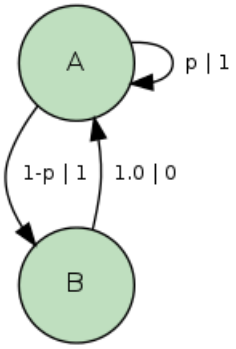


Figure 3: Non-unfilar generator of the Golden Mean Process

3. The mixed state machine is shown in Figure 4. It is easy to see that there are no causally equivalent states, since the two states predict different futures already on the first symbol of their future, so this is *the* ε-machine of the Golden Mean Process.

Notice that the node-labels used in Figure 4 are dependent on what generating HMM we used as base of the mixed-state algorithm, so they are clearly not unique to the ε-machine.

Implementation Even when the algorithm ends in finitely many steps it is still a lot of calculations to go through. Most of it though, is linear algebra, and as such very well suited of solving on a computer. By working together with people at the UC Davis Complexity Sciences Center I have had access to a Python package CMPy which is specifically designed to perform calculations with hidden Markov machines and in particular ε-machines. The package is purely numerical though, so in order to do more careful analysis, I have written a script which performs the mixed-state algorithm symbolically using SymPy.

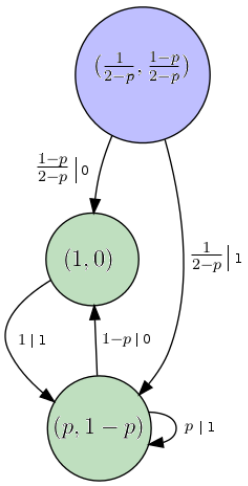


Figure 4: ε-machine of the Golden Mean process including transient states

6 Topological ε-Machines

The word support of a process $\mathbb{P}$ is the set of words which have probability strictly greater than zero. Words in the word support are called allowed words, and words which are not in the word support are said to be forbidden. By looking at the ε-machine of the golden mean process, Figure 4, we can easily convince us selves that all words containing consecutive 0's are forbidden.
The word support of a process says a lot about its structure. And it says a lot about how an observer comes to 'synchronize' to the process (how observed sequences can lead to exact knowledge of the internal state). To be able to study these properties we will give a 'generator description' of word-supports similar to how we have discussed word probabilities so far.

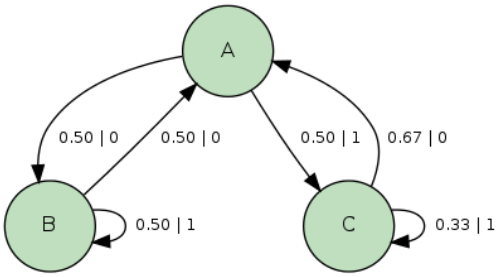


Figure 5: Example of an ε-machine which is *not* a topological ε-machine. The states B and C predicts different probabilities of seeing a 0 as the next symbol, but they agree on the possible future words.

Topological ε-machines The causal states of a process are induced by a equivalence relation, $\sim$, over pasts, identifying two pasts if they lead to the exact same prediction of the future. Similarly we can construct 'topological states'[6] induced by an equivalence relation, $\approx$, which identifies pasts which lead to the same allowed future words. It is clear that $\approx$ is a relaxation of $\sim$. That is for any two pasts w and w' it holds that $w \sim w' \Rightarrow w \approx w'$. From this it follows that any topological state is a union of one or more causal states. Just like we constructed ε-machines by describing how attaching 'the next output' symbol to the end of a past defined a transition from one causal state to the next, we can construct a 'topological ε-machine' by listing out how symbols induce transitions between topological states. Only this time we cannot assign a probability to the transitions, since different pasts corresponding to the same topological state might disagree on this. In stead of labelled transition matrices we will construct labelled adjacency matrices $A^{(x)}$. The entry $A^{(x)}_{ij}$ is 1 if the topological state $\tilde{s}_i$ transitions into $\tilde{s}_j$ on seeing the symbol x, and it is 0 otherwise. Note that if all the causal states of an ε-machine are also 'topological causal states', we say that it is an topological ε-machine even though it has probability-labels on the edges. Figure 5 is an example of an ε-machine which is *not* a topologica ε-machine.

