Memory and Thermodynamic Costs of Sampling and Biased Sampling

By

Cina Aghamohammadi

Dissertation

Submitted in partial satisfaction of the requirements for the degree of

Doctor of Philosophy

in

Physics

in the

Office of Graduate Studies

of the

University of California

Davis

Approved:

James P. Crutchfield, Chair

John B. Rundle

Richard Scalettar

Committee in Charge

2018

In the Name of Allah, the Beneficent, the Merciful.

Praise be to Allah, the Cherisher and Sustainer of the worlds;

To Him.

Contents

List of Figures

List of Tables

Abstract

Memory and Thermodynamic Costs of Sampling and Biased Sampling

With probabilistic behaviors, emergent patterns, nonlinearities, and multiple components, complex systems are challenging. A common method to study them is simulation. Sampling behaviors is central in this and so it has become a common simulation method, one used in many different fields. The goal is to mimic the system of interest in a way that both the real system and the simulation exactly show the same stochastic behavior. One then estimates properties of interest from the set of realizations.

The focus of the thesis are the resources required for sampling. It demonstrates that when a system becomes increasingly complex, the memory resources that sampling algorithms need diverge rapidly. This presents a barrier to studying complex systems.

The past several decades gave rise to the hope that a quantum computer will solve problems more efficiently than a classical computer. Efficiency here refers to the run time that algorithm takes or to the memory it uses. Chapter 2 introduces a quantum algorithm for the sampling problem, demonstrating that it is often much more memory efficient than its classical counterpart. Moreover, there are complex systems for which it is hard for the best known classical algorithm to sample, but with quantum algorithms it can be done with finite small resources. Since being introduced, the quantum algorithm was built experimentally by an independent group of scientists using a quantum photonics circuit.

Chapter 3 introduces a new phenomenon called "ambiguity of simplicity"–that classical and quantum theories can markedly disagree on structural complexity. Recent rapid progress in quantum computation and quantum simulation suggest that the ambiguity of simplicity will strongly impact statistical inference and, in particular, model selection. The idea is as follows, consider the memory-optimal classical algorithm and memory-optimal quantum algorithm for sampling. We say system A is simpler than system B, if the memory requirement for the algorithm is less than the memory requirement for system B. It turns out that system A can be simpler than system B when we consider the classical algorithm and more complex than system B when we consider quantum algorithm. As a result,

there is no total order for physical simplicity. This phenomenon is fundamental since we consider memory-optimal algorithm for both classical and quantum domain. Recently, experimentalists observed this phenomenon, too, by testing it on sampling algorithms for a simple nearest-neighbor Ising model.

For many complex systems we are interested in their rare behavior, not typical behaviors. For example, financial market crashes, tsunamis, industrial accidents, and earthquakes are thankfully rare, but extremely important. Simulating rare events, though, is particularly challenging. In these scenarios, one needs a new class of algorithms for biased sampling. Chapter 4 introduces a new class of biased-sampling algorithm based on memoryful finite-state machines. It also studies their memory cost. It turns out that sampling some classes of rare events can be extremely memory consuming while sampling others is practically free. This new tool allows both sampling rare events of interest and also determining the exact memory requirements beforehand.

Recalling the memory-efficient quantum algorithm for sampling leads one to ask if quantum computers help reduce the memory cost of biased sampling problem, too. Chapter 5 introduces a new quantum algorithm for biased sampling. The algorithm can be extremely memory efficient for the sampling of many classes of rare events.

Naturally, memory is not the only cost of computation. There are also energy costs that manifests itself as heat. Every time we do a computation, a device uses an energy resource and dumps heat into its environment. Is this cost fundamental? The answer is positive: There is a limit on how much the cost can be reduced. This chapter takes the first steps toward calculating the thermodynamic cost of sampling. It limits the analysis to the case where the process is memoryless–independent, identically distributed. This is also known as random number generation (RNG). It turns out its thermodynamic cost depends strongly on the available resources. Chapter 6 investigates the thermodynamic cost of several random number generation algorithms, including true random number generation, pseudo-random number generation, and the RNG problem. The latter is analyzed in detail via the von Neumann, Knuth and Yao, and Roche and Hoshi RNG algorithms.

Acknowledgments

Praise be to Allah who has displayed such effects of His authority and the glory of His sublimity through the wonders of His might that they dazzle the pupils of the eyes and prevent the minds from appreciating the reality of His attributes.

I am thankful to my Ph.D. advisor Jim Crutchfield who supported me through this journey, pointing me in many productive directions. We attacked many interesting problems together and sometimes we were lucky enough to solve them. I learned many things from him and I am grateful for that.

Thousand thanks to my family for their incredible support and friendship, my kind, and smart father, Amir Aghamohammadi, who taught me how to think, my wonderful, and gracious mother, Mina Donyagard, who taught me how to live, and my brilliant and determined brother, Alireza Aghamohammadi, who has been always there for me.

I want to thank Papa Saki, Amirali Nojoomi, Susan Aghamohammadi, Iman Soltani, Rouzbeh Kananizadeh, Ali Shayegan, Farzad Pourbabaee, and Erfan Nourbakhsh for their support.

I learned many things from John Mahoney, Paul Riechers, Samuel Loomis, Ryan James, Adam Rupe, Greg Wimsatt, Ariadna Venegas-Li, and Xincheng Lei. They played an important role in my research and I am thankful for that.

Chapter 1

Sampling and Its costs

1.1 Overview

One common way to study complex systems is to simulate them. Sampling is that part of simulation in which the goal is to generate a realization of a system's behavior. That sample will be studied later in the process of simulation. Sampling has its own costs. What are these costs and why do they appear?

This thesis attempts, first, to understand these costs and, second, to offer new approaches to reduce them. This chapter briefly reviews different types of cost in sampling, discusses why each of cost exists, and offers alternatives to common sampling methods. It lays the foundation for the rest of the thesis.

After reviewing basic concepts, assumptions, and notation, I formally introduce sampling and briefly study different classes of algorithm. The focus is on finite-state machines, especially Hidden Markov models–an important class of sampling algorithm. The quantum sampling algorithm is then introduced as an alternative that can reduce memory costs.

When the goal is to study the rare behavior of complex systems simple sampling algorithms fail. One solution is a biased sampling algorithm. Next, I introduce classical and quantum biased sampling algorithms and their memory costs.

Thermodynamic dissipation is another important cost that appears in simulation. Here I discuss briefly why this cost exists for the simulation process. Specifically, I address the thermodynamic cost of several cases of sampling algorithm when the user has access to

different type of resources.

1.2 Stochastic Processes

In the beginning, we need to define some basics. A stationary ergodic discrete-time stochastic process [3, 4] is a probability distribution $\mathbb{P}(.)$ over the bi-infinite chain of random variables $\cdots X_{-1}X_0X_1\cdots$. As a result for a known process, one has access to $\mathbb{P}(X_iX_{i+1}\cdots X_{i+d})$ for any i and d or one can assign a probability to any events. The word "Stationary" here means there are no particular points in time that are different. All the points in time are equal or $\mathbb{P}(X_iX_{i+1}\cdots X_{i+d}) = \mathbb{P}(X_jX_{j+1}\cdots X_{j+d})$ for any i, j, and d. The event space for all random variables X_is or as we will refer to it in the future, the alphabet, are the same and denoted by $\mathcal{A}$. Here we only focus on the case where $\mathcal{A}$ is a finite set.

The simplest class of processes are I.I.D processes where they do not have any memory of the past. This means for any i and d, $\mathbb{P}(X_iX_{i+1}\cdots X_{i+d}) = \prod_{k=0}^{d}\mathbb{P}(X_{i+k})$. The next simple but commonly used class is the Markovian processes where they only remember the last generated event. This means for any i and d, $\mathbb{P}(X_iX_{i+1}\cdots X_{i+d}) = \mathbb{P}(X_i)\prod_{k=1}^{d}\mathbb{P}(X_{i+k}|X_{i+k-1})$. The definition can be extended to the Markov order-R class where the processes only remember the last generated R events. $\mathbb{P}(X_iX_{i+1}\cdots X_{i+d}) = \mathbb{P}(X_iX_{i+1}\cdots X_{i+R-1})\prod_{k=1}^{d}\mathbb{P}(X_{i+k+R-1}|X_{i+k-1}X_{i+k}\cdots X_{i+k+R-2})$.

1.3 Simulator and Finite-state machine

Simulation of a given process $\{\mathbb{P}, \mathcal{A}\}$ here refers to generating a sample with the same distribution as the process. A simulator is an algorithm that simulates the process. Consider an algorithm that can generate random samples with any size n, $X_0X_1\cdots X_{n-1}$ with probability $\mathbb{Q}(X_0X_1\cdots X_{n-1})$. Then the algorithm is the simulator of the given process if $\mathbb{P} = \mathbb{Q}$. Sometimes the term simulation also refers to the case where the two measures are approximately equal $\mathbb{P} \simeq \mathbb{Q}$, or they are equal at the large n limit, b ut we do not consider these cases a simulation.

There are two main algorithms for the sampling of stochastic systems, Monte Carlo algorithms, and finite-state machine algorithms. In this work, we only consider the

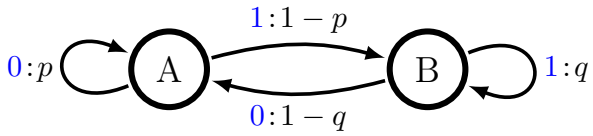


Figure 1.1: Example Hidden Markov Model. If the model is in state A with probability p it will generate 0 and stay at state A and with probability $1-p$ it will generate 1 and make a transition to the state B. If the model is in state B with probability q it will generate 1 and stay at the same state and with probability $1-q$ it will generate 0 and make a transition to the state A.

finite-state machine algorithms. Each of them has its own advantages. Depending on the situation one may choose one or another.

The general idea behind Monte Carlo algorithm is as follows. Let us say the user wants a sample with size n generally $n \gg 1$. The algorithm starts with a particular realization of size n, then depending on the algorithm the realization may be deterministic–all the random variables take the same value–or completely random?drawn from an IID uniform distribution. Then by doing many stochastic local changes based on some designed law the realizations eventually changes to the sample of interest. During the process, the algorithm must memorize the whole realization and as a result, the memory requirement scales with the sample size n.

There are many different types of finite-state machine algorithms but the core idea behind is the same. For simplicity, let us consider Hidden Markov Models (HMMs), an important class of finite-state machine algorithm [4–6]. An HMM is a directed network consisting of nodes or "states" and edges or "transitions" labeled with two numbers There is a random walker on the network. Each time the random walker make a transition from one state to the other, a symbol on the edge he took would be generated. The other number on the edge is the probability that the random walker choose that particular edge. Figure 1.1 shows an example of an HMM.

As a result, at each step, the finite-state machine algorithm generates a symbol, unlike the Monte Carlo algorithm that generates the whole sample at the same time. This is the important difference between the two which makes each of them advantageous in different scenarios.

If the user of the sample and the one that runs the algorithm are the same then often the Monte Carlo algorithm will be used. If the computer that runs the algorithm does not memorize the sample that generates and instead writes it, symbol by symbol, on an external memory register for the user then the finite-state machine would be a better choice. This situation happens when the user of the sample –equivalently the external memory– and the one that runs the algorithm –equivalently the computer– are different.

Before introducing the finite-state machine algorithm let us look at several simple sampling algorithms to understand why there is a need for more sophisticated algorithms.

Let us say for a given process $\{\mathbb{P}, \mathcal{A}\}$, which for now we assume it can be any process with any Markov order, the user wants to generate a sample with size n. The most trivial algorithm memorize the whole probability distribution for any size n. Every time the user wants a sample, the algorithm refers to the probability distribution. It generates a random number r between 0 and 1 and by comparing it with probability distribution $\mathbb{P}(X_0X_1\cdots X_{n-1})$ generate a sample with a correct distribution. To do this the algorithm needs to memorize the distribution $\mathbb{P}(X_0X_1\cdots X_{n-1})$ and since the distribution has $\mathcal{A}^n$ elements the algorithm needs to memorize $\mathcal{A}^n$ numbers. Since the algorithm should have the ability to generate a sample with any size n then this algorithm needs to memorize the full distribution which is impossible due to the infinite amount of memory requirement. This is due to the fact that we did not put any condition on the target process. As a result, one needs to look for a more elegant algorithm assuming we have some partial information about the target process.

Now instead of a general process with arbitrary Markov order let us assume the given process has Markov order, R. For this process to generate a sample with size n, the information about the full distribution is not necessary and having the marginal conditional distribution $\mathbb{P}(X_R|X_0X_1\cdots X_{R-1})$ is sufficient. Since the marginal distribution has $\mathcal{A}^{R+1}$ elements the algorithm only needs to memorize $\mathcal{A}^{R+1}$ numbers not $\mathcal{A}^n$ numbers. This is a great progress compared to the trivial algorithm since the number of things that the algorithm needs to remember is independent of sample size n. One can see that we still have exponential cost in the Markov order R.

As pointed out earlier, remembering $\mathcal{A}^{R+1}$ numbers is sufficient but the important question is, is it necessary to remember all of them? Fortunately, in many cases the answer is No. The idea behind this answer can be explained by an example. Let us say for the two sequences $x_0x_1\cdots x_{R-1}$ and $x'_0x'_1\cdots x'_{R-1}$ the conditional distributions are equal $\mathbb{P}(X_R|x_0x_1\cdots x_{R-1}) = \mathbb{P}(X_R|x'_0x'_1\cdots x'_{R-1})$. As a result, there is no need to memorize both of these two distributions. In this way, one can save memory by memorizing only $\mathcal{A}$ numbers instead of $2 * \mathcal{A}$ numbers. For a given process if this condition happens often for pairs of sequences then one can save a lot of memory.

Using this idea, constructing a more efficient algorithm is simple. The procedure is as follows: if the probability distributions of the next symbol condition on two past sequences are equal then put those two sequences on the same subset. Doing this we partition the set of sequences with length R into subsets. The next step is to label each subset. Let us denote the set of these partitions by $\mathcal{S}$. Now since the algorithm is not interested in the sample itself, the only thing it needs to remember is the label of the subset and not more than that. Let us call these labels, "states". In this way, it can correctly generate the sample of interest without remembering too much. Each time the algorithm generates a new symbol, the state changes, since the last R symbols of the sample is different from before. Using the conditional probability distribution $\mathbb{P}(X_R|X_0X_1\cdots X_{R-1})$, one can find the transition matrix between states $\mathbb{P}(S_{t+1}|S_t, X_t)$. As a result instead of memorizing the whole distribution the algorithm only memorize the conditional distribution $\mathbb{P}(S_{t+1}|S_t, X_t)$ which is much more compact.

Now let us define the Hidden Markov Model more formally. An HMM is a tuple $\{\mathcal{S}, \mathcal{A}, \{T^{(x)} : x \in \mathcal{A}\}\}$ where $\mathcal{S}$ is the set of states of the machine, $\mathcal{A}$ is the alphabet of the process, and $\{T^{(x)} : x \in \mathcal{A}\}\}$ is a set of substochastic matrices. This is how it works. Let us say at time t the state of the HMM is $S_t = i$ then at the time $t + 1$ the algorithm with probability $T^{(x)}_{ij}$ change the state of the HMM to $S_{t+1} = j$ and generate the symbol x. Continuing this procedure the algorithm samples the target process.

1.4 Quantum Simulator

A simulator is an algorithm that simulates a process. A simulator is a quantum simulator if the algorithm is a quantum algorithm. Loosely, a quantum algorithm is a procedure that runs on a quantum computer. Quantum algorithms are interesting since sometimes they achieve a speedup compare to the classical algorithms, or other efficiency improvements such as memory efficiency [7].

Quantum simulators or quantum samplers have been studied extensively for the past six years and it has been shown that they may be extremely efficient compared to their classical counterparts [2,8–16]. In the past two years, simple implementations have been also been tested experimentally [17,18]. In chapter 2 I show there are physical systems whose quantum sampler is highly memory efficient while the memory requirement for their classical sampler diverges rapidly with the system size.

There are many equivalent definitions of the quantum simulator. One that is more similar to the definition of finite-state machine algorithm is a tuple $\{\mathcal{H}, \mathcal{A}, \{K^{(x)} : x \in \mathcal{A}\}\}$ where $\mathcal{H}$ denotes the Hilbert space in which quantum states reside, $\mathcal{A}$ is the same alphabet as the given process', and $\{K^{(x)} : x \in \mathcal{A}\}$ is a set of Kraus operators [19] we use to specify the measurement protocol.

This is how it works. Assume we have the *quantum state* ρ in hand. Generally, this state can be mixed, which means it is part of a bigger pure system. At each step, the algorithm performs a measurement by applying Kraus operators $\{K^{(x)} : x \in \mathcal{A}\}$ to the state. The measurement has two results, first, it gives an outcome and second change the state of the system. Since the measurement is not projective measurement every time we apply the operator the state of the system potentially would change. Denoting the outcome of measurement by a random variable X, the probability of measuring particular value x can be written as $\mathbb{P}(X = x|\rho) = tr\left(K^{(x)}\rho K^{(x)^\dagger}\right)$. After measurement with the outcome $X = x$, the new quantum state would be $\rho' = \frac{K^{(x)}\rho K^{(x)^\dagger}}{tr(K^{(x)}\rho K^{(x)^\dagger})}$. Repeating the measurement process, again and again, generates a stochastic process. The detail of the procedure will be introduced in Chapter 5.

Since the state is mixed and the operation is general CPTP map, to implement the

algorithm in the lab one should translate the procedure into a unitary operation on a pure state. This has been recently done [20].

Now an important question is, given a stochastic process of interest, how can we design a quantum sampler which samples that specific process? Or, equivalently, given a classical sampler, how can we design a quantum sampler? Chapter 2 and 5 will go through the details of the algorithm construction.

1.5 Simultaneous Simulation and Its Memory cost

Usually, the user does not want to sample the process just for one single time. In the simultaneous sampling problem, the goal is to generate M samples of a process simultaneously, each of which is statistically independent of the others. The net result is M running algorithms. Each algorithm must memorize the current state of its HMM. If each algorithm uses its own memory, each needs $\log_2 |\mathcal{S}|$ bits of memory as before. The total memory is then $M \log_2 |\mathcal{S}|$ bits.