Topological Mixed State Algorithm With slight modifications, the mixed state algorithm can be used to find the topological ε-machine of a process. The difference is that we only need to keep track of whether a state have *some* probability or not. In stead of probability-distributions the 'topological mixed states' can be thought of as a row-matrix with entries in $\{ 0, 1 \}$. This simplifies the calculations a lot, and make the 'topological mixed state algorithm' run way faster than the probabilistic mixed state algorithm. Further, if a process is generated by a HMM with $n \in \mathbb{N}$ states, then there can be no more than $2^n < \infty$ 'topological mixed states'. This mean that the topological mixed state algorithm is guaranteed to terminate.
While the topological ε-machine of a process carries much less information than the full ε-machine, in some cases it might be enough to answer the questions we have. In others cases calculating the topological ε-machine can be a helpful step on the way to finding the full ε-machine.

[6]The use of the word 'topological' is supposed to remind us that we have thrown away a lot of structure.

7 Reverse Machines

The processes which are the subject of this project are all bi-infinite, well ordered sequence of random variables $\{ X_t \}$. When the process is described like this, it doesn't have a particular direction. Sure the index runs from negative to positive and we have a strong tendency to put the negative to the left, or associate it with the past. This, however, is probably more of a cultural inheritance than anything else. The process itself is a stationary mathematical object outside of time and space.
On the other hand, when we describe a process by a generating Markov machine (hidden or not), there is clearly a direction involved. In the generating models the outcome of each random variable dependeds only on the 'preceding' random variables, the ones with a lower index, and affects only the future ones. There is, however, no reason why we shouldn't be able to call the variables with higher index 'the past', and the variables with lower index 'the future', or equivalently we could flip the sign of every index.
With this in mind a natural question arises: If a machine M generates a process $\mathbb{P}$ from negative to positive, what machine generates the process in the opposite direction?

Reverse processes When we restrict ourselves to stationary processes in discrete time, over a finite alphabet, a process is fully described by the set of word-probabilities of all words of arbitrary length. In this description it is natural to define the 'reverse' process as a process in which the the word-probabilities of 'inverted' words matches the probabilities of the original(/forward) process. So if there is a probability p of seeing the word '00101' is in the forward process, then there should be probability p of the word '10100' in the reverse process.
To formalize this let $\vec{\mathbb{P}}$ and $\overleftarrow{\mathbb{P}}$ be two processes both over the alphabet $\mathcal{A}$. Let $\vec{w}^n = x_1x_2...x_n$ where $x_i \in \mathcal{A}$ be some word of length n with the 'inverted word' $\overleftarrow{w}^n = x_nx_{n-1}...x_1$. The the processes are each others inverses if:

$$Pr(\vec{w}^n|\vec{\mathbb{P}}) = Pr(\overleftarrow{w}^n|\overleftarrow{\mathbb{P}}), \quad \text{for all } n \in \mathbb{N} \text{ and all } \vec{w}^n \in \mathcal{A}^n$$

Since it is more convenient to stick to the habit of time running from low to high, we will recast the question in terms of reversing the process. If a machine M generates the process $\vec{\mathbb{P}}$, what machines generates $\overleftarrow{\mathbb{P}}$?
In particular, since ε-machines are in a one-to-one correspondence with their processes we want to find a way of finding reverse ε-machines.

Reversing Markov Machines The basic idea behind finding reverse-machines is to think about the machine's graph-representation. At a given time t the random-walker is on a given node, let's say s_t, and the edges/arrows leaving this node tell where the random-walker is about to go. At the same time the arrows pointing into node s_i tell us about where the random-walker came from. Since the process is generated by the path of the random walker, the reverse-process is intuitively described by playing the movie backwards. That is

$$Pr(X_1 = x_j|X_0 = x_i)Pr(X_0 = x_i) = Pr(X_0 = x_i, X_1 = x_j) = Pr(X_0 = x_i|X_1 = x_j)Pr(X_1 = x_j)$$

We divide through by $Pr(X_1 = x_j)$ and get:

$$\overleftarrow{T}_{ji} = Pr(X_0 = x_i|X_1 = x_j) = Pr(X_1 = x_j|X_0 = x_i)\frac{Pr(X_0 = x_i)}{Pr(X_1 = x_j)} = \vec{T}_{ij}\frac{\langle \pi \,|\, e_i \rangle}{\langle \pi \,|\, e_j \rangle}$$