However, we can reduce the amount of memory required by using one large shared memory among the algorithm. In this way, according to Shannon's coding theorem [21], we can encode the states to reduce the amount of memory down to $MH(\mathcal{S}) \leq M \log_2 |\mathcal{S}|$ bits, where $H(.)$ is the Shannon entropy. The memory used per sample is then just $H(\mathcal{S})$ which can be potentially much less than $\log_2 |\mathcal{S}|$.

For a given process there are many sampling algorithms. But one may be interested in the memory-optimal algorithm, the one that between all those algorithms uses the minimum memory. This algorithm for a generic process is still unknown. The most memory efficient classical algorithm known today is called the ϵ-machine [22].

Now assume for a given process we know the ϵ-machine, the optimal algorithm which is denoted by a $\{\mathcal{S}, \mathcal{A}, \{T^{(x)} : x \in \mathcal{A}\}\}$. What is the memory cost for this sampler and how should one calculate it? As was pointed out earlier, the memory cost is the Shannon entropy of the stationary distribution over the state. One can easily see that since the matrix $T = \sum_{x \in \mathcal{A}} T^{(x)}$ is stochastic matrix the solution to the equation $\pi T = \pi$ gives us the stationary distribution. As a result, calculating the memory cost for the classical

sampling algorithm is straightforward.

Similar to the classical case for a given process there are many quantum sampling algorithms. The most memory efficient quantum algorithm known today is called the $q-$machine and was introduced two years ago [12]. What is the memory cost for this sampler? It turns out there is a similarity between the classical and quantum cases. The memory cost for the quantum sampler is the von Neumann entropy of the stationary state of the algorithm. Chapter 2,3, and 5 will go through the details of why is that.

How should one calculate the stationary state of the algorithm? The answer is not as straightforward as the classical case. The details of the procedure were introduced in Ref. [13], while a similar result recently appeared in Ref. [20].

1.6 Biased Sampling and Its Memory Cost

Another interesting and widely encountered problem related to sampling is biased sampling problem [23, 24]. For a given process, sometimes we are interested in the rare behavior of the system. For example in the financial market, extreme events are sometimes more important than typical behavior, or in earth's layer dynamics, earthquakes are really important. To study rare events, one typical approach is to sample them first and then study those samples. Consequently, one needs algorithms that sample rare events.

The first step is to find a mathematical framework that characterizes the rare events to different subsets of interest. As we will see later, this step is necessary both to have a self-consistent theory and also to apply the theory to real situations. The second task is to come up with an algorithm that samples the particular subset of interest. In chapter 4 and 5, we go through these two steps in detail by introducing a class of algorithms for biased sampling for generic processes with any Markov order.

As we will see in chapter 4, memory consumption for biased sampling algorithms depends on which class of rare events we want to sample. Depending on the process' structure, some rare-event classes can be inexpensive to sample, while some are expensive. In chapter 4 we introduce a new tool to calculate the memory costs for the sampling of different subsets of rare events.

As discussed earlier, quantum algorithms can be more efficient than classical algorithms. In chapter 5, I will introduce a new quantum algorithm for biased sampling. The algorithm can be quite memory efficient compared to the best known classical algorithm.

1.7 Thermodynamic Cost of Sampling

When using your computer to do a computational task, it's temperature rises. In doing this, the computer converts ordered (low-entropy) energy to heat. Looking at this as a thermodynamic system, the battery acts as a work reservoir, and the air around the computer acts as a thermal reservoir at the ambient room temperature. Especially in large-scale computations, such as simulations, this thermodynamic cost is not negligible. This is why the cooling system is an important component in modern computing. Obviously, no current systems are perfectly efficient. And, as a result, there is the opportunity to reduce such costs. Is there fundamental lower bound on these costs or can one can zero-energy computing with some proper implementation?

It turns out that information processing?specifically computing with access to a finite amount of memory–has an irreducible thermodynamic cost [25]. For a simple case, consider erasing a bit. To model one bit of memory consider a particle in a double well potential with energy barrier ΔE which is in contact with a thermal reservoir at temperature T. The setup is shown in Figure 1.2. Since the particle is in contact with the thermal reservoir the thermal noise can move the particle in the potential. The noise is random and its energy is on the order of magnitude of $k_{\mathrm{B}}T$. To make sure that the bit is reliable the energy barrier ΔE should be much larger than this amount. Now consider the case that the particle is on the right well and one wants to erase it or equivalently move it to the left well. To do this one needs to raise the particle beyond the energy barrier and as a result, spend at least $\Delta E > k_{\mathrm{B}}T$. This well-known result is often called Landauer's Principle [26]. This simple example qualitatively shows one of the reasons behind the thermodynamic cost of computation.

An important case of information processing and computation is random number

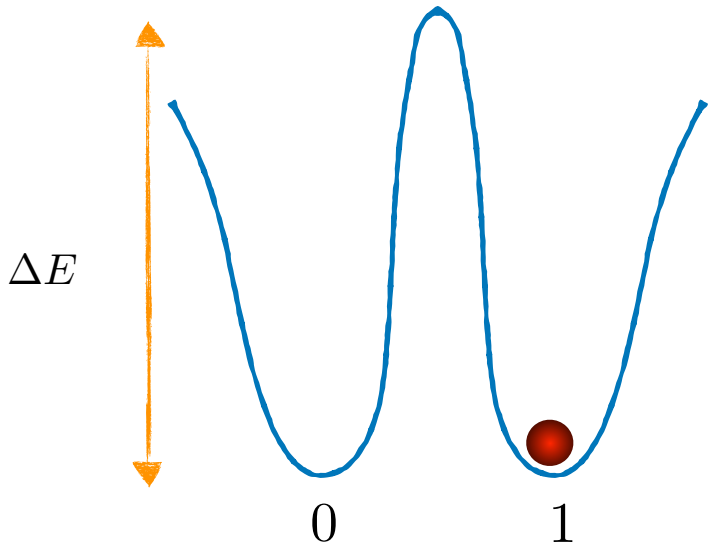


Figure 1.2: Schematic figure showing a particle in double well potential modeling one bit of memory. Thermal noise can move the particle around. A reliable bit requires ΔE being much larger than $k_B T$.

generation which is a specific case of sampling problem when the process is IID. The thermodynamic cost of sampling problem can be quite different depending on the available resources. For example, do we have access to a specific source of random number generator? An example is generating random numbers with a uniform distribution for free. Or do we want to use a deterministic algorithm and instead of a truly random number, generate pseudo-random number? Another important case is when we want to use some physical system that has some inherent randomness to design a true random number generator. In Chapter 6 I study the thermodynamic cost of random number generation in detail.

Chapter 2

Extreme Quantum Advantage when Simulating Classical Systems with Long-Range Interaction

2.1 Overview

Classical stochastic processes can be generated by quantum simulators instead of the more standard classical ones, such as hidden Markov models. One reason for using quantum simulators has recently come to the fore: they generally require less memory than their classical counterparts. Here, we examine this quantum advantage for strongly coupled spin systems—in particular, the Dyson one-dimensional Ising spin chain with variable interaction length. We find that the advantage scales with both interaction range and temperature, growing without bound as interaction range increases. In particular, it is impossible to simulate Dyson's original spin chain with the most memory-efficient known classical algorithm since it requires infinite memory, while quantum simulators can do so since they use only finite memory. Thus, quantum systems can very efficiently simulate strongly coupled one-dimensional classical systems.

2.2 Introduction

The idea of a quantum computer, often attributed to Feynman [27], recognizes that while simulating quantum many-body systems is difficult, it is apparently something that the

physical quantum system to be simulated itself accomplishes with ease. For this reason, it was conjectured that a "quantum computer"—one that operates on a quantum instead of classical substrate—might have a significant advantage in such a simulation. As modern computational technology approaches its quantum limits, the potential for a quantum advantage is becoming increasingly appealing. This has motivated diverse implementations of quantum hardware from trapped ions [28,29], cold atoms in optical lattices [30,31], superconducting circuits [32,33], photons [34,35] to liquid and solid-state NMR [36,37] and quantum dots [38].

The phrase "quantum simulation" often refers to (as originally conceived) the simulation of a *quantum* system [39]. However, this is not the only avenue in which we find quantum advantages. For instance, there is a variety of *classical* systems that can be simulated quantally with advantage [40] including simulating thermal states [41], fluid flows [42,43], electromagnetic fields [44], diffusion processes [45,46], Burger's equation [47], and molecular dynamics [48].

Quantum advantages can also be found outside of the realm of simulation. Some mathematical problems can be solved more efficiently using a quantum computer. The most well-known of these include Shor's factorization algorithm [49], Grover's quantum search algorithm [50], quantum eigen-decomposition algorithm [51], and quantum linear systems algorithm [52]. For factorization, Shor's algorithm scales polynomially [49] while the best classical algorithm currently known scales exponentially [53]. While neither algorithm has been proven optimal, many believe that the separation in scaling is real [54].

Quantum advantages also exist in the context of stochastic process generation. Sequential generation and simultaneous generation are two important problems in this field [55]. In 1988, Crutchfield and Young [56] introduced memory efficient classical algorithms for both of these problems. While there are a small number of known cases (processes) for which this algorithm can be surpassed [57–59], there remains no better general classical algorithm. Our focus here is the problem of *simultaneous generation*, the potential quantum advantage therein, and the separation in classical-quantum scaling. [Quantum algorithms for sequential generation have been studied recently [60–62].]

Reference [8] provided a quantum algorithm that can generally perform simultaneous generation using less memory than the best known classical algorithms. Recently, we introduced a new quantum algorithm—the *q-machine*—that improved this efficiency. The latter demonstrated constructively how attention to higher-order correlations in the stochastic process can lead to an improved quantum algorithm for generation [12]. A sequel provided more detailed analysis and derived the quantum advantage of the q-machine in closed form [13]. This quantum advantage has also been verified experimentally for a simple case [17]. Just as for integer factorization, proof of optimality of a simultaneous-generation algorithm is challenging in both classical and quantum settings. However, with minor restrictions, one can show that the current quantum algorithm is almost always more efficient than the classical [12].

While the existing results demonstrate a quantum advantage for generic processes, a significant question remains: *What is the scaling behavior of this advantage?* That is, to truly understand the nature of the advantage, it is critical to know how it depends on problem size. The strong separation in scaling between the classical and quantum integer factorization algorithms led many to expect that the separation will persist even as new algorithms are developed. We wish to demonstrate an analogous separation in scaling, thus solidifying the importance of the current quantum construction—the q-machine.

We choose as our testing ground the generation of equilibrium states for the one-dimensional Ising system with N-nearest neighbor interaction. Here, the coupling range N is our problem-size parameter. We choose a spin-spin coupling that decays as a power law in N. This is a natural choice, both since this model has been studied in detail over four decades [63–66] and since it provides a physically grounded benchmark.

To understand the use of such a system in this problem context, consider a one-dimensional chain of spins (with arbitrary classical Hamiltonian) in contact with a thermal reservoir. After thermalizing, the resulting bi-infinite chain of spins (considered together at some instant $t = t_0$) can be regarded as a (spatial) stochastic process. Successful generation of this stochastic process is then equivalent to generating its equilibrium states.

We quantitatively define the quantum advantage as the ratio of necessary memories for

classical and quantum algorithms. Our main result is that the quantum advantage scales as $NT^2/\log T$. We also show that classically simulating Dyson's original model requires infinite memory. In other words, exact classical simulation of the Dyson spin chain is impossible.

2.3 Dyson-Ising Spin Chain

We begin with a general one-dimensional ferromagnetic Ising spin chain [67,68] defined by the Hamiltonian:

$$\mathcal{H} = -\sum_{\langle i,j \rangle} J(i,j) s_i s_j \ , \tag{2.1}$$

in contact with thermal bath at temperature T, where s_i, the spin at site i, takes on values $\{+1, -1\}$, and $J(i,j) \geq 0$ is the spin coupling constant between sites i and j. [Throughout, T denotes the effective temperature $k_B T$.] Assuming translational symmetry, we may replace $J(i,j)$ by $J(k)$ with $k \equiv |i-j|$. Commonly, $J(k)$ is a positive and monotone-decreasing function. An interaction is said to be *long-range* if $J(k)$ decays more slowly than exponential. In the following, we consider couplings that decay by a power law:

$$J(k) = \frac{J_0}{k^\delta} \ , \tag{2.2}$$

where $\delta > 0$. The spin chain resulting from these assumptions is called the *Dyson* model [63].

To approximate such an infinite-range system we consider one with finite-range interactions. For every interaction range N, we define the approximating Hamiltonian:

$$\mathcal{H}_N = -\sum_i \sum_{k=1}^{N} \frac{J_0}{k^\delta} s_i s_{i+k} \ . \tag{2.3}$$

[N is the interaction range and should not be mistaken with the size of the lattice which is infinite here.] This class of Hamiltonians can certainly be studied in its own right, not simply as an approximation. But why is the Dyson model interesting? The ferromagnetic Ising linear spin chain with finite-range interaction cannot undergo a phase transition at

any positive temperature [69]. In contrast, the Dyson model has a standard second-order phase transition for a range of δ. Dyson analytically proved [63] that a phase transition exists for $1 < \delta < 2$. The existence of a transition at $\delta = 2$ was established much later [65]. It is also known that there exists no phase transition for $\delta > 3$ [64], where it behaves as a short-range system. Finally, it was demonstrated numerically that the parameter regime $2 < \delta \leq 3$ contains a phase transition [66], however, this fact has resisted analytical proof. For $\delta \leq 1$, the model is considered nonphysical since the energy becomes non-extensive.

Notably, the driven quantum Dyson model has been studied experimentally of late, since it exhibits many interesting nonequilibrium phases, such as the recently introduced *discrete time crystal* (DTC) [70]. The experimental system consists of a lattice of hundreds of spin-half particles stored in a Penning trap. Particles have been chosen to be $^{9}Be^{+}$ [71], $^{40}Ca^{+}$ [72] or $^{171}Yb^{+}$ [73, 74] ions. Using a combination of static electric and magnetic fields, the Penning trap confines ions. A general spin-spin coupling is implemented with an optical dipole force (ODF) induced by a pair of off-resonance laser beams. The ODF then produces Dyson-type interactions, where δ is tunable over $0 \leq \delta \leq 3$. Physically, $\delta = 1, 2, 3$ corresponds to Coulomb-like, monopole-dipole, and dipole-dipole couplings, respectively.

For these reasons this family of Hamiltonians, derived from the Dyson spin system, offer a controlled way to investigate the consequences of nontrivial correlations.

2.4 Simulators

The concept of a stochastic process is very general. Any physical system that exhibits stochastic dynamics in time or space may be thought of as *generating* a stochastic process. Here, we consider not time evolution, but rather the spatial "dynamic". For example, consider a one-dimensional spin chain with arbitrary classical Hamiltonian in contact with thermal reservoir. After thermalizing, a spin configuration at one instant of time may be thought of as having been generated left-to-right (or equivalently right-to-left). The probability distribution over these spatial-translation invariant configurations defines a stationary stochastic process.

We focus, in particular, on *stationary, discrete-time, discrete-valued stationary stochastic processes*. Informally, such a process can be seen as a joint probability distribution $\mathbb{P}(.)$ over the bi-infinite chain of random variables $\ldots X_{-1}X_0X_1\ldots$. Formally, the process denoted by $\mathcal{P} = \{\mathcal{A}, \Sigma, \mathbb{P}(.)\}$ is a probability space [3, 4]. Each spin random variable X_i, $i \in \mathbb{Z}$, takes values in the set $\mathcal{A}$. For specificity, the observed symbols come from an alphabet $\mathcal{A} = \{\downarrow, \uparrow\}$ of local spin states, but our results easily extend to any finite alphabet. $\mathbb{P}(\cdot)$ is the probability measure over the bi-infinite chain of random variables $X_{-\infty:\infty} = \ldots X_{-2}X_{-1}X_0X_1X_2\ldots$ and Σ is the σ-algebra generated by the cylinder sets in $\mathcal{A}^\infty$. Stationarity means that $\mathbb{P}(\cdot)$ is invariant under index translation. That is, $\mathbb{P}(X_iX_{i+1}\cdots X_{i+m}) = \mathbb{P}(X_{i+n}X_{i+1+n}\cdots X_{i+m+n})$, for all $m \in \mathbb{Z}^+$ and $n \in \mathbb{Z}$. For more information on stochastic processes generated by spin system we refer to Refs. [1, 75].

2.4.1 Classical Simulator

Consider a device that generates a stochastic process. We call this device a *simulator* of the process if and only if there is no way to distinguish the process outputs from those of the simulator. Given a physical system that yields a stochastic process, a device that generates this process is then said to simulate the physical system. In some contexts, the word "simulation" implies an approximation. In contrast, we require our simulators to be exact.

How do these simulators work? Generally, we implement the algorithms by writing computer programs. Two common formalisms used as algorithms for generating stochastic processes are *Markov Chains* (MC) [76, 77] and *Hidden Markov Models* (HMM) [4–6]. The latter can be significantly more compact in their representations (more efficient algorithms) and, for this reason, are sometimes the preferred implementation choice.

HMMs represent the generating mechanism for a given process by a tuple $\{\boldsymbol{\mathcal{S}}, \mathcal{A}, \{T^{(x)} : x \in \mathcal{A}\}\}$, where $\boldsymbol{\mathcal{S}}$ is a finite set of states, $\mathcal{A}$ is a finite alphabet (set of symbols), and $\{T^{(x)} : x \in \mathcal{A}\}$ is a set of $|\boldsymbol{\mathcal{S}}| \times |\boldsymbol{\mathcal{S}}|$ substochastic symbol-labeled transition matrices. The latter's sum $\mathbf{T} = \sum_{x\in\mathcal{A}} T^{(x)}$ is a stochastic matrix.

As an example, consider the Even Process [78, 79]. The process can be explained by a simple procedure. Consider a biased coin that with probability p generates heads and

with $1-p$ generates tails. To generate the Even Process we use the algorithm:

$$\begin{cases} \textit{Step A: Flip the coin.} \\ \textit{Step B: If the result is heads, output } 0 \textit{ and go to Step A. Else output } 1 \textit{ and go to Step C.} \\ \textit{Step C: Output } 1 \textit{ and go to Step A.} \end{cases}$$

This algorithm is depicted by the HMM shown on the left of Fig. 2.1a. For this HMM, $\boldsymbol{\mathcal{S}} = \{A, B\}$, $\mathcal{A} = \{0, 1\}$, $T^{(0)} = \begin{pmatrix} p & 0 \\ 0 & 0 \end{pmatrix}$, and $T^{(1)} = \begin{pmatrix} 0 & 1-p \\ 1 & 0 \end{pmatrix}$. The HMM, as an algorithm, simply tells the computer that: if we are in state A then, with probability p, output 0 and stay at state A and, with probability $1-p$, output 1 and go to state B. If we are in state B, output 1 and go to state A.

The goal of sequential generation is to produce a very long realization of the process. For this, we use one computer with a code that runs the algorithm. At each step, the computer must memorize the current HMM state. Since the HMM has 2 states, we require 1 bit of memory for this process, independent of its bias p.

Here, though, we are interested in simultaneous generation where the goal is to generate M realizations of a process simultaneously, each of which is statistically independent of the others. The net result is M computers each with the above code, as on the right side of Fig. 2.1a. Similar to the sequential problem, each computer must memorize the current state of its HMM. If each computer uses its own memory, each needs 1 bit of memory as before. The total memory is then M bits.

However, we can reduce the amount of memory required by using one large shared memory among the computers. Figure 2.1a emphasizes this schematically. In this way, according to Shannon's coding theorem [21], we can encode the HMM states to reduce the amount of memory down to $M\,\mathrm{H}(\mathcal{S}) \leq M$ bits, where $\mathrm{H}(\mathcal{S}) = -\Pr(A)\log_2\Pr(A) - \Pr(B)\log_2\Pr(B)$. The memory per instance is then just $\mathrm{H}(\mathcal{S})$.

Every process has an infinite number of alternative HMMs that generate it. For example, Fig. 2.1b shows three HMMs that each generate the Even Process, each with different $\mathrm{H}(\mathcal{S})$ and as a result different memory costs. Now, an important question when considering all possible generators is, which HMM needs the minimum memory or, equivalently, minimum

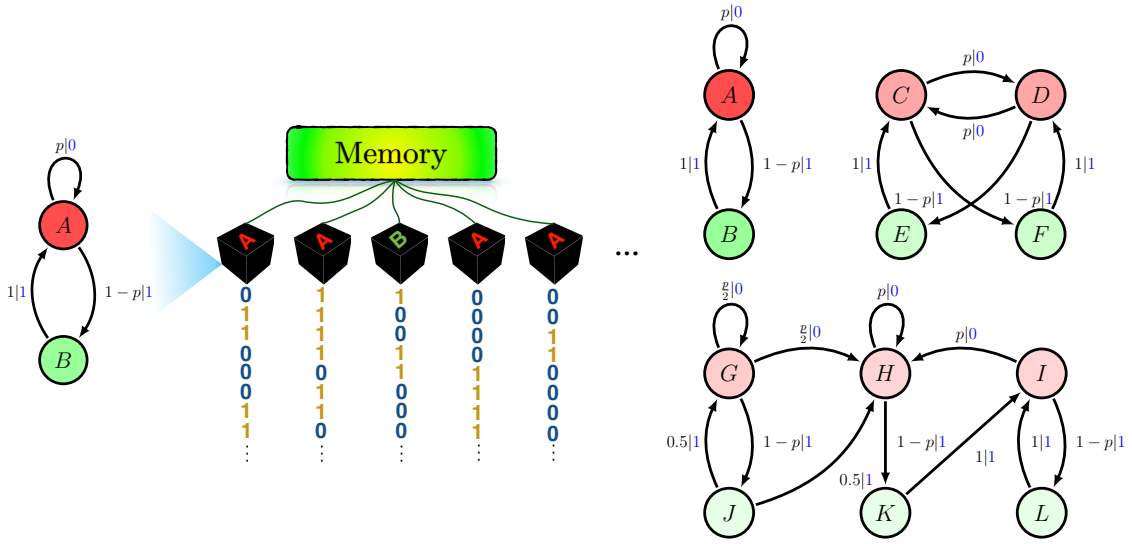


Figure 2.1: (a) Left: Even Process ϵ-machine. Right: Schematic of simultaneous generation problem. Each black box contains an Even Process generator. They all share the same memory for tracking the individual generator states. (b) Alternative HMMs: Even Process generators, each with different memory costs. Top: Unifilar HMMs, since for every state and symbol there is at most one outgoing edge from that state emitting that symbol. Below: Nonunifilar HMM, since for example state G can go to different states G or H emitting symbol 0.

$\mathrm{H}(\mathcal{S})$?

A *unifilar* HMM is one in which each row of each substochastic matrix has at most one nonzero element. Informally, this means the current state and next symbol uniquely determine the next state. Many statistical and informational quantities can be calculated in closed form from a process's unifilar HMM; see Ref. [80] and discussion therein. For example, in Fig. 2.1b the top two HMMs are unifilar and the bottom one is nonunifilar.

For a given process, finding the optimal HMM for simultaneous generation—an HMM with minimum state-entropy $\mathrm{H}(\mathcal{S})$—in the space of all HMMs is still an open question. Restricting to the space of unifilar HMMs, though, the optimal HMM can be found. It is the *ϵ-machine* [22], first introduced in Ref. [56]. ϵ-Machine states $\boldsymbol{\mathcal{S}}$ are called *causal states*. Due to ϵ-machine's unifilarity property, every generated past uniquely maps to a causal state. A process' *statistical complexity* C_μ [22] is the the Shannon entropy of the ϵ-machine's stationary state distribution: $C_\mu = \mathrm{H}(\mathcal{S}) = -\sum_{\sigma\in\mathcal{S}} \Pr(\sigma) \log_2 \Pr(\sigma)$. And,

this is the required memory for simultaneous generation.

Attempts have been made to find smaller models among *nonunifilar* HMMs [57]. As of now, though, only a handful of examples exist [57–59, 81]. Practically speaking, the ϵ-machine is the most memory-efficient algorithm for generating stochastic processes. Its memory C_μ has been determined for a wide range of physical systems [82–88]. Helpfully, it and companion informational measures are directly calculable from the ϵ-machine, many in closed-form [80].

We denote the process generated by the physical system with Hamiltonian Eq. (2.3) at temperature T by $\mathcal{P}(N, T)$. How do we construct the ϵ-machine that simulates the process $\mathcal{P}(N, T)$? First, we must define process' *Markov order* [77]: the minimum history length R required by any simulator to correctly continue a configuration. Specifically, R is the smallest integer such that:

$$\mathbb{P}(X_t | \ldots, X_{t-2}, X_{t-1}) = \mathbb{P}(X_t | X_{t-R}, \ldots, X_{t-2}, X_{t-1}) \ . \tag{2.4}$$

[More precisely, an ensemble of simulators must yield an ensemble of configurations that agree (conditioned on that past) with the process' configuration distribution.]

Reference [1, Eqs. (84) − (91)] showed that for any finite and nonzero temperature T, $\mathcal{P}(N, T)$ has Markov order N. One concludes that sufficient information for generation is contained in the configuration of the N previously generated spins. (Figure 2.2a shows this fact schematically for $N = 2$.) More importantly, the ϵ-machine that simulates $\mathcal{P}(N, T)$ has 2^N causal states and those states are in one-to-one correspondence with the set of length-N spin configurations.

Second, another key process characteristic is its *cryptic order* [89, 90]: the smallest integer K such that $\mathrm{H}[\mathcal{S}_K | X_0 X_1 \ldots] = 0$, where $\mathrm{H}[W|Z]$ is the conditional entropy [21] and $\mathcal{S}_K$ is the random variable for K^{th} state of the ϵ-machine after generating symbols $X_0, X_1 \ldots$. Using the fact that the ϵ-machine's states are in one-to-one correspondence with the set of length-N spin configurations [1], it is easy to see that $\mathcal{P}(N, T)$'s cryptic order $K = N$, the Markov order. We will use this fact in the quantum algorithm construction to follow.

Figure 2.2b shows the ϵ-machines of the processes $\mathcal{P}(N, T)$ for $N = 1, 2$, and 3. Let's

explain. First, consider the spin process $\mathcal{P}(1,T)$ that, as we pointed out, is a Markov-order $R=1$ process. This means to generate the process the simulator only need remember the last spin generated. In turn, this means the ϵ-machine (Fig. 2.2b left) has two states, $\uparrow$ and $\downarrow$. If the last observed spin is $\uparrow$, the current state is $\uparrow$ and if it is $\downarrow$, the current state is $\downarrow$. We denote the probability of generating a $\downarrow$ spin given a previous generated $\uparrow$ spin by $p_{\uparrow}^{\downarrow}$. The probability of an $\uparrow$ spin following a $\uparrow$ spin is the complement.

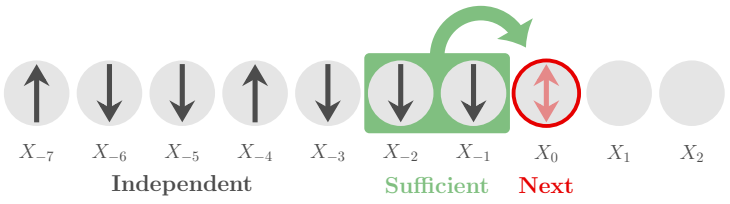
Second, consider the process $\mathcal{P}(2,T)$ with Markov-order $R=2$ and so longer-range interactions. Sufficient information for generation is contained in the configuration of the two previously generated spins. Thus, the ϵ-machine (Fig. 2.2b middle) has four states that we naturally label $\uparrow\uparrow$, $\uparrow\downarrow$, $\downarrow\uparrow$, and $\downarrow\downarrow$. If the last observed spin pair $x_{-1}x_0$ is $\uparrow\downarrow$, the current state is $\uparrow\downarrow$. Given this state, the next spin will be $\uparrow$ with probability $p_{\uparrow\downarrow}^{\uparrow}$ and $\downarrow$ with probability $p_{\uparrow\downarrow}^{\downarrow}$. Note that this scheme implies that each state has exactly two outgoing transitions. That is, not all state-to-state transitions are allowed in the ϵ-machine.

Having identified the state space, to complete the ϵ-machine construction we determine the ϵ-machine transition probabilities $\{T^{(x)}\}_{x\in\mathcal{A}}$. To do this, we first compute the transfer matrix V for the Ising N-nearest neighbors with the Hamiltonian in Eq. (2.3) at temperature T and then extract conditional probabilities, following Ref. [1]. (See the Method section following for details.) The minimum memory for simultaneous generation or, as it is called, the statistical complexity $C_\mu(N,T)$ of process $\mathcal{P}(N,T)$ follows straightforwardly from the process' ϵ-machine.

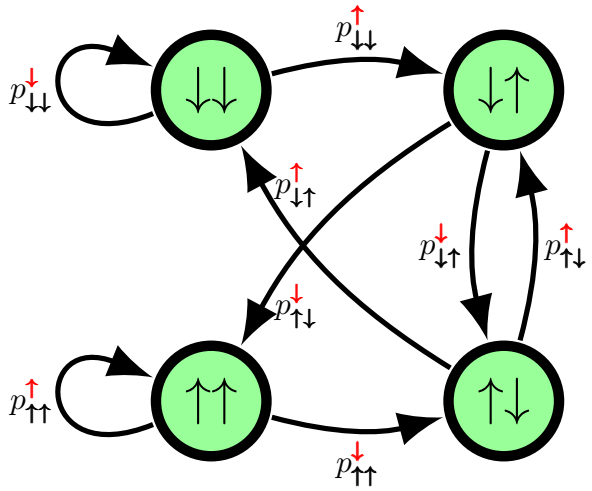
2.4.2 Quantum Simulator

By studying a specific process (similar to the ϵ-machine in left of Fig. 2.2b), Ref. [8] recently demonstrated that quantum mechanics can generate stochastic processes using less memory than C_μ. This motivates a search for more efficient quantum simulators of processes with richer correlational structure.

A process' quantum simulator is a pair $\{f,\mathcal{M}\}$, where $f:\mathcal{A}^\infty\to\Omega$ is a function from the set $\mathcal{A}^\infty$ of past sequences to a set of quantum states Ω and $\mathcal{M}$ is some measurement process. Given a particular past $x_{-\infty:0}$, applying the measurement $\mathcal{M}$ to the quantum state $f(x_{-\infty:0})$ leads to a correct probability distribution over future $\mathbb{P}(x_{0:n}|x_{-\infty:0})$. If



(a)



(b)

Figure 2.2: (a) A Markov order-N process generates a spin configuration from left-to-right. Markov order $N = 2$ shown. The values of an isolated spin S_0, say, is undetermined. To make this (stochastic) choice consistent with the overall process and the particular instantiation on the left, it is sufficient to consider only the previous N (2) spins (highlighted in green). (b) ϵ-Machine generators of 1D-configuration stochastic processes in Dyson-Ising systems of increasing correlational complexity ($N = 1, 2, 3$): $\mathcal{P}(1, T)$ (left), $\mathcal{P}(2, T)$ (middle) and $\mathcal{P}(3, T)$ (right).

$f(\cdot)$ is a deterministic function, the simulator is called unifilar and if f is probabilistic function, the simulator is called nonunifilar. After generating $x_{0:n}$, the new past is $x_{-\infty:n}$ and f can be used to map it to the new quantum state $f(x_{-\infty:n})$. By repeating the same measurement and mapping procedure, we generate a realization of the process. One can also define quantum simulator in a way that $\mathcal{M}$ automatically maps $f(x_{-\infty:0})$ to the correct quantum state $f(x_{-\infty:n})$ and generates the correct probability distribution over $x_{0:n}$ [2].

Reference [12] introduced a class of unifilar simulators, called *q-machines*, that can generate arbitrary processes. As in the classical setting, nonunifilar quantum simulators are much less well understood [60, 61, 81]. The q-machine construction depends on an

encoding length L, each with its own quantum cost $C_q(L)$. Each of these simulators simulate the same process correctly. It is known that the cost $C_q(L)$ is constant beyond the process' cryptic order [90]. Based on numerical evidence, it is conjectured that this is also the minimal C_q value. Thus, we restrict ourselves to this choice ($L = K$) of encoding length and refer simply to the q-machine and its cost C_q.

The q-machine's quantum memory C_q is upper-bounded by C_μ, with equality only for the special class of zero-cryptic-order processes [90]. And so, C_μ/C_q gives us our quantitative measure of *quantum advantage*.

Reference [13] recently introduced efficient methods for calculating C_q using spectral decomposition. Those results strongly suggest that the q-machine is the most memory-efficient among all unifilar quantum simulators, but as yet there is no proof. The quantum advantage C_μ/C_q has been investigated both analytically [2,10,12,13,15] and experimentally [17].

A process' q-machine is straightforward to construct from its ϵ-machine. First, since the ϵ-machine is unifilar, every generated past realization maps to a unique causal state. Second, every causal state σ_i maps to a pure quantum state $|\eta_i\rangle$. Using these two maps we can map every generated past realization uniquely to a quantum state. Each signal state $|\eta_i\rangle$ encodes the set of length-K (cryptic order) sequences that may follow σ_i, as well as each corresponding conditional probability:

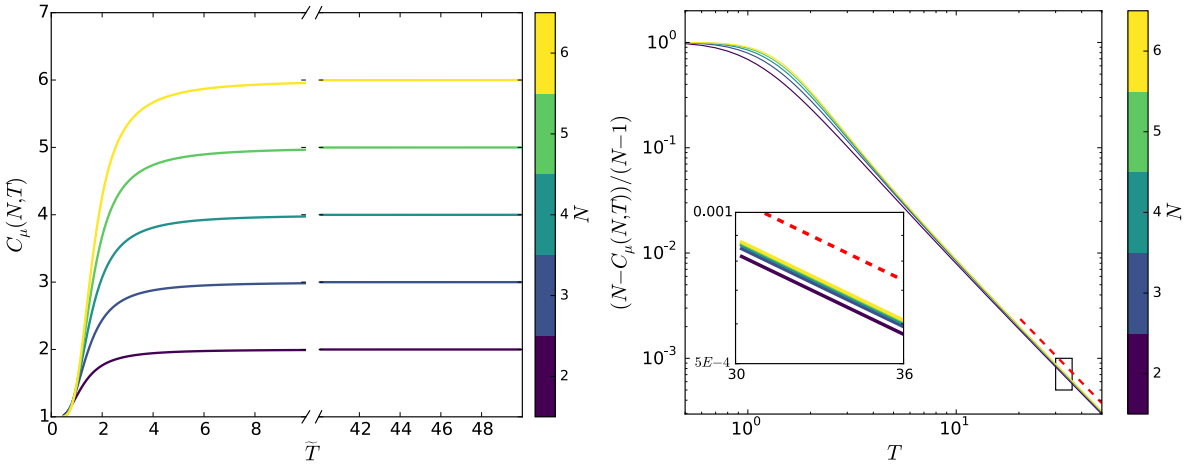
$$|\eta_i\rangle \equiv \sum_{w \in \mathcal{A}^K} \sum_{\sigma_j \in \boldsymbol{\mathcal{S}}} \sqrt{\mathbb{P}(w, \sigma_j|\sigma_i)}\ |w\rangle\,|\sigma_j\rangle\ , \tag{2.5}$$

where w denotes a length-K sequence and $\mathbb{P}(w, \sigma_j|\sigma_i) = \mathbb{P}(X_0 \cdots X_{K-1} = w, \mathcal{S}_{K-1} = \sigma_j|\mathcal{S}_0 = \sigma_i)$. The resulting Hilbert space is the product $\mathcal{H}_w \otimes \mathcal{H}_\sigma$. Factor space $\mathcal{H}_\sigma$ is of size $|\mathcal{S}|$, the number of classical causal states, with basis elements $|\sigma_i\rangle$. Factor space $\mathcal{H}_w$ is of size $|\mathcal{A}|^K$, the number of length-K sequences, with basis elements $|w\rangle = |x_0\rangle \cdots |x_{K-1}\rangle$. Performing a projective measurement in the $|w\rangle\,|\sigma\rangle$ basis results in a correct probability distribution. After generation of a particular realization by measurement, the next corresponding quantum state can be indicated uniquely. This means we can repeat the measurement process and continue generating the process.