If we define the diagonal matrix D_π by $(D_\pi)_{i,i} = \langle \pi \,|\, e_i \rangle$ we can write this even more compactly as:

$$\overleftarrow{T} = D_\pi^{-1}\vec{T}^T D_\pi$$

Reversing HMMs Since different hidden Markov models can generate the same process, there is a bit of ambiguity in finding 'the reverse machine'. However, there is a straight forwards way of finding *a* machine which will generate the reverse process. Reversing the internal Markov machine can be thought of as flipping all the arrows (edges), and renormalizing their transition-probabilities. If we do this without changing the labels attached to each edge will exactly correspond to 'playing the movie backwards'. Since the internal state path determines the emitted word, having the same probability of each inverted internal state path will also lead to having the same probability of the inverted emitted words.
Formally we can consider two processes $\vec{\mathbb{P}}$ and $\overleftarrow{\mathbb{P}}$ generated by a hidden Markov models who's labelled transition matrices satisfy $\overleftarrow{T}^{(x)} = D_\pi^{-1}(\vec{T}^{(x)})^T D_\pi$. By using the know formula for word-probabilities, taking the transpose and simplifying we find:

$$\begin{aligned} Pr(x_1x_2...x_n|\vec{\mathbb{P}}) &= \langle\pi|\,\vec{T}^{(x_1)}...\vec{T}^{(x_n)}\,|\mathbf{1}\rangle = \langle\mathbf{1}|\,(\vec{T}^{(x_n)})^T...(\vec{T}^{(x_1)})^T\,|\pi\rangle \\ &= \langle\mathbf{1}|\,D_\pi\overleftarrow{T}^{(x_n)}D_\pi^{-1}...D_\pi\overleftarrow{T}^{(x_1)}D_\pi^{-1}\,|\pi\rangle \\ &= \langle\pi|\,\overleftarrow{T}^{(x_n)}...\overleftarrow{T}^{(x_1)}\,|\mathbf{1}\rangle = Pr(x_n...x_1|\overleftarrow{\mathbb{P}}) \end{aligned}$$

which is exactly the statement that $\overleftarrow{\mathbb{P}}$ is the reverse-process of $\vec{\mathbb{P}}$.

Reversing ε-machines ε-machines are uniquely determined by their process and so we can unambiguously define the reverse ε-machine $\overleftarrow{\varepsilon\,\mathrm{M}}$ of an ε-machine $\overrightarrow{\varepsilon\,M}$ as the ε-machine of the reverse process.
While defining reverse machines in this way is intuitive and straight forward, working with processes as abstract mathematical objects is almost impossible, which is why we have introduced generating machines, and in particular ε-macines in the first place. It is therefore necessary to give an algorithmic description of how to find reverse ε-machines, for them to have any practical meaning at all.

The idea is as follows:

1. Give the ε-machine $\overrightarrow{\varepsilon\,\mathrm{M}}$ of the process $\overrightarrow{\mathbb{P}}$ in form of a unifilar hidden markov model, we can use the method described above to find an HMM $\overleftarrow{\mathrm{rM}}$ which generates the reverse process $\mathbb{P}$.
2. Since $\overleftarrow{rM}$ is a generator of $\overleftarrow{\mathbb{P}}$ we can use it as the base of the 'mixed state algoritm' to find the ε-machine $\overleftarrow{\varepsilon\,\mathrm{M}}$ of $\overleftarrow{\mathbb{P}}$.

While the first step is always very well behaved in the sense that it conserves the number of nodes, edges, edge-labels and so on, this is not generally true for the second step. As we have already hinted earlier the mixed state algorithm does not give any promise of terminating. Later we will show by example that an ε-machine with finitely many states can indeed have an infinitely large reverse machine. First, however we will start by giving a well-behaved example, to get a better understanding of the algorithm described above.

Example: Reversing the Golden Mean Process The ε-machine of the Golden Mean Process was found as an example of how to use the mixed-state algorithm. We start with recurrent part of the Golden Mean process. Which have the labelled transition matrices:

$$\overrightarrow{T^{(0)}} = \begin{pmatrix} 0 & 1-p \\ 0 & 0 \end{pmatrix} \text{ and } \overrightarrow{T^{(1)}} = \begin{pmatrix} p & 0 \\ 1 & 0 \end{pmatrix}$$

From this we find the invariant distribution $\langle\pi| = \left(\frac{1}{2-p}, \frac{1-p}{2-p}\right)$, which we use to construct the diagonal matrix D.
Then we find the labelled transition matrices $rT^{(0)}$ and $rT^{(1)}$ of a generator of the reverse process:

$$rT^{(0)} = D^{-1}\overrightarrow{T^{(0)}}^T D = \begin{pmatrix} 2-p & 0 \\ 0 & \frac{2-p}{1-p} \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1-p & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2-p} & 0 \\ 0 & \frac{1-p}{2-p} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

$$\text{and } rT^{(1)} = D^{-1}\overrightarrow{T^{(1)}}^T D = \begin{pmatrix} 2-p & 0 \\ 0 & \frac{2-p}{1-p} \end{pmatrix} \begin{pmatrix} p & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2-p} & 0 \\ 0 & \frac{1-p}{2-p} \end{pmatrix} = \begin{pmatrix} p & 1-p \\ 0 & 0 \end{pmatrix}$$