Now, let us return to simultaneous generation where the goal is to generate M process realizations simultaneously where each is statistically independent of the others. As before, we have M q-machines as in Fig. 2.1a. Also similar to the classical setting, we can reduce the amount of required memory by having the q-machines use a single shared memory. According to quantum coding theorem, we can encode the HMM states to reduce the amount of memory to $MS(\rho)$ qubits where $S(\cdot)$ is von Neumann entropy and ρ is the density matrix defined by:

$$\rho = \sum_i \pi_i \left|\eta_i\right\rangle \left\langle\eta_i\right| . \tag{2.6}$$

As a result, each q-machine needs $C_q = S(\rho)$ qubits of memory for simultaneous generation.



(a) (b)

Figure 2.3: (a) Classical memory $C_\mu(N,T)$ required for simulating process $\mathcal{P}(N,T)$ for interaction ranges $N = 1,\ldots,6$, a range of temperatures $T = 1,\ldots,50$, and $\delta = 2$. Note $C_\mu(\cdot)$ is an increasing function of N and T. (b) Rescaling the classical memory requirement $C_\mu(N,T)$ to $(N - C_\mu)/(N-1)$ shows a tight data collapse, which is especially strong at high temperatures ($T > 2$). The asymptotic behavior is a power-law with scaling exponent $\gamma = 2$. The inset zooms in to show C_μ's convergence with increasing N. While the figure shows the case $\delta = 2$, the slope γ at high T is independent of δ.

2.5 Simulation of Dyson Chain

We begin by considering the case where spin couplings decay with exponent $\delta = 2$. Figure 2.3a displays $C_\mu(N,T)$ and Fig. 2.4a displays $C_q(N,T)$—the C_μ and C_q of processes

$\mathcal{P}(N,T)$—versus T for interaction ranges $N = 1, \ldots, 6$. The most striking feature is that the classical and quantum memory requirements exhibit qualitatively different behaviors.

The classical memory increases with T, saturating at $C_\mu = N$, since all transitions become equally likely at high temperature. As a result there are 2^N equally probable causal states and this means one needs N bits of memory to store the system's current state. For example, in the nearest-neighbor Ising model (process $\mathcal{P}(1,T)$) high temperature makes spin-$\uparrow$ and spin-$\downarrow$, and thus the corresponding states, equally likely. [At $T = \infty$ these processes have only a single causal state and thus $C_\mu = 0$. This is a well known discontinuity that derives from the sudden predictive-equivalence of all of the causal states there.]

Also, in the low-temperature limit, this system is known to yield one of only two equally likely configurations—all spin-$\uparrow$ or all spin-$\downarrow$. In other words, at low temperature $p_\uparrow^\downarrow$ and $p_\downarrow^\uparrow$ converge to zero, while $p_\uparrow^\uparrow$ and $p_\downarrow^\downarrow$ converge to one. [It should be pointed out that at any finite temperature $p_\uparrow^\downarrow$ and $p_\downarrow^\uparrow$ are nonzero and, therefore, the ϵ-machine states remains strongly-connected.] This is reflected in the convergence of all curves at $C_\mu = 1$ bit. Equivalently, this means one needs only a single bit of memory to store the current state.

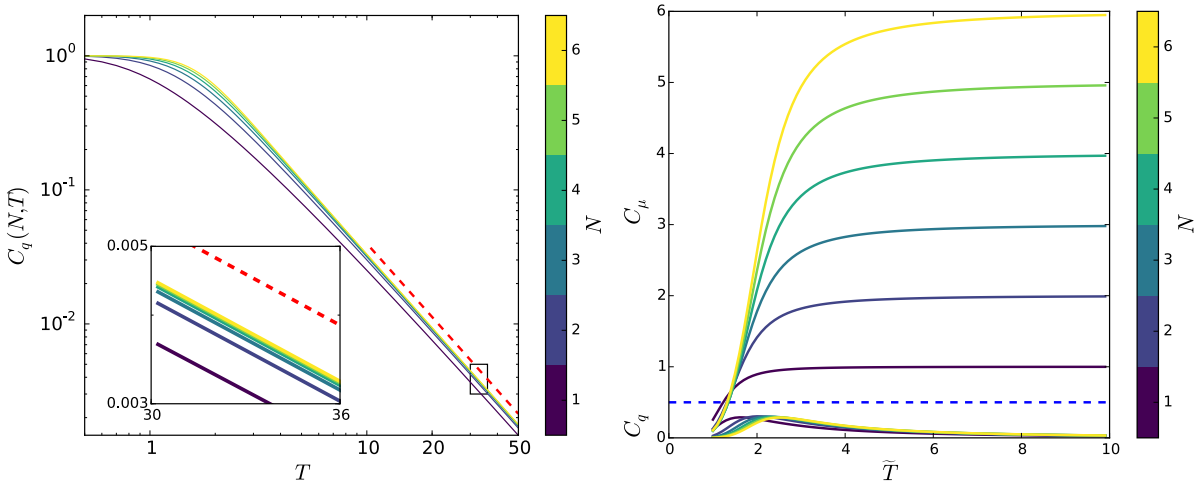
We can similarly understand the qualitative behavior of $C_q(N,T)$ for a fixed N. As temperature increases, all length-N signal states become equivalent. This is the same as saying that all length-N spin configurations become equally likely. As a consequence, the signal states approach one another and, thus, $C_q(N,T)$ converges to zero.

In the low temperature limit, the two N-$\uparrow$ and N-$\downarrow$ configurations are distinguished by the high likelihood of neighboring spins being of like type. This leads to a von Neumann entropy (C_q) of $S(\rho) = 1$ qubit.

Figure 2.3a reveals strong similarities in the form of $C_\mu(T)$ at different N. A simple linear scaling leads to a substantial data collapse, shown in Fig. 2.3b. The scaled curves $(N - C_\mu)/(N-1)$ exhibit power-law behavior in T for $T > 2$. Increasing the temperature to $T = 300$ (beyond the scale in Fig. 2.3b) numerical estimates from simulations indicate that this scaling is given by $\gamma \simeq 2.000$. The scaling determines how the classical memory

saturates at high temperature.

This behavior is generic for different coupling decay values $\delta > 1$ and, more to the point, the scaling is independent of δ. We do not consider $\delta < 1$, where the system energy becomes nonextensive.



(a) (b)

Figure 2.4: (a) $C_q(\cdot)$ is an increasing function of N, but a decreasing function of T and bounded by 1 qubit, independent of N and T. Quantum memory $C_q(N,T)$, similar to $C_\mu(N,T)$, shows a data collapse in N that is especially tight at high temperature ($T > 2$). The asymptotic behavior is a power-law with numerically estimated scaling exponent $\alpha = 2$. (Red dashed line.) The lower inset zooms to highlight convergence with increasing N. Though the curves are for the case with $\delta = 2$, the slope α at high T is independent of T. (b) Magnetic field effects on classical $C_\mu(N,T)$ and quantum $C_q(N,T)$ memory requirements for simulating the processes generated by Hamiltonian $\widehat{\mathcal{H}}_N$ for $N = 1, \ldots, 6$ over a range of temperatures $T = 1, \ldots, 10$ at $B = 0.3$. $C_q(N,T)$ curves are those under the dashed blue line.

Now, we can analyze the decrease in C_q with temperature. Figure 2.4a shows that C_q is also a power-law in T. By measuring this scaling exponent in the same way as above, we determined that $\alpha \simeq 2.000$. Furthermore, we find analytically that for high T:

$$C_q(N,T) \propto \frac{\log_2(T)}{T^2} \ . \tag{2.7}$$

To see this, first consider nearest-neighbor coupling $N = 1$. Due to symmetry we have $p \equiv \Pr(\uparrow \mid \uparrow) = \Pr(\downarrow \mid \downarrow) = F/D$, where $F = \exp(\beta J)$ and $D = \exp(\beta J) + \sqrt{\exp(-2\beta J)}$ with $\beta = 1/T$. At high temperature β is small and we have $D = 2 + \beta^2$ and $F = 1 + \beta + \beta^2$.

Again, by symmetry we have $\pi_1 = \pi_2 = 1/2$ and, therefore, the density matrix in Eq. (2.6) is:

$$\rho = \begin{pmatrix} 1/2 & \sqrt{p(1-p)} \\ \sqrt{p(1-p)} & 1/2 \end{pmatrix} , \tag{2.8}$$

which has two eigenvalues: $\beta^2/4$ and $1 - \beta^2/4$. As a consequence C_q, being ρ's von Neumann entropy, is:

$$C_q = S(\rho) \simeq -\left(\frac{\beta^2}{4}\log_2\frac{\beta^2}{4} + \left(1-\frac{\beta^2}{4}\right)\log_2\left(1-\frac{\beta^2}{4}\right)\right) \simeq \frac{\log_2(T)}{2T^2} . \tag{2.9}$$

Examining the numerator, for any $r > 0$ we have $\log_2(T) < T^r$. So, for large T:

$$\frac{1}{T^2} < \frac{\log_2(T)}{T^2} < \frac{1}{T^{2-r}} , \tag{2.10}$$

for all $r > 0$. This explains the fat tails of C_q for large T and establishes that for $N = 1$ the scaling exponent is $\alpha = 2$.

Increasing the temperature the link between spins weakens. At high temperature the only important neighbor is the nearest. As a consequence, the high temperature behavior is similar to the case of $N = 1$ and, in addition, it is independent of N. This verifies and adds detail to our numerical estimate.

This behavior is generic for different coupling decay values $\delta > 1$ and, moreover, the scaling exponent α is independent of δ. Notably, in this case no rescaling is required. The exponent directly captures the extreme compactness of high-temperature quantum simulations.

Taking these results together, we can now appreciate the substantial relative advantage of quantum versus classical simulators.

Define the *quantum advantage* η as the ratio of the minimum required memory for the classical simulation to the minimum required memory for the quantum simulation:

$$\eta(N,T) \equiv C_\mu(N,T)/C_q(N,T) . \tag{2.11}$$

For fixed temperature $T \gtrapprox 2$, $C_\mu(N,T)$ is approximately linear in N and for a fixed N is approximately independent of T. As a consequence, the asymptotic quantum advantage

is:

$$\eta(N,T) \propto N\frac{T^2}{\log_2(T)} \ , \tag{2.12}$$

which scales faster than any T^r for $r < 2$. Thus, the answer to our motivating question is that the quantum advantage does, in fact, display scaling: it increases with interaction range N and also increases strongly with temperature T.

Up to this point we focused on finite interaction-range systems, interpreting the chosen models as a family of approximations to the Dyson model. Consider, though, Dyson's original spin chain [63] which has infinite-range interactions. In this case, the classical memory cost of simulation diverges: $\lim_{N\to\infty} C_\mu(N,T) \to \infty$. That is, it is impossible to simulate the Dyson model classically. In contrast, quantum memory cost is finite—$\lim_{N\to\infty} C_q(N,T) < 1$ qubit—and so it can be simulated quantally. There is perhaps no clearer statement of quantum advantage.

Naturally, one might ask how our results are modified by the presence of an external magnetic field. Consider the one-dimensional ferromagnetic Ising spin chain with Hamiltonian:

$$\widehat{\mathcal{H}}_N = -\sum_i \sum_{k=1}^{N} \frac{J_0}{k^\delta} s_i s_{i+k} - \sum_i B s_i \ . \tag{2.13}$$

Figure 2.4b shows that, due to symmetry breaking at low temperature, both $C_q(N,T)$ and $C_\mu(N,T)$ converge to zero. (All spins at low temperature align with magnetic field and, as a consequence, no memory is needed.) The high temperature behaviors for both memory costs are the same as before, though, and the quantum advantage remains the same.

2.6 Conclusions

It is notoriously hard to find quantum advantage and even harder to prove [91]. We found such an advantage in the realm of stochastic process simulation. Concretely, we analyzed the N-nearest neighbor Ising spin system and demonstrated that its quantum advantage displays a generic scaling behavior—quadratic in temperature and linear in interaction range. What does this mean? The most striking conclusion is that a strongly interacting

classical system can be simulated with unbounded quantum advantage. One stark contrast is that it is impossible to classically simulate Dyson's original spin chain while quantum simulators can do so and with finite memory cost.

How broadly might we expect to see this quantum advantage? Or, is it merely a feature of strongly coupled spin systems? Define a *universal* spin model as one that can simulate any other spin model. That is, by using the low-energy sector of such universal models, the physics of every classical spin model can be reproduced. Recently, Ref. [92] showed that the 2D Ising model with external fields is universal in this sense. This suggests that the quantum advantage described here may not be limited to the particular spin system we consider, but might also be universal. As a result, one should expect to see the quantum advantage for other physical systems.

The Ising model has lent great insight to condensed matter physics, however it is a classical model. Given that we are examining the difference between classical and quantum simulators, it is natural to wonder about this difference in the context of a truly quantum Hamiltonian. Is the quantum advantage amplified? Are there systems for which we find no quantum advantage? And, is this their defining characteristic?

Here, we studied the cost of exact simulation of stochastic processes. Both classical and quantum costs, though, can be very different when approximation is allowed. For example, at high (but finite) temperature, we can approximate the process $\mathcal{P}(N,T)$ as independent, identically distribution (IID). One does not require any classical or quantum memory to generate an IID process and, as a result, there would be no quantum advantage. Apparently, the difference between required classical memory for exact simulation and approximate simulation can be quite large. In contrast, the price we pay to go from approximate to exact quantum simulation is relatively small.

2.7 Appendix

We show how to construct the ϵ-machine simulator of the process $\mathcal{P}(N,T)$, following Ref. [93]. Consider a block of spins of length $2N$, divided equally into two blocks. We denote spins in the left (L) and right (R) halves by: s_i^L and s_i^R for $i = 1, \ldots, N$, respectively.

We map the left and right block configurations each to an integer η_* by:

$$\eta_* = \sum_{i=1}^{N} \left(\frac{s_i^* + 1}{2} \right) 2^{i-1} \ , \tag{2.14}$$

where $*$ can be either L or R. For each block we can have 2^N different configurations. Consequently, the label η_* varies between 0 and $2^N - 1$. The internal energy of a given block with configuration η_* is given by:

$$X_{\eta_*} = -B \sum_{i=1}^{N} s_i^* - \sum_{i=1}^{N-1} \sum_{k=1}^{N-i} J_i s_k^* s_{k+i}^* \ , \tag{2.15}$$

and the interaction energy between two blocks is:

$$Y_{\eta_L, \eta_R} = - \sum_{i=1}^{N} \sum_{k=1}^{i} J_i s_{N-k+1}^L s_k^R \ . \tag{2.16}$$

With these we construct the transfer matrix:

$$V_{\eta_L, \eta_R} = e^{-(1/2 X_{\eta_L} + Y_{\eta_L, \eta_R} + 1/2 X_{\eta_R})/T} \ . \tag{2.17}$$

The right eigenvector of V corresponding to the largest eigenvalue is denoted by u. Reference [1] shows that the ϵ-machine labeled-transition matrices can be written as:

$$T^{(x)}_{\eta_0, \eta_1} = \begin{cases} \frac{1}{\lambda} V_{\eta_0, \eta_1} \frac{u_{\eta_1}}{u_{\eta_0}}, & \eta_1 = \left(\lfloor \frac{\eta_0}{2} \rfloor + x(2^{N-1}) \right) \\ 0, & \text{otherwise} \end{cases} \ , \tag{2.18}$$

where $x \in \{0, 1\}$, 0 for spin down and 1 for spin up. Then, the ϵ-machine simulator of $\mathcal{P}(N, T)$ is $\left\{ \boldsymbol{\mathcal{S}}, \mathcal{A}, \{T^{(x)}\}_{x \in \mathcal{A}} \right\}$, where $\mathcal{A} = \{0, 1\}$ and $\boldsymbol{\mathcal{S}} = \{i : 0 \leq i \leq 2^N - 1\}$.

Chapter 3

The Ambiguity of Simplicity in Quantum and Classical Simulation

3.1 Overview

A system's perceived simplicity depends on whether it is represented classically or quantally. This is not so surprising, as classical and quantum physics are descriptive frameworks built on different assumptions that capture, emphasize, and express different properties and mechanisms. What is surprising is that, as we demonstrate, simplicity is ambiguous: the *relative* simplicity between two systems can *change sign* when moving between classical and quantum descriptions. Here, we examine the minimum required memory for simulation. We see that the notions of absolute physical simplicity at best form a partial, not a total, order. This suggests that appeals to principles of physical simplicity, via Ockham's Razor or to the "elegance" of competing theories, may be fundamentally subjective. Recent rapid progress in quantum computation and quantum simulation suggest that the ambiguity of simplicity will strongly impact statistical inference and, in particular, model selection.

3.2 Introduction

Beyond his theory of gravitation, development of the calculus, and pioneering work in optics, Newton engendered a critical abstract transition that has resonated down through the centuries, guiding and even accelerating science's growth: Physics began to perceive the world as one subject to concise mathematical Laws. Above, Newton suggests that

these Laws are not only a correct perception but they are also *simple*. Consequently, one should abandon the Ptolemaic epicycles for Newton's elegant $F = ma$ and $F_g \propto m_1 m_2 / r^2$.

The desire for simplicity in a theory naturally leads us to consider *simplicity as a means for comparing* alternative theories. Here, we compare the parsimony of two descriptions of stochastic processes—one classical and one quantum. Classical versus quantum comparisons have, of late, captured our attention both for reasons of principle and of experiment. *Quantum supremacy* holds that quantum systems behave in ways beyond those that can be efficiently simulated by classical computers [94]. A single cold 2D Fermi gas supports coexistence of both quantum mechanical states at its core and classical states on its periphery [95, 96]. The overriding impression is that now is an interesting time for the foundations of quantum mechanics. The following adds a new phenomenon to these debates on the balance of classical and quantum theories, as concerns the simplicity of their descriptions.

To start, we consider a Nature full of stationary stochastic processes. A theory, then, is a mathematical object capable of yielding a process' probabilities. We can straightforwardly say that one process is more random than another via comparing their temperatures or thermodynamic entropies. But how to compare them in terms of their structural simplicities? We make use of a well developed measure of simplicity in stochastic processes—the statistical complexity—a measure of internal memory [56] or the minimum required memory to simulate a process. It provides a concrete and interpretable answer to the question, which process is structurally simpler? By applying this comparison, we may order all processes from the simplest to the most complicated [97].

With recent progress in quantum computation [98–100], an interesting twist comes about if we add quantum mechanics to our modeling toolbox. Descriptions that act on a quantum substrate offer new and surprising options. For example, it was shown that a quantum mechanical description can lead to a simpler representation [2, 8, 12, 13, 60] and even in some cases infinitely simpler [11, 14]. Recently, this quantum advantage was verified experimentally [17]. Proceeding with these methods, we discover what is most surprising: the *relative simplicity* of classical and quantum descriptions can change. Specifically, there

are stochastic processes, A and B, for which classical theory says A is simpler than B, but quantum mechanics says B is simpler than A. What started out as a neat classical array is upended by a new quantum simplicity order. This means quantizing a simple classical model may not be as simple as quantizing a more complicated classical model. As a consequence model selection is complicated by the addition of a quantum model class.

3.3 Classical and Quantum Simplicity

We consider stationary, ergodic processes: each a bi-infinite sequence of random variables $X_{-\infty:\infty} = \ldots X_{-2}X_{-1}X_0X_1X_2\ldots$ where each random variable X_t takes some value x_t in a discrete alphabet set $\mathcal{A}$ and where all probabilities $\mathbb{P}(X_t, \ldots, X_{t+L})$ are time-invariant.

How is their degree of randomness quantified? Information theory [21] measures the uncertainty in a single observation X_0 via the *Shannon entropy*: $H[X_0] = -\sum_{x\in\mathcal{A}} \mathbb{P}(x) \log_2 \mathbb{P}(x)$ and the irreducible uncertainty per observation via the *entropy rate* [101]: $h_\mu = \lim_{L\to\infty} H[X_{0:L}]/L$. If we interpret the left half $X_{-\infty:0} = \ldots X_{-2}X_{-1}$ as the "past" and the right half $X_{0:\infty} = X_0X_1X_2\ldots$ as the "future", we see that the entropy rate is the average uncertainty in the next observable given the entire past: $h_\mu = H[X_0|X_{-\infty:0}]$. Thus, as we take into account past correlations, the naive uncertainty $H[X_0]$ reduces to h_μ.

How reducible is our uncertainty in the future $X_{0:\infty}$ knowing the past $X_{-\infty:0}$? The answer is given by the mutual information between the past and the future—the *excess entropy* [102]: $\mathbf{E} = I[X_{-\infty:0} : X_{0:\infty}]$. With h_μ and $\mathbf{E}$, we measure randomness and predictability, respectively.

Let's say we want to simulate a given process. To do this we write a computer code that follows an algorithm and allocate the memory the algorithm needs. For a given process *computational mechanics* [22] identifies the optimal algorithm—the process' *ϵ-machine*. This is a *unifilar hidden Markov model* [103] that uses only the minimum required memory for simulation. We view a process' ϵ-machine as the "theory" of a process in that it specifies a mechanism that exactly simulates a process' behaviors. In this way, computational mechanics supplements $\mathbf{E}$ and h_μ with a measure of structure—the minimum required amount memory to simulate the given process.

The ϵ-machine consists of *causal states* $\sigma \in \boldsymbol{\mathcal{S}}$ defined by an equivalence relation $\sim$ that groups histories, say $x_{-\infty:t}$ and $x_{-\infty:t'}$, that lead to the same future predictions $\mathbb{P}(X_{t:\infty}|\cdot)$: $x_{-\infty:t} \sim x_{-\infty:t'} \iff \mathbb{P}(X_{t:\infty}|x_{-\infty:t}) = \mathbb{P}(X_{t':\infty}|x_{-\infty:t'})$. From this, one concludes that a process' ϵ-machine is, in a well defined sense, its simplest predictive theory.

Translating this notion of simplicity into a measurable quantity, we ask: *What is the minimum memory necessary to* implement *optimal prediction?* The answer is the historical information stored in the ϵ-machine. Quantitatively, this is the Shannon entropy of the causal-state stationary distribution $\{\pi_\sigma\}$, the *statistical complexity*:

$$C_\mu = H[\boldsymbol{\mathcal{S}}] = -\sum_{\sigma \in \boldsymbol{\mathcal{S}}} \pi_\sigma \log_2 \pi_\sigma \ , \tag{3.1}$$

It is well known that the excess entropy is a lower-bound on this structural measure: $\mathbf{E} \leq C_\mu$. In fact, this relation is only rarely an equality [104]. And so, while $\mathbf{E}$ quantifies the amount to which a process is subject to explanation by its ϵ-machine "theory", this simplest theory is typically larger, informationally speaking (C_μ), than the predictability benefit it confers. That said, the ϵ-machine is the best (simplest) theory. Thus, we use C_μ to define our notion of classical simplicity. It provides an interpretable ordering of processes—process A is simpler than process B when $C_\mu^A < C_\mu^B$.

We may also consider the recently proposed quantum-machine representation of processes [8, 12, 13]. The quantum-machine consists of a set $\{|\eta_k(L)\rangle\}$ of pure *signal states* that are in one-to-one correspondence with the classical causal states $\sigma_k \in \boldsymbol{\mathcal{S}}$. Each signal state $|\eta_k(L)\rangle$ encodes the set of length-L words that may follow σ_k, as well as each corresponding conditional probability. Fixing L, we construct quantum states:

$$|\eta_j(L)\rangle \equiv \sum_{w^L \in \mathcal{A}^L} \sum_{\sigma_k \in \boldsymbol{\mathcal{S}}} \sqrt{\mathbb{P}ob(w^L, \sigma_k|\sigma_j)} \ |w^L\rangle \, |\sigma_k\rangle \ , \tag{3.2}$$

where w^L denotes a length-L word and $\mathbb{P}(w^L, \sigma_k|\sigma_j) = \mathbb{P}(X_{0:L} = w^L, \mathcal{S}_L = \sigma_k|\mathcal{S}_0 = \sigma_j)$. The resulting Hilbert space is the product $\mathcal{H}_w \otimes \mathcal{H}_\sigma$. Factor space $\mathcal{H}_\sigma$ is of size $|\boldsymbol{\mathcal{S}}|$, the number of classical causal states, with basis elements $|\sigma_k\rangle$. Factor space $\mathcal{H}_w$ is of size $|\mathcal{A}|^L$, with basis elements $|w^L\rangle = |x_0\rangle \cdots |x_{L-1}\rangle$.

The quantum measure of memory is the von Neumann entropy of the stationary state:

$$C_q = -\operatorname{Tr}(\rho \log \rho) \ , \tag{3.3}$$

where $\rho = \sum_i \pi_i |\eta_i\rangle \langle \eta_i|$. This quantum analog of memory is generically less than the classical: $C_q \leq C_\mu$. Also, due to the Holevo bound [8, 105], $\mathbf{E} \leq C_q$. Though rare in process space, the classical and quantum informational sizes are equal exactly when both models are "maximally simple" : $\mathbf{E} = C_q = C_\mu$.

3.4 Ising Chain Simplicity

The Ising spin-chain Hamiltonian is given by:

$$H = -\sum_{<i,j>} (J s_i s_j + b s_i) \ , \tag{3.4}$$

where s_i, the spin at site i, takes values $\{-1, +1\}$, J is the nearest-neighbor spin coupling constant, and b is the strength of the external magnetic field.

In equilibrium the bi-infinite chain of spin random variables defines a stationary stochastic process which has been analyzed using computational mechanics [106]. Importantly, spins obey a conditional independence: $\mathbb{P}(X_{0:\infty}|x_{-\infty:0}) = \mathbb{P}(X_{0:\infty}|x_0)$. That is, the "future" spins (right half) depend not on the entire past (left half) but only on the most recent spin x_0. The conclusion (see Appendix) is that the two-state Markov chain process is minimally represented by the ϵ-machine in Fig. 3.1. Using Eq. (3.1), the statistical complexity is directly calculated as a function of p and q. Figure 3.2 shows that C_μ is a monotonically increasing function of temperature T: $1 - C_\mu \propto T^{-2}$ at high T. In particular, for the three processes chosen at temperatures $T_\alpha < T_\gamma < T_\delta$, $C_\mu^\alpha < C_\mu^\gamma < C_\mu^\delta$.

Consider now the quantum representation of these spin configurations. Each causal state is mapped to a pure quantum state that resides in a spin one-half space [2]:

$$\begin{aligned} |\sigma_1\rangle &= \sqrt{p}\,|\uparrow\rangle + \sqrt{1-p}\,|\downarrow\rangle \\ |\sigma_2\rangle &= \sqrt{1-q}\,|\uparrow\rangle + \sqrt{q}\,|\downarrow\rangle \ . \end{aligned} \tag{3.5}$$

(We use a more compact spin up/down notation, rather than the quantum machine notation of Eq. (3.2).) Intuitively, the quantum overlap accounts for the fact that the

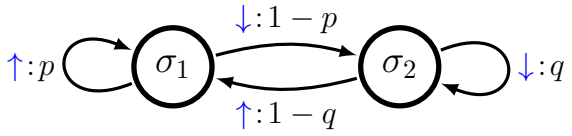


Figure 3.1: The ϵ-machine for the nearest-neighbor Ising spin chain has two causal states σ_1 and σ_2. If the last observed spin x_0 is up ($s_0 = +1$) the current state is σ_1 and if it's down ($s_0 = -1$) is σ_2. If the current state is σ_1, with probability p the next spin observed is up and, if the current state is σ_2, with probability q the next spin observed is down.

conditional predictions $\mathbb{P}(X_{0:\infty}|\sigma_1)$ and $\mathbb{P}(X_{0:\infty}|\sigma_2)$ share some subset of future outcomes. The density matrix is then:

$$\rho = \pi_1 \left|\sigma_1\right\rangle \left\langle\sigma_1\right| + \pi_2 \left|\sigma_2\right\rangle \left\langle\sigma_2\right| \ . \tag{3.6}$$

Computing the quantum analog $C_q = -\operatorname{Tr}(\rho \log \rho)$ as a function of temperature, Fig. 3.2 shows that this quantum size is generically well below the classical size C_μ. Thus, the quantum theory for the Ising chain is simpler than the classical: $C_q^\alpha < C_\mu^\alpha$, $C_q^\gamma < C_\mu^\gamma$, and $C_q^\delta < C_\mu^\delta$. Given the nature of progress in quantum information and computation [19, 107], it is notable, but perhaps no longer so surprising, that there exists such a quantum representational advantage.

3.5 Ambiguity of Simplicity

Absolute sizes aside, what can we say about the associated process *rankings*? How does the notion of "simpler" survive the transition from classical to quantum description?

Observe (Fig. 3.2) that, unlike the classical measure C_μ, the quantum simplicity C_q is not monotonic in temperature: $C_q^\alpha < C_q^\delta < C_q^\gamma$. Moreover, the maximum C_q occurs at temperature $T_{C_q} \simeq 1.63$ while the excess entropy is maximized at temperature $T_{\mathbf{E}} \simeq 1.53$. These straightforward observations provide the kernel of several counterintuitive consequences.

First, what is the consequence of nonmonotonicity? Take the processes α and γ in Fig. 3.2. Classically and quantally, α is simpler than γ. In contrast, the ranking of processes γ and δ changes, $C_\mu^\gamma < C_\mu^\delta$ and $C_q^\gamma > C_q^\delta$.

In this way, even the familiar 1D Ising spin chain illustrates what is a general phenomenon—the ambiguity of simplicity. How general? Consider two generic pro-

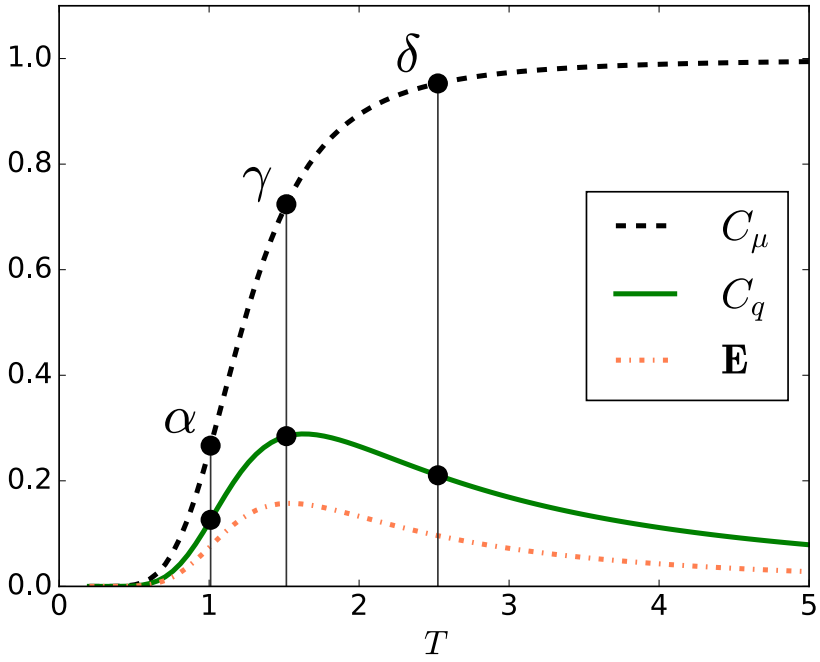


Figure 3.2: Classical and quantum measures of Ising chain simplicity: Statistical complexity C_μ, quantum state complexity C_q, and excess entropy $\mathbf{E}$ versus temperature T in units of J/k_B at $b = 0.3$ and $J = 1$. ($C_\mu(T)$ and $\mathbf{E}(T)$ after Ref. [1] and $C_q(T)$ after Ref. [2].) Three particular spin processes are highlighted α, γ, and δ at temperatures T_α, T_γ, and T_δ.

cesses A and B, for which no change in ranking occurs under the quantum lens. This indicates a *consistency* between the two representational viewpoints, at least with respect to processes A and B: $C_\mu^A > C_\mu^B$ and $C_q^A > C_q^B$. Figure 3.3(left) illustrates this circumstance. It can also be the case that the simplicity ranking of A and B *changes* when moving from classical to quantum representation. We refer to this as *ambiguity*. See Fig. 3.3(right). One concludes that the basic question—"Which process is simpler?"—no longer has a well defined answer.

How generic are consistency and ambiguity in the Ising spin chain parameter space? In Fig. 3.4 we construct an ambiguity diagram that compares all pairs of processes at temperatures T_1 and T_2 in the range $[0, 5]$. There, we fix the magnetic field $b = 0.3$ and coupling constant $J = 1$. We find that the only consistent pairs are those within a shrinking envelope around the axes ($T_1 = 0$ and $T_2 = 0$). The bulk of parameter space, then, contains ambiguously ranked pairs. The singular feature of the diagram is the leftmost point along

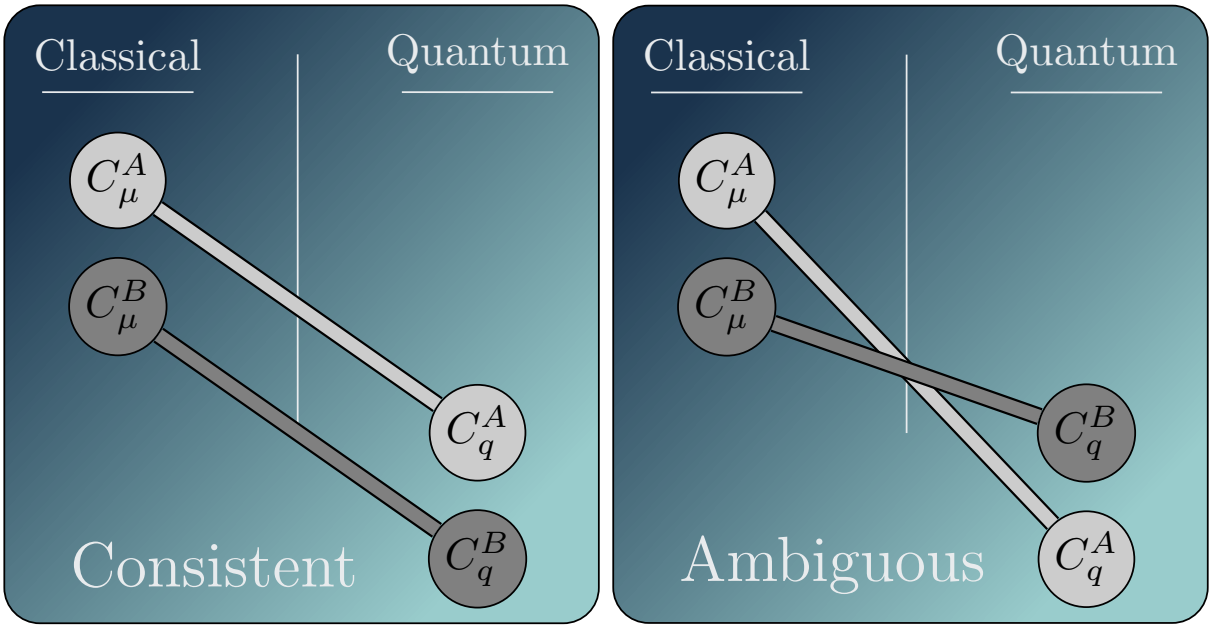


Figure 3.3: (left) Classical and quantum rankings provide a consistent interpretation of which process is simpler. (right) Rankings reverse. And so, the question of simplicity is ambiguous.

the boundary between the two regimes. This occurs at the temperature $T_{C_q} \simeq 1.63$ where we find the maximum value of C_q. Monotonicity of C_μ ensures that the transition between consistency and ambiguity depends on a reordering of C_q (not C_μ) values.

A notable case occurs when $b = 0$: there is no external-field induced symmetry-breaking. As a consequence, $C_\mu = 1$ for all temperatures and C_q is a decreasing function of temperature. This means that, for every pair of temperatures, we are at the border line of ambiguity. Classically they are as simple as each other, but quantally the system at the higher temperature is simpler.

3.6 Robustness of ambiguity

One may object that this ambiguity is merely an artifact of the particular quantum construction or its size measure C_q. This is a valid concern, especially since minimality of this (or any other quantum representation) has not been established. Critically, the essence of ambiguity does not depend on this contingency, as we now show.