Now these are, as expected, the transition-matrices of a non-unifilar Hidden markov machine. In fact they are the same one as the generator we used to define the Golden Mean Process in the first place. To find the ε-machine of the reversed process we must therefore use the mixed-state algorithm. We already know the result, namely the ε-machine of the Golden Mean Process.
The reverse ε-machine of the Golden Mean Process is identical to the forward ε-machine for all choices of the transition probability parameter p. Since ε-machine are unique to their processes this implies that the Golden Mean process is identical to its reverse process.

Irreversible processes A process is said to be reversible if it is identical to its reverse-process. That is $\overrightarrow{\mathbb{P}} = \overleftarrow{\mathbb{P}}$, or equivalently for all words $\overrightarrow{w}$ with the reversed word $\overleftarrow{w}$ it holds that

$$Pr(\overrightarrow{w}|\,\overrightarrow{\mathbb{P}}) = Pr(\overleftarrow{w}|\,\overrightarrow{\mathbb{P}})$$

If a process is not reversible it is irreversible. From the definition above we see that a process is irreversible if there exists just one word which has a different probability than it's reverse word. Since ε-machines are uniquely determined by their process, reverse ε-machine of irreversible process must also be different.
The Golden Mean Process is an example of a reversible process. In fact all ε-machines[7] with one or two states, and binary output alphabet, generate reversible processes (see Table 1 on page 20). For ε-machines with three states, more than half generate reversible processes as well. Since it is convenient to give examples of ε-machines with few states, this can give the misleading idea that irreversibility is some kind of rare artefact. However, as the last rows of Table 1 shows, the share of reversible machines seem two grow with the number of states, and already at six states far most processes are irreversible.

To give an idea of how different irreversibility can be, the next pages will go through thorough analyses of three different kinds of irreversibility.

[7]The set of all ε-machines up to eight states, with a binary output alphabet, have been enumerated. This is mentioned in Ref. [1]

8 Support Driven Irreversibility

A process have irreversible word support if and only if its topological ε-machine is different from the topological ε-machine of the reverse process. From performing an exhaustive survey of all topological ε-machines with up to 6 states, Table 1, we find that this is in fact far the most common case. Here is an example to illustrate what that can look like. Consider the process generated by the ε-machine in Figure 6, with some $p, q \in (0, 1)$. By inspection we find that all states are predictively distinct for any choice of p and q so we don't have to worry about that.
We find the reverse machine by applying the standard procedure. First we find the invariant distribution:

$$\langle \pi | = \left(\frac{1}{p(2-q)+2}, \frac{1}{p(2-q)+2}, \frac{p}{p(2-q)+2}, \frac{p(1-q)}{p(2-q)+2} \right)$$

Then we 'reverse the arrows'. Find a generator of the reverse process:

$$rT^{(x)}_{ij} = T^{(x)}_{ji} \frac{\pi_j}{\pi_i}$$

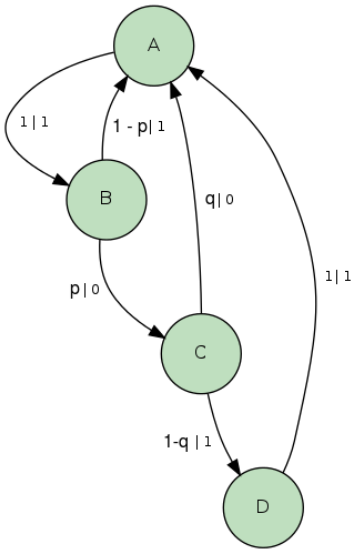


Figure 6: ε-machine of a process with irreversible word support

We now use this generator of the reversed time process as base for the mixed state algorithm, and when the process terminates we isolate the recurrent part of the Mixed State Machine. This leaves us with a unifilar generator with the states:

E	F	G	H	I	J
$(0,1,0,0)$	$(1,0,0,0)$	$(0,0,1,0)$	$\left(0, \frac{1-p}{1-pq}, 0, \frac{p(1-q)}{1-pq}\right)$	$\left(0, \frac{1-q}{1-pq}, \frac{q(1-p)}{1-pq}, 0\right)$	$\left(\frac{1-p}{1-pq}, 0, \frac{p(1-q)}{1-pq}, 0\right)$

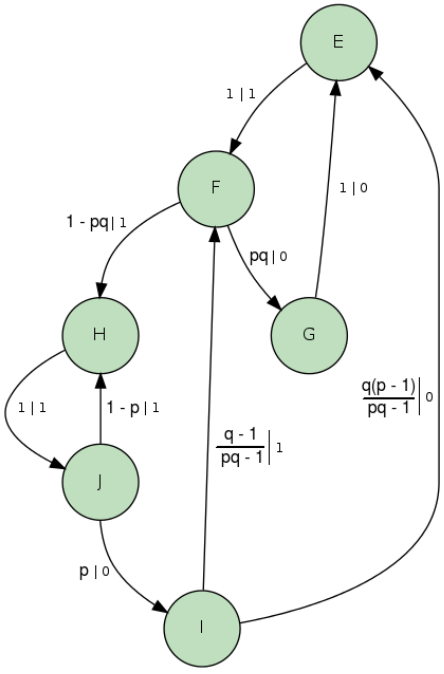
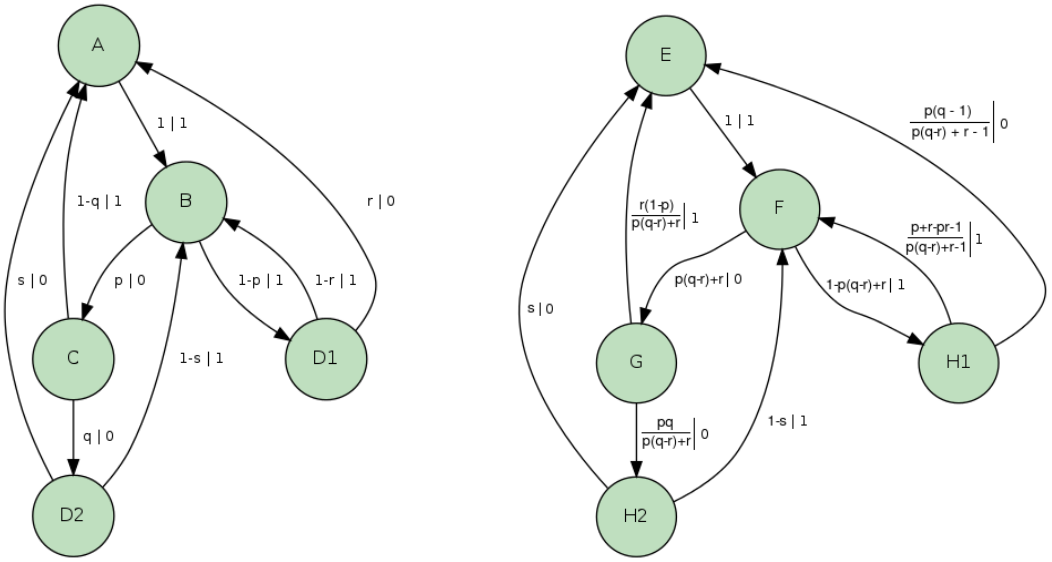


Figure 7: And the ε-machine of the reverse time process.

These states, together with the set of transitions between them (also found during the execution of the mixed state algorithm) form a unifilar hidden Markov model Figure 7.
By inspection we find that no two state are equivalent for prediction, so this is indeed the ε-machine of the reverse process. In fact, we can check that the words which can possibly follow each state of the reversed machine are different, so if we ignore the transition probabilities it is a topological ε-machine (same thing apply to the forward machine). The forward machine and the reverse-machines are clearly different, so the word-supports of the forward and reverse processes must also be different.
As a final remark on this example we notice that the forward machine has three states from which the transitions are uncertain F, I, and J. Thus it would require three parameters to describe the space of processes whose ε-machines have this structure. The forward-machines can be specified with only two parameters. So if we pick a machine with the topology of the reverse-machine but randomize all the transition probabilities, and reverse it, the resulting machine will in general not have the structure of the forward-machine. It will, however, have the same word-support, so we know that it can be found by 'splitting' one or more of the state.