Denote by $\widetilde{C_q}$ the memory measure of an optimal quantum model[1] $\widetilde{Q}$ built according

[1]The quantum model with minimum Von Neumann entropy over it's states.

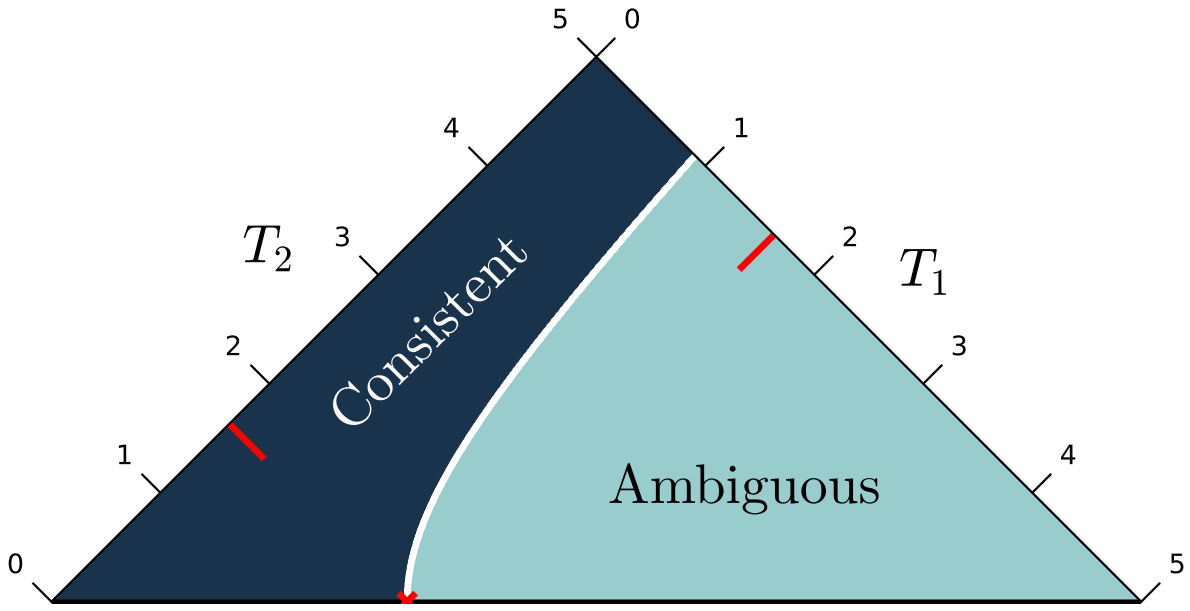


Figure 3.4: Ambiguity diagram for Ising spin chain: Each point corresponds to a pair of Ising spin chains at temperatures T_1 and T_2 with $J = 1$ and $b = 0.3$. Given the inherent symmetry, the figure shows only half of the $T_1 \times T_2$ square. Consistency is found near the $(T = 0)$ axes, while ambiguity dominates the remainder of parameter space. Curved boundary between these two regions ends at a temperature corresponding to $\max(C_q)$: $T_{C_q} \simeq 1.63$ (marked as a red dash).

to some hypothetical, quantum scheme. Since $\widetilde{C_q}$, like C_q, is also bounded between $\mathbf{E}$ and C_μ [8, 105], we can define sufficient criteria for consistency and ambiguity between $\widetilde{C_q}$ and C_μ. We assume that the hypothetical model $\widetilde{Q}$ is no less efficient than the original quantum-machine: $\widetilde{C_q} \leq C_q$.

Assume that for processes A and B, B is classically simpler. Then, the stronger criterion $\mathbf{E}^A > C_q^B$ ensures that any $\widetilde{Q}$ must yield consistency in rankings and is therefore, what we call, *certainly consistent*. See Fig. 3.5(left). Similarly, if $\mathbf{E}^B > C_q^A$, we know that any $\widetilde{Q}$ must yield an ambiguous ordering and is *certainly ambiguous*. See Fig. 3.5(right).

Figure 3.6 illustrates these stricter relations within the same Ising parameter region; compare Fig. 3.4. The central region does not satisfy either strict constraint. As expected, the certainly consistent (ambiguous) area is a proper subset of the consistent (ambiguous) area.

One concludes that no matter what future improvements may be found in quantum

representations, these "certain" subregions are robust. This is a strong statement about how one can or cannot systematically rank the simplicity of systems classically and quantally. Again, the basic Ising spin chain is sufficiently rich to illustrate these new phenomena.

3.7 Discussion

How common is ambiguity? The Appendix shows that it is quite common in the analogous (nearest-neighbor, ferromagnetic) two-dimensional Ising system. Perhaps, however, the ambiguity of simplicity is special to spin systems. The Appendix establishes that it is, in fact, a much more general phenomenon, by introducing a set of easily satisfied conditions such that two simplicity functions over a set of structured objects must yield ambiguous ordering. In particular, taking the space of all ϵ-machines as a set and C_μ and $\widetilde{C_q}$ as the two measures , we find that these conditions are satisfied. The general consequence is that either the two measures selected are trivially equal or ambiguity must exist. In other words, if the world is not ambiguous, quantum mechanics cannot simplify its explanation. One concludes that ambiguity is necessary for quantum simplification.

3.8 Conclusions

The comparison of classical and quantum descriptions calls into question basic scientific practices that rely on a belief in the simplicity of the physical world. These two worlds disagree on simplicity ranking. Monitoring model simplicity is far from being the sole domain of physics. It is key in a variety of statistical inference tasks, notably in model selection [108]. Thus, the ambiguity of simplicity will have major practical consequences in a future that relies on quantum computing instead of classical.

Imagine competing models of some finite data $\mathcal{D}$. In Bayesian inference, one widely employed methodology, choosing one model over another requires specifying a prior probability distribution over models [109]. Such priors are commonly constructed to favor simpler models. Indeed, there is a long history of methods that avoid overfitting to data by incorporating simplicity measures into model selection, including Akaike's Information Criterion [110], Boltzmann Information Criterion [111], Minimum Description Length [112],

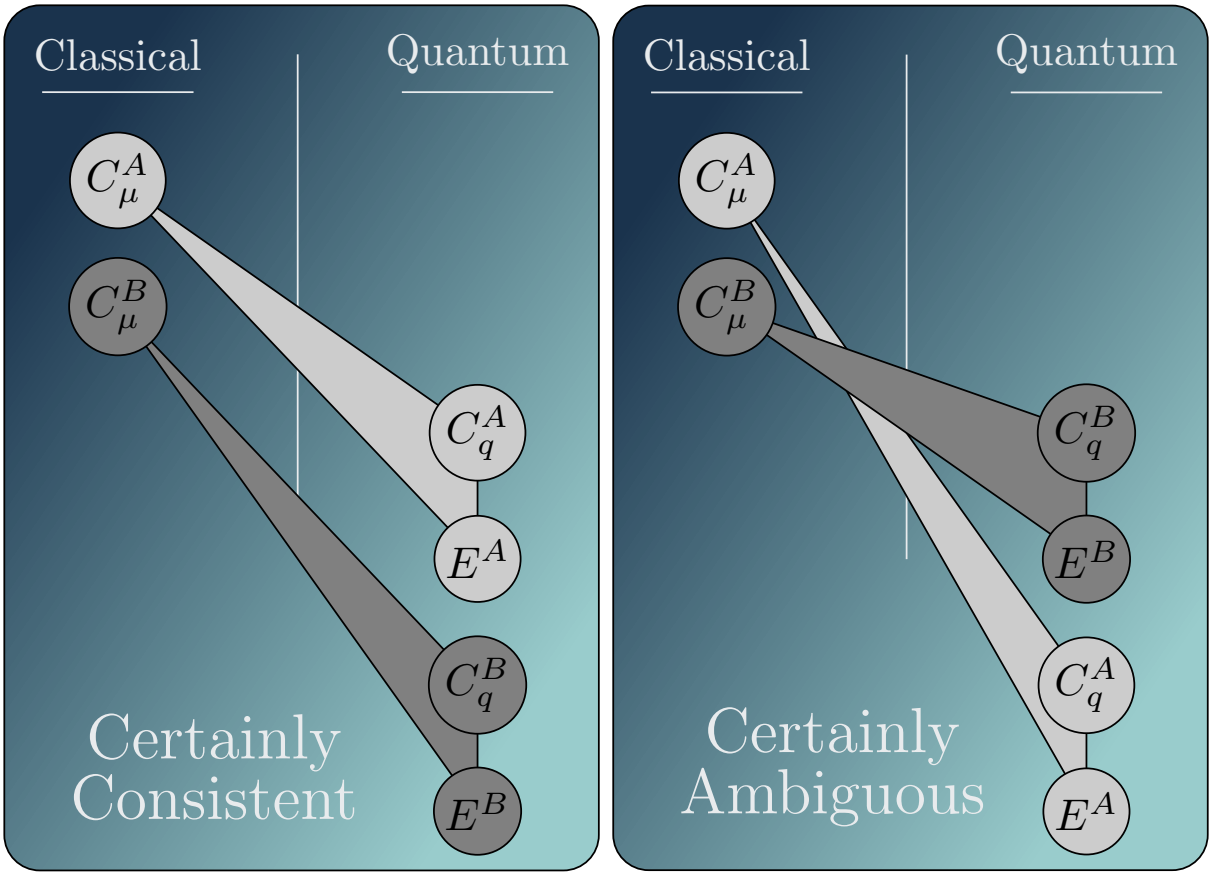


Figure 3.5: Constraining hypothetical, as-yet-unknown frameworks for building quantum models $\widetilde{Q}$: Appealing to size measures C_q and $\mathbf{E}$ and without knowing any further details about $\widetilde{Q}$, we can still identify processes for which classical and quantum simplicity orderings must *certainly* be consistent or ambiguous. Cases exist that fall into neither of these stricter categories.

and Minimum Message Length [113].

Classically, we may find that model A is simpler than B and increase its prior accordingly. Given that the two likelihoods $\mathbb{P}(\mathcal{D}|A)$ and $\mathbb{P}(\mathcal{D}|B)$ are similar enough, our inference identifies A as preferred. As we showed, the tables may turn dramatically when evaluating quantum models; we might find there that B is much simpler. We must then reconcile the fact that the quantum lens reveals a different answer.

We introduced the ambiguity of simplicity focusing on classical and quantum descriptions of classical processes. Quantum supremacy [94] suggests we go further to explore how (and if) ambiguity manifests when modeling quantum processes. This can be probed in the 1D quantum Heisenberg spin chain [67], for example. Measuring each spin in the

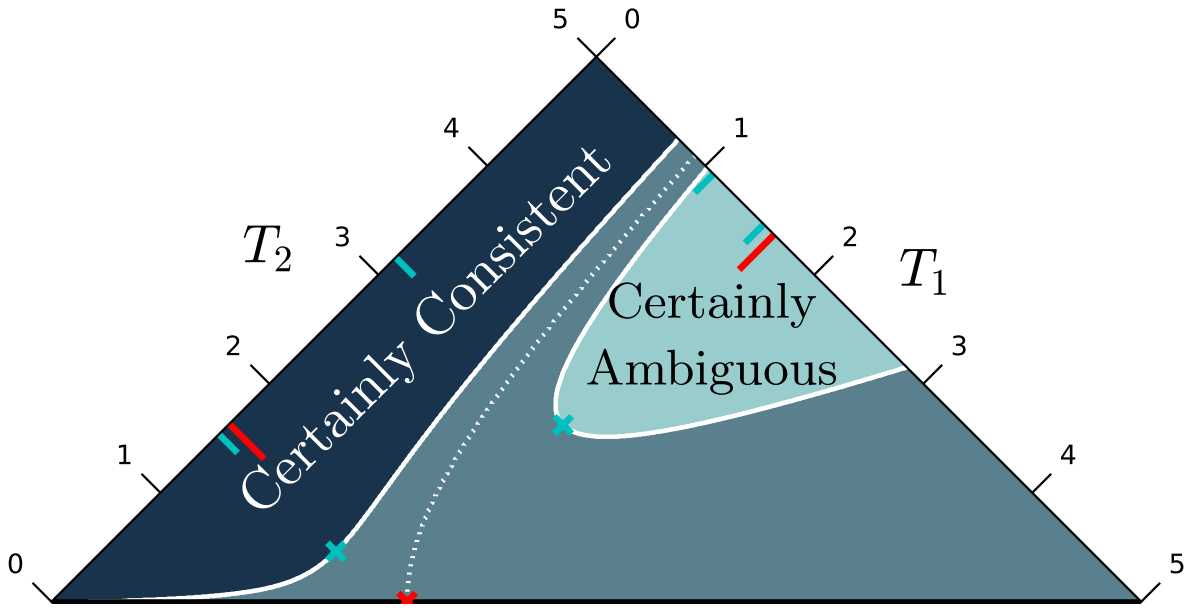


Figure 3.6: Certain ambiguity diagram: Each point corresponds to a pair of Ising spin chains at temperatures T_1 and T_2 with $J = 1$ and $b = 0.3$. Dashed line marks Fig. 3.4's consitent-ambiguous border. Certainly consistent (ambiguous) is a proper subset of consistent (ambiguous). Local extrema of $\max(\mathbf{E})$ and $C_q = \max(\mathbf{E})$ along the new boundaries are marked with short blue lines at the corresponding temperatures. Long red lines mark the same values as in Fig. 3.4.

z-direction yields a stochastic process—one that can be described classically or quantally. The Heisenberg spin chain is realized experimentally in the quasi-1D magnetic order found in antiferromagnetic $KCuF_3$ crystals [114–116]. One can then adapt the methods of 1D chaotic crystallography [88] to extract the ϵ-machine and quantum-machine descriptions of the quantum crystalline structure from scattering measurements. These and perhaps other experiments will provide an entrée to analyzing the ambiguity of simplicity in quantum systems.

3.9 Appendices

To ground the notion of simplicity, the main text couched the discussion in terms as physical (and familiar) as possible by considering the Ising spin chain from statistical physics [117]—a model that historically played a critical role in understanding phase transitions [118], spin glasses [119], and lattice gasses [120]. Its impact has reached well beyond physics, too, to ecology [121], financial economics [122], and neuroscience [123].

Specifically, the main text focused on the one-dimensional nearest-neighbor Ising spin chain in the thermodynamic limit, showing how it inherently contains an ambiguous simplicity ordering. Here, we provide additional details underlying that analysis, generalize the result to show that ambiguity also appears in the, perhaps even more familiar, 2D Ising lattice, and finally establish the robustness of ambiguity via a theorem that lays out its most basic conditions.

3.9.1 On Spin Chain Simplicity

Importantly, spins in the 1D chain obey a conditional independence: $\mathbb{P}(X_{0:\infty}|x_{-\infty:0}) = \mathbb{P}(X_{0:\infty}|x_0)$. That is, the "future" spins (right half) depend not on the entire past (left half) but only on the most recent spin x_0. Therefore, spin configurations resulting from the Hamiltonian in Eq. (3.4) can be modeled by a simple two-state Markov chain consisting of up ($\uparrow$) and down ($\downarrow$) states with self-transition probabilities [106]:

$$p \equiv \mathbb{P}(\uparrow | \uparrow) = N_+/D \text{ and}$$
$$q \equiv \mathbb{P}(\downarrow | \downarrow) = N_-/D \ ,$$

where $N_\pm = \exp\beta(J \pm b)$ and:

$$D = \exp(\beta J)\cosh(\beta b) + \sqrt{\exp(-2\beta J) + \exp(2\beta J)\sinh(\beta b)^2} \ ,$$

with $\beta = 1/(k_B T)$.

Calculating the ϵ-machine via the causal-state equivalence relation is straightforward. There are exactly two causal states; except when $p = 1 - q$ where we find only one causal state. The conclusion is that the two-state Markov chain process is minimally represented by the ϵ-machine in Fig. 3.1. Using Eq. (3.1), the statistical complexity is directly calculated as a function of p and q:

$$C_\mu = -\left(\frac{1-q}{2-p-q}\right)\log_2\left(\frac{1-q}{2-p-q}\right) - \left(\frac{1-p}{2-p-q}\right)\log_2\left(\frac{1-p}{2-p-q}\right) \ .$$

Figure 3.2 showed that C_μ is a monotonically increasing function of temperature T: $1 - C_\mu \propto T^{-2}$ at high T. In particular, for the three processes chosen at temperatures $T_\alpha < T_\gamma < T_\delta$:

$$C_\mu^\alpha < C_\mu^\gamma < C_\mu^\delta \ .$$

Recall that Fig. 3.4 demonstrated that the presence of ambiguity is robust: There exist parameter regions in which the ambiguity is stable against alternative quantum representations—alternatives that arguably could lead to different simplicity metrics.

To address how common ambiguity is, consider ambiguity in the analogous (nearest-neighbor, ferromagnetic) two-dimensional Ising system. To answer this question we need to come back and look closely at the important difference between C_μ and C_q. While C_q is a smooth function of ϵ-machine's transition probabilities, this is not generally true for C_μ. Consider Fig. 3.1 for p and q close to $p = 1/2$. In this case, we have a uniform distribution over causal states and consequently $C_\mu \simeq 1$. For the quantum-machine, using Eq. (3.5), two states are very close to each other meaning $\langle\sigma_1|\sigma_2\rangle \simeq 1$. The consequence is $C_q \simeq 0$. Now, lets look at the case where $p = q = 1/2$. The two causal states are not distinguishable and we only have one causal state and as a result $C_\mu = 0$. We also have $|\sigma_1\rangle = |\sigma_2\rangle$ which leads to $C_q = 0$. The lesson here is that C_q smoothly tracks how distinguishable states are, but C_μ tracks if causal states are exactly distinguished or not.

Recall that for the 1D Ising chain at high temperatures $p \neq q$; in particular, though they are both are close to $1/2$, they are never equal. The consequence is $C_q \simeq 0$ and $C_\mu = 1$. What about for the 2D Ising model? At the extreme $T = 0$ and for any nonzero value of external field, the ground state will be in uniform alignment with the field. This means that any random variable constructed from spin variables must have vanishing entropy. Lacking a complete computational mechanics of structure in two-dimensional patterns (though see Ref. [106, 124]), it is still clear that any analog of statistical complexity (and thereby C_q) will vanish at $T = 0$ for such uniform configurations.

At very high T, though, spins become increasingly uncorrelated and the configurational conditional probability distribution associated with each causal state approaches uniformity, *but as a set the distributions remain distinct*. That is, for any sufficiently high finite temperature, the system has some, perhaps weak, correlation that keeps the distributions from becoming identical. In other words, causal states in this regime remain probabilistically distinct. So, as with the 1D case, at very high temperature ($T \gg 1$, but $T \neq \infty$) $C_\mu(T)$ is not zero.

What can we say about C_q in this limit? For high $T \gg 1$ spin randomness makes the quantum states $\{|\eta\rangle\}$ (Eq. (3.2)) more and more indistinguishable. And so, their increasing overlaps $\langle \eta_i | \eta_j \rangle \to 1$, driving C_q to zero monotonically. The conclusion is that for the 2D Ising, at $T \approx 0$ and $T \gg 1$, we have the same qualitative picture for the simplicity measures as depicted in Fig. 3.2. This brief argument says that ambiguity exists in the 2D Ising spin model as well.

3.9.2 Ambiguity Robustness Theorem

First, we lay bare the mathematical argument and then we interpret it in terms of the physical setting of the main text.

Consider a set of objects S and two functions over the set $F_1 : S \to G$ and $F_2 : S \to G$. Space G consists of elements that can be compared as follows.

If there exists $s_1, s_2 \in S$, such that $F_1(s_1) > F_1(s_2)$ and $F_2(s_1) < F_2(s_2)$, then we say these functions are *ambiguous* over S.

We define three conditions for the set and functions.

Condition A The two functions map onto the whole space G: $F_1(S) = G$ and $F_2(S) = G$.

Condition B For all $g \in G$ there exists $x \in S$ such that $F_1(x) = F_2(x) = g$.

Condition C Assume $\preceq$ is a dense, total order on space G.

Theorem 1. *Given two functions F_1 and F_2 that map set S to space G and satisfy Conditions A, B, and C: No ambiguity implies that for all $x \in S$, $F_1(x) = F_2(x)$.*

Proof. *We prove the contrapositive by contradiction. Assume there exists $x \in S$ such that $F_1(x) \neq F_2(x)$. Without loss of generality, let $F_1(x) \succeq F_2(x)$. Since $\preceq$ is a dense total order on G, there is $g \in G$ such that $F_1(x) \succeq g \succeq F_2(x)$. By Condition B, there exists $y \in S$ such that $F_1(y) = F_2(y) = g$. Trivially then, $F_1(x) \succeq F_1(y)$ and $F_2(x) \preceq F_2(y)$. This demonstrates ambiguity and completes the proof.*

We can interpret this in the setting of stationary processes with measures C_μ and $\widetilde{C_q}$ and discuss the space of all possible quantum sizes. More specifically, consider the case $F_1 = C_\mu$ and $F_2 = \widetilde{C_q}$. We know that for any value $y \in \mathbb{R}$, there exists an ϵ-machine with

$C_\mu = \widetilde{C_q} = \mathbf{E} = y$. This satisfies the assumption. Then, our results say that if the world is not ambiguous, the two measures are equivalent. In other words, the quantum advantage $C_\mu/\widetilde{C_q}$ *requires* ambiguity.

Chapter 4

Memory Cost of Biased Sampling

4.1 Overview

We classify the rare events of structured, memoryful stochastic processes and use this to analyze sequential and parallel generators for these events. Given a stochastic process, we introduce a method to construct a new process whose typical realizations are a given process' rare events. This leads to an expression for the minimum memory required to generate rare events. We then show that the recently discovered classical-quantum ambiguity of simplicity also occurs when comparing the structure of process fluctuations.

4.2 Introduction

One of the most critical computations today is identifying the statistically extreme events exhibited by large-scale complex systems. Whether in the domains of geology, finance, or climate, or whether in natural or designed systems (earthquakes and hurricanes versus market crashes and internet route flapping), one can argue that this class of problem is rapidly coming to define our present scientific and technological era [125]. Success in understanding the origins and occurrence of extreme events will have a major impact on social infrastructure and its sustainability.

Large deviation theory [126–131] is a relatively new and key tool for analyzing a process' full range of statistical fluctuations. Presaged by Shannon-McMillman-Breiman type theory in communication theory [21, 132], the mathematical development of large

deviations was first pursued by Donsker and Varadhan [133]. In essence, it can be seen as a refinement of the Central Limit Theorem [134] or as a generalization of Einstein's fluctuation theory [135,136]. Today, large deviation theory enters into physics in many different circumstances [131]. One can also formulate statistical mechanics in the language of large deviation theory [131,137,138]. And, it appears in abstract dynamical systems under the rubric of the thermodynamic formalism [139].

The following analyzes the memory resources required to generate, and so study, extreme events in structured temporal stochastic processes. It extends large deviation theory in a constructive way that leads to exact calculations of the spectrum of fluctuations for processes generated by finite-state hidden Markov models. Fortunately, in this setting the generation and fluctuation problems can be simply stated. And so, we first give a suitably informal introduction to process generators and fluctuation theory, leaving technical results for later.

4.3 Markov Processes and Their Generators

A discrete-time, discrete-value *stochastic process* [3, 4] is the probability space $\mathcal{P} = \left\{\mathcal{A}^\infty, \Sigma, \mathbb{P}(\cdot)\right\}$. Here, $\mathbb{P}(\cdot)$ is the probability measure over the bi-infinite chain $X_{-\infty:\infty} = \ldots X_{-2}X_{-1}X_0X_1X_2\ldots$, where random variables X_i take values in a finite discrete alphabet $\mathcal{A}$ and Σ is the σ-algebra generated by the cylinder sets in $\mathcal{A}^\infty$. The following only considers ergodic stationary processes; that is, $\mathbb{P}(\cdot)$ is invariant under time translation—$\mathbb{P}(X_{i_1}X_{i_2}\cdots X_{i_m}) = \mathbb{P}(X_{i_1+n}X_{i_2+n}\cdots X_{i_m+n})$ for all n, m—and over successive realizations. A familiar important property of stochastic processes is *Markov order* [77]. This is the minimum history length R required by any generator to correctly generate the process. Specifically, R is the smallest integer such that:

$$\mathbb{P}(X_t|\ldots, X_{t-2}, X_{t-1}) = \mathbb{P}(X_t|X_{t-R}, \ldots, X_{t-2}, X_{t-1}) \ .$$

To keep matters uncomplicated, consider a process consisting of time series $\ldots 10010011\ldots$ of binary symbols. Having raw sequences in hand does represent the process' behaviors, but in and of themselves the sequences are not that useful. For example, how can we predict future symbols? What mechanisms drive the process' behaviors? Much more helpful in

answering such questions is an algorithm that can produce the process' sequences. And, a good one can be used to simulate the process—generating example sequences, perhaps not even in the original data, but statistically similar—that allow one to predict future sequences, gain insight into the process' internal mechanisms, and estimate statistical properties.

Note that in most cases representing a process by specifying the probability measure $\mathbb{P}(\cdot)$ is impossible due to the infinite number of possible sequences. So, how should we represent processes? Is there a more compact way than specifying in-full the probability measure on the sequence sigma algebra? In a rather direct sense, *Markov chains* and *hidden Markov models* provide constructive answers. The quality of those answers depends, of course, on how useful these representations are. We now fill in their technical details, so that we can work with them.

Markov chains (MCs) [76,77] and hidden Markov models (HMMs) [4–6] are widely-used algorithms for generating stochastic processes. Both consist of a set $\boldsymbol{\mathcal{S}}$ of states and a set of state-transition probabilities. Formally, both MCs and HMMs are specified by a tuple $\left\{\boldsymbol{\mathcal{S}}, \mathcal{A}, \{T^{(x)}, x \in \mathcal{A}\}\right\}$. In this, $\boldsymbol{\mathcal{S}}$ is a finite set of states, $\mathcal{A}$ is a finite alphabet, and $\{T^{(x)}, x \in \mathcal{A}\}$ is a set of $|\boldsymbol{\mathcal{S}}| \times |\boldsymbol{\mathcal{S}}|$ substochastic symbol-labeled transition matrices whose sum $T = \sum_{x \in \mathcal{A}} T^{(x)}$ is an stochastic matrix. In MCs states are past words, whereas in HMMs states and words are distinct. Hence, their states are hidden—not directly observed.

Consider an example HMM where $\boldsymbol{\mathcal{S}} = \{A, B\}$, $\mathcal{A} = \{0, 1\}$, $T^{(0)} = \begin{bmatrix} p & 0 \\ 0 & 0 \end{bmatrix}$, and $T^{(1)} = \begin{bmatrix} 0 & 1-p \\ 1 & 0 \end{bmatrix}$. An HMM such as this is graphically depicted via its state-transition diagram—a directed graph with labeled edges. $\boldsymbol{\mathcal{S}}$ is the set of graph nodes and the edge from node i to j is labeled by $p|x$ corresponding to the HMM transition with probability $p = T^{(x)}_{ij}$ that goes from state i to j and generates symbol x. Fig. 4.1 shows the state-transition diagram for a two-state HMM that generates a process called the binary-symbol *Even Process* [102].

The Even Process highlights why HMMs are such useful algorithms. Since MC states

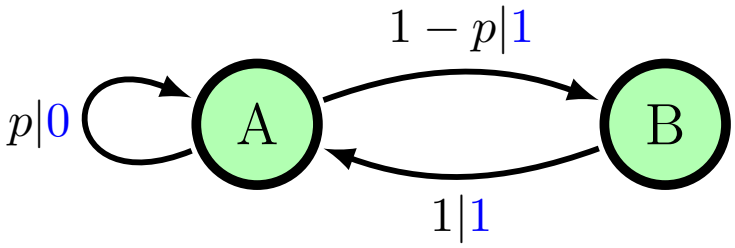


Figure 4.1: State-transition diagram for the hidden Markov generator of the Even Process, which consists of random binary sequences with an even number of 1s separated by arbitrary-length blocks of 0s.

are constrained to be individual past, HMMs can be arbitrarily more compact than MCs for the same process. In this case, the Even Process is an infinite Markov order process since its current state can depend on arbitrarily long histories. (If only 1s have been observed, it can be in either state A or state B.) Said in terms of algorithmic complexity, the MC representing the Even Process requires an infinite number of Markov states, each associated with a history 1^k0, $k = 0, 1, 2, \ldots$. In contrast, as the figure makes plain, the Even Process' HMM takes only two states. This is why HMMs are preferred algorithms compared to MCs when it comes to generating processes.

When using HMMs as process generators we can restrict attention to those that are *unifilar*: the current state and next symbol uniquely determine the next state. Unifilar HMMs are important since they are perfect predictors of their process. (The same is not generally true of a process' nonunifilar HMM generators. We return to the important, but subtle distinction between prediction and generation using HMMs at the end.) For any given process there is an infinite number of unifilar HMM generators; so the restriction imposes no loss of representational generality. Given all of the alternative HMMs, though, which do we choose?

4.4 Optimal Serial and Parallel Generators

Let's say Alice wants to generate the Even Process. The previous remarks indicate that she should not use an MC algorithm since it has infinite states and, as a result, needs an infinite amount of memory to generate the process. Ands so, she uses an HMM algorithm, which is finite. To do this, she writes a computer program: If the current state is A, with

probability p the program emits symbol 0 and stays at state A and with probability $1-p$ it emits symbol 1 and goes to state B. However, if the current state is B, it generates symbol 1 and goes to the state A. The program continues in this fashion, again and again, and in the long run generates a realization of the Even Process. Moreover, if Alice chooses to start in A or B using the asymptotic state probability distribution π, then the resulting realization is stationary.

Imagine that a long time has passed and the HMM is in state A. Alice decides to stop the program for now and return tomorrow to continue generating the same realization. She must make a decision, does she use the realization generated today or start all over again tomorrow? Not wanting to waste the effort already invested, she decides to use today's realization tomorrow and simply concatenate newly generated symbols.

The next day, though, can she randomly pick a state and continue generating? The answer is no. If she randomly picks state B, then there is a chance that after concatenating the old and new realizations together, the sequence has odd number of 1s between two 0s. However, she knows that the Even Process never generates such subsequences. Thus, if she wants to use today's realization tomorrow then, she must record the HMM's current state and continue generating from that state tomorrow.[1]

Information theory [21] tells us that to record the current state Alice needs $\log_2 |\boldsymbol{\mathcal{S}}|$ bits of memory. This is the cost of *sequential generation*. And, it gives a quantitative way to compare algorithms across the infinite number of alternatives. If Alice wants to use less memory, she selects the HMM with the minimum number $|\boldsymbol{\mathcal{S}}|$ of states. Which representation achieves this?

Before answering, let's contrast another scenario, that for *simultaneous generation*. Now, Alice wants to generate $N \gg 1$ realizations for a given process simultaneously, but insists that the individual sequences be statistically independent. The latter means that she cannot simply generate a single realization and copy it N times. At first blush, it seems that she needs $N \log_2 |\boldsymbol{\mathcal{S}}|$ bits of memory. According to Shannon's source coding

[1] The time period over which Alice pauses generation can be set to any duration—an hour, a minute, or a second. In particular, the period can be that required to generate a single symbol. In this case, after every symbol emitted Alice must know in what state the generator is. In short, Alice needs to remember the current state during generation.

theorem [21,140], though, she can compress the sequence information and, for large N, she needs only $N\,\mathrm{H}[\boldsymbol{\mathcal{S}}] \leq N \log_2 |\boldsymbol{\mathcal{S}}|$ bits of memory, where $\mathrm{H}[\boldsymbol{\mathcal{S}}] = -\sum_{\sigma \in \boldsymbol{\mathcal{S}}} \pi(\sigma) \log_2 \pi(\sigma)$ is the Shannon entropy of the stationary probability distribution $\pi(.)$ over the HMM's states. That is, on average Alice needs $\mathrm{H}[\boldsymbol{\mathcal{S}}]$ bits of memory to generate each realization. So, if Alice wants to use less memory, she selects the process HMM with the minimum $\mathrm{H}[\boldsymbol{\mathcal{S}}]$ in the set of unifilar HMMs. Again, which representation achieves this?

Crutchfield and Young [56] showed that over all unifilar HMMs that generate a given process, there is a unique HMM with the minimum number of states. Surprisingly, this same HMM is also the one with the minimum entropy over its states. It is now known as the *ϵ-machine* [22,141] and its state entropy is the process' *statistical complexity* C_μ [22,56]. The consequence is that, for a given stochastic process, the minimum memory required for any unifilar HMM to sequentially generate it is $\log_2 |\boldsymbol{\mathcal{S}}_\epsilon|$ bits, where $\boldsymbol{\mathcal{S}}_\epsilon$ is the set of states in the process' ϵ-machine. And, for simultaneous generation the average minimum required memory for each realization is C_μ.

Today, C_μ is often used as a measure of structural complexity for stochastic processes, from stochastic resonance [142] to hydrodynamic flows [143], atmospheric turbulence [144], geomagnetic volatility [145], and single-molecule dynamics [86,87,146]. In short, we use ϵ-machines and C_μ to measure the memory inherent in a stochastic process. And, by the preceding argument we now know how they determine the memory required for sequential and parallel generation.

4.5 Typical and Atypical Behaviors

So far, the discussion implicitly assumed that models captured a process' typically observed behaviors. However, most stochastic processes exhibit statistical fluctuations and so occasionally generate atypical, statistically extreme behaviors. Now, we turn to define what we mean by typical and atypical behaviors. Once done, we finally state our problem: How much memory is needed to generate a process' atypical behaviors.

So, what does it mean that a process exhibits statistical fluctuations? Let's say Alice has a biased coin, meaning that when she flips it, the probability p of seeing heads is

greater than one half. Alice now flips the coin $n \gg 1$ times and sees k heads. The Strong Law of Large Numbers [147] guarantees that for large n, the ratio k/n almost surely converges to p:

$$\mathbb{P}\left(\lim_{n\to\infty} \frac{k}{n} = p\right) = 1 \ .$$

Informally, for large n the *typical sequence* has close to p percent Heads. This does not mean that Alice never sees long runs of all Heads or all Tails, for example. It simply means that the latter are rare events.

We now show that a process' typically observed realizations are those sequences in its so-called typical set. Consider a given process and let $\mathcal{A}^n$ denote the set of length-n sequences. Then, for an arbitrary $\epsilon > 0$ the process' *typical set* [21,148,149] is:

$$A_\epsilon^n = \{w : 2^{-n(h_\mu+\epsilon)} \leq \mathbb{P}(w) \leq 2^{-n(h_\mu-\epsilon)}, w \in \mathcal{A}^n\}, \tag{4.1}$$

where h_μ is the process' *metric entropy* (Shannon entropy rate) [150]:

$$h_\mu(\mathcal{P}) = -\lim_{n\to\infty} \frac{1}{n} \sum_{w\in\mathcal{A}^n} \mathbb{P}(w) \log_2 \mathbb{P}(w) \ .$$

According to the *Shannon-McMillan-Breiman theorem* [140,151,152], for a given $\epsilon \ll 1$ and sufficiently large n:

$$\mathbb{P}(w \notin A_\epsilon^n, w \in \mathcal{A}^n) \leq \epsilon \ . \tag{4.2}$$

There are two important lessons here. First, coming from Eq. (5.1), all sequences in the typical set have approximately the same probability. Second, coming from Eq. (5.2), for large n the probability of sequences falling outside the typical set is close to zero—they are rare.