9 Probability Drive Irreversibility

The next example illustrates a process which has a reversible word support, but irreversible word probabilities. It also illustrates that an ε-machine topology can be reversible for some transition-probabilities while irreversible for others, and it allow us to get some intuition about how this works. We consider the process with the forward and reverse ε-machines in Figure 8



(a) ε-machine of forward process (b) ε-machine of reverse process

Figure 8: The reversibility of these ε-machines depend on the parameters p, q and r.

There are several things to be noticed about these two machine. *First*, and most eye-catching, the structure (or topology) of the two machines are identical, while the transition-probabilities are different. From this we can conclude that the word-supports are identical, while the processes are not, so the irreversibility is purely probabilistic.
Secondly it is worth noting that these machine are not topological ε-machines, since there are states (D1/D2 in the forward machine and H1/H2 in the reverse-machine), which are redundant for some choices of parameter values. For the forward [reverse] process this happens at $r = s$ $\left[\frac{p(q-1}{p(q-r)+r-1} = s\right]$, since the futures of states D[/H] 1 and 2 will be going to the same states one the same transitions with the same probabilities. When the states are redundant we must 'identify' them to get the ε-machine of the process, so the ε-machine will have only 4 states when these conditions are satisfied. Since the conditions are different in the forward and reverse machines, this shows that the ε-machine of one direction can change structurally (by changing the number of states), while the structure of the reversed machine stays fixed.
While the transition probabilities in the forward and in the reverse-machines are not generally identical, their are values of (p, q, r, s) such that they are. This means that the ε-machine is reversible for some transition-probabilities and irreversible for others. To find the transition-probabilities which make the process reversible, we simply solve the set of equations:

$$p = p(q-r) + r \text{ and } q = \frac{pq}{p(q-r)+r} \text{ and } r = \frac{p(q-1)}{p(q-r)+r-1} \text{ and } s = s$$

In this case it turns out that except for $s = s$ the other equations are actually equivalent. So if one of the other transition probability is identical to the corresponding transition probability in the reverse machine, then so are all of them.

To get a more visual grasp of what we have just described, we can consider the space of processes which have ε-machines of the shape in Figure 8a as a 4D-hyper-cube parametrized by the numbers (p, q, r, s) all in the open interval $(0, 1)$. The reversible processes, the processes which have a 4-state ε-machine, and the processes whose reverse processes have 4-state ε-machines will then form three subsets which can be though of as hyper planes. In order to plot it out we can fix one of the parameters and make a plot like Figure 9. I have chosen to fix s since it is fixed in the mapping between the forward and the reverse machines, and therefore it doesn't change the shape of the 'reversible-manifold' (yellow).

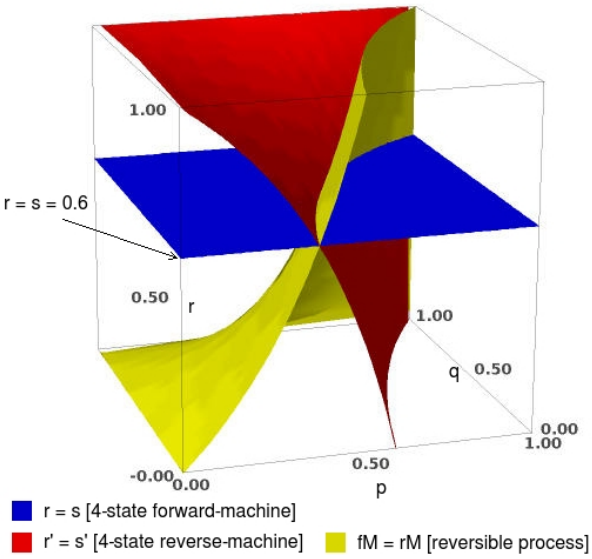


Figure 9: The space of machines with topology as in Figure 8a, parametrized by the transition probabilities p, q and r. s kept fixed to keep it down in 3D

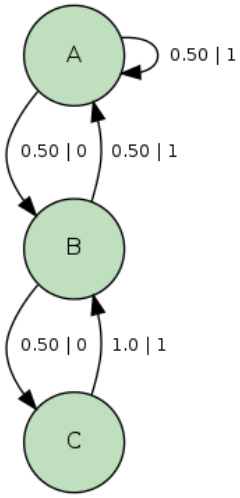
Can any of this be generalized? The calculations in this section pretty much tells us everything there is to know about the reversibility of this class of processes. However, this model class was not chosen for being related to any physical system or any other real world problem, so the results are particularly interesting by them selves. This machine was chosen because it illustrates couple of properties about reverse-machines in general.
The number of states in the reverse-machine can change without any structural changes happening in the forward machine. However, these seemingly discontinuous change are not completely out of control. If the word-support of the forward process i fixed, then so is the word-support of the reversed process, thus the states of the reversed ε-machine must be subsets of the same support-causal-states. So the number of states can only change by two (or more) states becoming causally equivalent for certain values of the transition probabilities. A necessary condition for such a 'collapse' is that the transition-probabilities out of the 'equivalent states' are equal. Such an equality reduces the degree of freedom, so it will always correspond to a hyper-surface in the parameter hyper-cube representing the class of processes (in the sense that it will always correspond to a subset of lower dimension). The 'volume' of a hyper surface is zero, implying that if the transition probabilities are chosen randomly from a uniform distribution over the interval $(0, 1)$ then the probability of a point on the surface is zero.
By the same reasoning, the same thing applies to the reversibility of ε-machines, namely that if a certain ε-machine is not reversible for all choices of transition-probabilities, then it is only reversible on a null-set.

Calculating reverse-machines can be done much faster using numerical transition probabilities than symbolic ones. If we want to describe the irreversibility of a class of processes with a fixed ε-machine structure, we might be fooled by calculating the irreversibility of only one of them (e.g. in the example described in this section the ε-machine is reversible for the most typical choice of transition probability: 50/50 chance on all non-deterministic transitions). The observations described above tells us that if we do the calculations with randomly chosen transition probabilities, the results will be as 'complex' as possible. In the sense that if the reverse machine have a different number of states for different parameters, we will find the largest one. And if there are parameters for which the reverse machine is reversible, we will know.

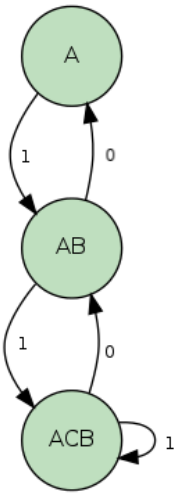
10 Explosive Irreversibility

When using the mixed state algorithm there is no guaranty that it is going to terminate. We might keep on adding new states ad infinitum and there is no easy way to determine whether it's going to end or not. This can also happen when we try to reverse find reverse ε-machines. A process which have only finitely many casal states when generated in one direction can have infinitely many causal states when generated in the opposite direction. Here is the simplest example of this. Consider the process generated by the ε-machine in Figure 10a. We won't get anywhere by applying the standard procedure for finding reverse-machines, so we have to proceed a little more carefully. The first step is to find the topological mixed state machine of it's reverse process Figure 10b:

1. We start by not knowing anything about what state the process is in. (corresponding to state ABC)
2. From ABC, seeing a '1' we could still be in any state, seeing a '0' we can exclude state C (corresponding to state AB)
3. From AB, seeing a '0' will exclude B and 'synchronize' us to state A. Seeing a '1' on the other hand will send us back to ABC
4. From A, we will certainly see a 1 and return to state AB



(a) The explosive 3-state machine



(b) And it's topological reverse machine

Figure 10: Graphs of the only 3-state ε-machine which have an infinite reverse machine.

The first thing we notice is that the topologically reverse machine has the same structure (same topology) as the forward machine. This means that the process has a reversible word-support.
The next thing is that it is possible to synchronize to the singe state A. This is useful, because it means we already know one of the recurrent states of the actual reverse ε-machine. If we use the mixed state algorithm, but uses a recurrent state as start state, instead of the invariant distributions, then we will only find recurrent states as well. This means we don't have to worry about whether the states we find are recurrent or transient.

11 Reverse Machine Survey

That we are at all able to describe all topological ε-machines is due to a complete enumeration build in the the software package CMPy, developed by the people at the UC Davis Complexity Science Center. Finding topological reverse-machines is done by a script which I have written for the purpose. The script extracts the labelled adjacency matrices of an ε-machine generated with the CMPy. It then uses the adjacency to find the topological ε-machine of the reverse process by following the procedure described in the section on the mixed state algorithm. When the reversed ε-machine is found, the CMPy package is used again, to test if the forward and the reverse machine are identical.
The interesting facts to see in this table are: 1) The percentage of processes with irreversible word support seem to grow rapidly with the number of causal states, and already at six states far most machines have irreversible word support. This suggests that irreversibility is more of a

States	Top. ε-machines	Reversible word support	Reversible process	Shared state
1	3	3	3	3
2	7	7	7	7
3	78	$60(\sim 77\%)$	49 $(54)^*$	70
4	1388	$364(\sim 26\%)$	256 $(311)^*$	1322
5	$35,186$	$2,604(\sim 7.5\%)$	$(2,079)^*$	$33,970$
6	$1,132,613$	$17,108(\sim 1.5\%)$	$(12,990)^*$	$1,111,897$

Table 1: Survey of reversibility of all topological ε-machines up to 6 states. Numbers in $()^*$ are machines with uniform transition probabilities taken from reference [1] for comparison. 'Shared state' are number of machines with at least one state being causal in both time directions.

rule than an exception in one-dimensional complex processes. 2) As expected more ε-machines are found to be irreversible when they are given random transition probabilities than when the transition probabilities are chosen uniformly. 3) While the share of reversible ε-machines seem to go down as the number of states go up, the share of machines which have a state shared between the forward and reverse process seem to go up (think of the state A in the explosive irreversibility example). This suggests using the topological mixed state algorithm as a preliminary step before before applying the probabilistic mixed state algorithm might very often be a computational advantage, if we are only interested in the recurrent part of the reverse ε-machine.

12 Conclusions

Through out this project we have been studying one dimensional *processes*. These are generally very abstract beasts, so we have restricted the field of study to discreet time-, finite alphabet-, reversible processes. To describe these we have introduced the concept of ε-machines, via the idea of causal states, and shown that there is a unique ε-machine for each process. We went on by defining *Hidden Markov Machines*, a class of process generators, and saw that if a process has only finitely many recurrent causal states the ε-machine is a *unifilar* Hidden Markov Machine.
In order to actually find ε-machines we introduced the *Mixed State Algorithm*, a systematic approach to determine the ε-machine of a process given any generating (finite state) Hidden Markov Model.
We then turned the attention to *reverse processes* - processes found by inverting the 'time'-index. We defined a process to be reversible if it is identical to its reverse process, and since ε-machines are unique to their processes this is equivalent to the ε-machine of the forward process being equal to the ε-machine of the reverse process. To put this to use we described an algorithmic approach to finding reverse ε-machines by first finding a generating Hidden Markov Machine of the reverse process and then apply the mixed state algorithm.
By closely examining a small handful of examples we found a wide variety of 'reversibilities' of ε-machines. Saw **1)** ε-machine topologies being reversible independent of transition-probabilities. **2)** Reverse ε-machines with different topologies than their forward ε-machine (corresponding to an irreversible word-support). **3)** Topologies for which the corresponding process is reversible for some transition probabilities but the forward and reverse ε-machines have a different number of states for others. **4)** Finite state ε-machines whose reverse machines have infinitely many states.

Finally I claimed, based on a generalization of a closely analysed example, but without giving

a formal proof, that given a ε-machine topology the number of states in the reverse machine can depend on the choice of transition parameters, but it will be maximal for *almost every* choice. And similarly given a topology if there exist a set of transition probabilities such that the ε-machine is irreversible, then *almost all* of the ε-machines will be so.

13 Acknowledgements

Being on exchange at UC Davis while writing this project have had a great impact on the final product. Even the choice of subject would hardly have been the same under any other circumstances. Being here I have had the opportunity of working with, and leaning from, the people at the *UCD - Complexity Science Center*. And I would like to thank everyone in this department for listening and sharing their knowledge. In particular I have been under close supervision by *Professor James Crutchfield* during the whole process, and I have had the privilege of long and enlightening meetings with PostDoc's *John R. Mahoney* and *Christopher Strelioff*. Finally I will thanks my friend *Alexandre Payot* for pointing out a huge number of typos an grammatical flaws.

References

[1] Christopher J. Ellison, John R. Mahoney, Ryan G. James,1, James P. Crutchfield, and Jörg Reichardt - *Information Symmetries in Irreversible Processes* - Complexity Sciences Center and Physics Department University of California at Davis Davis, CA 95616

[2] Christopher J. Ellison, John R. Mahoney, James P. Crutchfield - *Prediction, Retrodiction, and the Amount of Information Stored in the Present* - Santa Fe Institute 1399 Hyde Park Road Santa Fe NM 87501 USA - Journal of Statistical Physics

[3] While working on this thesis I've followed the class: *Natural Computation and Self Organization* offered by the Complexity Science department at UC Davis, and taught by James P. Crutchfield. The lectures given in this class are all pre-recorded video-lectures, at http://csc.ucdavis.edu/ chaos/courses/ncaso/Lectures/.
The Lectures cover, amongst other topics, Processes, ε-machines, and describes the Mixed State Algorithm.