One consequence is that sequences generated by a stationary ergodic process fall into one of three partitions; see Fig. 5.3. The first contains those that are never generated by a process—sequences with zero probability. (For example, the Even Process cannot generate realizations containing a subsequence in $\{01^{2k+1}0\}$, $k = 0, 1, 2, \ldots$—those with an odd number of 1s between 0s.) These are the *forbidden sequences*. The second partition

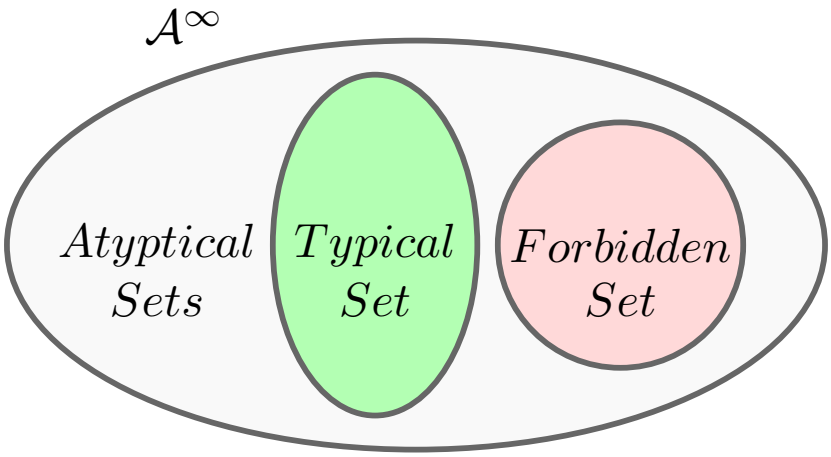


Figure 4.2: For a given process, the space $\mathcal{A}^\infty$ of its realizations is partitioned into forbidden sequences, sequences in the typical set, and sequences in atypical sets.

consists of those in the typical set—the set with probability close to one, as in Eq. (5.1). And, the last contains sequences in a family of atypical sets—realizations that are rare to different degrees. We now refine this classification.

Mirroring the familiar *Boltzmann weight* in statistical physics [153], in the $n \to \infty$ limit, we define the subsets $\Lambda_U^{\mathcal{P}} \subset \mathcal{A}^\infty$ for a process $\mathcal{P}$ as:

$$\Lambda_{U,n}^{\mathcal{P}} = \left\{ w : -\frac{\log_2 \mathbb{P}(w)}{n} = U, \ w \in \mathcal{A}^n \right\}$$
$$\Lambda_U^{\mathcal{P}} = \lim_{n \to \infty} \Lambda_{U,n}^{\mathcal{P}} \ . \tag{4.3}$$

In effect, this partitions $\mathcal{A}^\infty$ into subsets $\Lambda_U^{\mathcal{P}}$ in which all $w \in \Lambda_U^{\mathcal{P}}$ have the same probability decay rate U. Physics vernacular would speak of the sequences having the same *energy density* U.[2] Figure 4.3 depicts these subsets as "bubbles" of equal energy. (Though, to be clear about their "shape", these subsets are isomorphic to Cantor sets.) The definition guarantees that any bi-infinite sequence $\mathcal{P}$ generates belongs to one of these sets. Equation (5.1) says the typical set is that bubble with energy equal to the process' entropy rate: $U = h_\mu$. All the other bubbles contain rare events.

When Alice uses a process' HMM to generate realizations, what she does is generate sequences in the typical set with probability close to one and, rarely, atypical sequences. Imagine, though, that Alice is interested in a particular class of rare sequences, those

[2] U, considered as a random variable, is sometimes called a *self process* [129].

in a different isoenergy bubble; say, those with energy U in the set $\Lambda_U^{\mathcal{P}}$. How can Alice efficiently generate these rare sequences? We now show that she can find a new process $\mathcal{P}^U$ whose typical set is $\Lambda_U^{\mathcal{P}}$.

4.6 Generating Rare Events

To do this, we return to considering HMMs for a given process. With suitable HMMs and a precise definition of a process' atypical sequences we can now ask, How much memory is required to generate them? How does this compare to the memory required to generate typical behaviors?

Given a process $\mathcal{P}$ and its ϵ-machine $M(\mathcal{P})$, How do we construct an ϵ-machine $M(\mathcal{P}^U)$ that generates $\mathcal{P}$'s atypical sequences at some energy $U \neq h_\mu$? Here, we answer this question by constructing a map $\mathcal{B}_\beta : \mathcal{P} \to \mathcal{P}_\beta$ from the original $\mathcal{P}$ to a new process $\mathcal{P}_\beta$. The latter is parametrized by $\beta \in \mathbb{R}/\{0\}$ which indexes the atypical set of interest. Both processes $\mathcal{P} = \left\{\mathcal{A}^\infty, \Sigma, \mathbb{P}(\cdot)\right\}$ and $\mathcal{P}_\beta = \left\{\mathcal{A}^\infty, \Sigma, \mathbb{P}_\beta(\cdot)\right\}$ are defined on the same measurable sequence space. The measures differ, but their supports (allowed sequences) are the same. We refer to $\mathcal{B}_\beta$ as the *β-map*.

Assume we are given $M(\mathcal{P}) = \left\{\boldsymbol{\mathcal{S}}, \mathcal{A}, \{T^{(x)}, x \in \mathcal{A}\}\right\}$. We will now show that for every probability decay rate or energy U, there exists a particular β such that $M(\mathcal{P}_\beta)$ typically generates the words in $\Lambda_{U,n}^{\mathcal{P}}$ for large n. The β-map which establishes this is calculated by a construction that relates $M(\mathcal{P})$ to $M(\mathcal{P}_\beta) = \left\{\boldsymbol{\mathcal{S}}, \mathcal{A}, \{\mathbf{S}_\beta^{(x)}, x \in \mathcal{A}\}\right\}$—the HMM that generates $\mathcal{P}_\beta$:

1. For each $x \in \mathcal{A}$, construct a new matrix $\mathbf{T}_\beta^{(x)}$ for which $\left(\mathbf{T}_\beta^{(x)}\right)_{ij} = \left(\mathbf{T}^{(x)}\right)_{ij}^\beta$.
2. Construct a new matrix $\mathbf{T}_\beta = \sum_{x \in \mathcal{A}} T_\beta^{(x)}$.
3. Calculate $\mathbf{T}_\beta$'s maximum eigenvalue $\widehat{\lambda}_\beta$ and corresponding right eigenvector $\widehat{\mathbf{r}}_\beta$.
4. For each $x \in \mathcal{A}$, construct new matrices $\mathbf{S}_\beta^{(x)}$ for which:

$$\left(\mathbf{S}_\beta^{(x)}\right)_{ij} = \frac{\left(\mathbf{T}_\beta^{(x)}\right)_{ij} (\widehat{\mathbf{r}}_\beta)_j}{\widehat{\lambda}_\beta (\widehat{\mathbf{r}}_\beta)_i} \ . \tag{4.4}$$

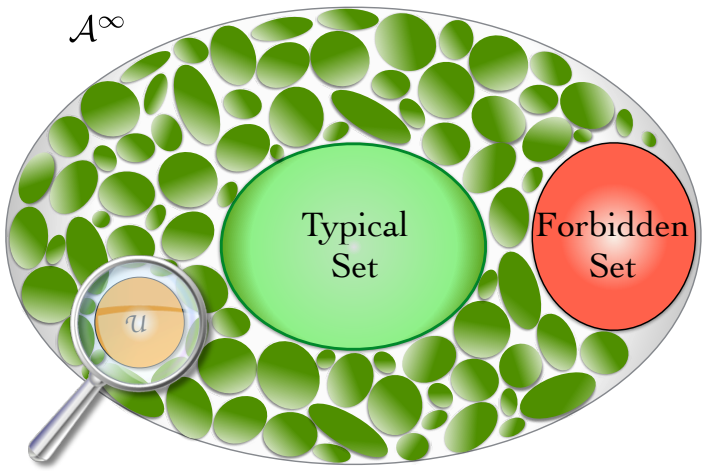


Figure 4.3: $\mathcal{A}^\infty$ partitioned into Λ_Us—isoenergy or equal probability-decay-rate bubbles—in which all sequences in the same Λ_U have the same energy U. The typical set is one such bubble with energy equal to metric entropy: $U = h_\mu$. Another important partition is that of the forbidden sequences, in which all sequences have zero probability. The forbidden set can also be interpreted as the subset of sequences with infinite energy.

Theorem 2. *For the new process $\mathcal{P}_\beta$ in the limit $n \to \infty$ the probability of the set $\Lambda^{\mathcal{P}}_{U,n}$ converges to one $\lim_{n\to\infty} \mathbb{P}_\beta(\Lambda^{\mathcal{P}}_{U,n}) = 1$ where:*

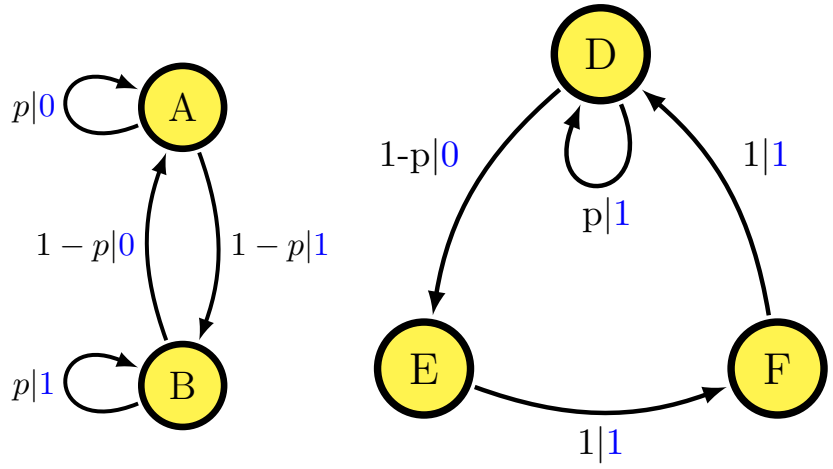
$$U = \beta^{-1}\big(h_\mu(\mathcal{P}_\beta) - \log_2 \widehat{\lambda}_\beta\big) \ . \tag{4.5}$$

Also, in the same limit the process $\mathcal{P}_\beta$ assigns equal energies over all the members of the set $\Lambda^{\mathcal{P}}_{U,n}$.

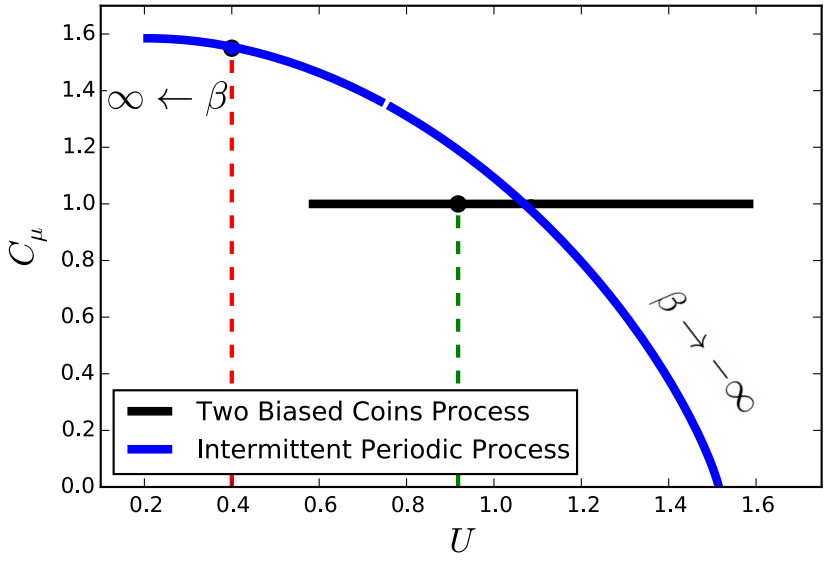
Proof. *See the appendix.*

As a result, for large n the process $\mathcal{P}_\beta$ typically generates the set $\Lambda^{\mathcal{P}}_{U,n}$, where $U = \beta^{-1}\big(h_\mu(\mathcal{P}_\beta) - \log_2 \widehat{\lambda}_\beta\big)$. And so, there is a one-to-one relationship between β and U and we can denote the process $\mathcal{P}_\beta$ by $\mathcal{P}^U$. More formally, every word in $\Lambda^{\mathcal{P}}_U$ with probability measure one is in the typical set of process $\mathcal{P}_\beta$.

This says that changing β controls which class of rare events we focus on. Informally, the β-map acts like a magnifier (Fig. 4.3) by enhancing particular isoenergy bubbles. That is, changing β moves the magnifier from one bubble to another. The β-map construction guarantees that the HMMs $M(\mathcal{P})$ and $M(\mathcal{P}_\beta)$ have the same states and transition topology:



(a) (b)



(c)

Figure 4.4: a Two-Biased Coins (TBC) Process ϵ-machine generator. b Intermittent Periodic Process (IPP) ϵ-machine generator. c Statistical complexity C_μ versus energy U (or fluctuation class) for each, along with the energies U^* at which their typical sets are found (vertical dashed lines).

$\left(\mathbf{T}_\beta^{(x)}\right)_{ij} \neq 0 \iff \left(\mathbf{S}_\beta^{(x)}\right)_{ij} \neq 0$. The only difference is in their transition probabilities. Thus, $M(\mathcal{P}_\beta)$ is also a unifilar HMM, but not necessarily an ϵ-machine, since the latter requires a minimal set of states. Minimality is not guaranteed by the β-map. Typically, though, $M(\mathcal{P}_\beta)$ is an ϵ-machine and there is only a finite number of βs for which it is not.

(More detailed development along these lines will appear in a sequel.)

Historically, a similar map was found for the first time in 1961 by Miller [154], but only for Markov order-one processes. In the setting of continuous-time first-order Markov evolution a similar map was introduced by Refs. [155,156] (*s-ensemble*), by Ref. [157] (*biased ensemble*), and Ref. [158,159] (*exponential tilting*). In these settings $\mathcal{P}_\beta$ is sometimes called an *auxiliary process* [157].

The β-map for unifilar HMMs and, consequently, for finite- or infinite-order discrete-time discrete-value Markov processes was introduced for the first time in 1993 [128]. A proof was not provided, which we remedy here, explaining why this β-map works so generally. There $\mathcal{P}_\beta$ was called the *twisted distribution*.

4.7 Memory Spectra

For an arbitrary stochastic process $\mathcal{P}$, using its ϵ-machine the last section presented a method to construct a (unifilar) generator whose typical set is the process $\mathcal{P}^U$—the rare events of the original $\mathcal{P}$. Now, we determine the minimum memory required to generate $\mathcal{P}^U$. Recalling the earlier coding-theoretic arguments, this is rather straightforward to answer. The minimum memory to generate $\mathcal{P}^U$ is determined by the size of its ϵ-machine. (As noted, this is the size of $M(\mathcal{P}_U)$ except for finite number of U.)

And so, except for a finite number of rare-event classes, to sequentially generate sequences in a given rare class, one requires the same memory—the number $|\boldsymbol{\mathcal{S}}|$ of states—as that to generate the original process. This is our first result on the minimum Markov memory for a process' rare events.

The story differs markedly, however, for simultaneous generation. The minimum required memory for simultaneous generation of $\mathcal{P}^U$ is $C_\mu(\mathcal{P}^U)$, putting the earlier coding argument together with last section's calculations. More to the point, this is generally not equal to $C_\mu(\mathcal{P})$. To better appreciate this result, let us examine three examples.

First, consider the Two-Biased Coins (TBC) Process with $p = 1/3$, whose ϵ-machine is shown in Fig. 4.4a(top left). To generate its realizations one flips a biased coin repeatedly. At first, label Heads a 0 and Tails a 1. After flipping, switch the labels and call a Head

1 and Tail 0. A TBC process sequence comes from repeating these steps endlessly. As Fig. 4.4a makes clear, there is a symmetry in the process. In the stationary distribution π, state A has probability half, as does state B, and this is independent of p. This gives $C_\mu(\mathcal{P}) = 1$ bit. Recalling the β-map construction, we see that changing β does not change the ϵ-machine topology. All that changes is p. This means, in turn, that the symmetry in states remains and $C_\mu(\mathcal{P}^U) = 1$ is constant over allowed Us (or βs); $C_\mu(U)$ versus U is the horizontal line shown in Fig. 4.4c.

What energies are allowed? The TBC Process has a finite energy range: $U \in [\approx 0.586, \approx 1.584]$. From Eq. (4.3) we see that the maximum $U_{\max}$ corresponds to the bubble with the rarest sequences that can be generated. Conversely, $U_{\min}$ corresponds to the bubble with the most probable. The energy extremes delimit the domain of the $C_\mu(\mathcal{P}^U)$ curves in Fig. 4.4c. In addition, the U associated with $\mathcal{P}$'s typical set is marked in the figure with a dashed (green) vertical line near $U \approx 0.9183$.

The difference between the typical set and that with $U_{\min}$ is important to appreciate. The typical set is that *set* of sequences with probability close to one and with energy $U = h_\mu$. The latter is generally different from $U_{\min}$. That is, typical sequences are not necessarily the most probable sequences, considered individually, but rather they belong to the most probable subset—the typical set.

As a result of this analysis for this example, one 1 bit of memory is uniformly required for generating the TBC Process' events, rare or not and independent of which class of rare events we examine.

Second, this is not the general case, since $C_\mu(\mathcal{P}^U)$ can be a nonconstant function of U, as we now show. Consider the Intermittent Periodic Process (IPP) with $p = 0.35$; its ϵ-machine is given in Fig. 4.4b(top right). It gets its name since when $p = 0$, it periodically emits the subsequence 101 and when $p > 0$, it randomly inserts 1s. Using the β-map and Thm. 3 we can find the processes $\mathcal{P}^U$ and calculate C_μ. Fig. 4.4c shows how their $C_\mu(\mathcal{P}^U)$ depends on U. The IPP is similar to the TBC Process in that it also has a finite energy range; IPP energies $U \in [\approx 0.207, \approx 1.515]$. It turns out that for any process with a finite ϵ-machine the allowed energy range is also finite. In addition, the U associated with $\mathcal{P}$'s

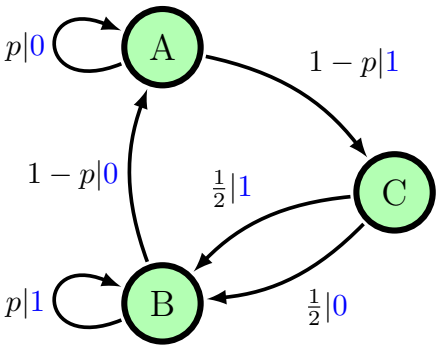
typical set is marked in the figure with a dashed (red) vertical line near $U \approx 0.406$.

Thus, IPP's $C_\mu(\mathcal{P}^U)$ is a nontrivial function of U. Practically, this means that generating various rare-sequence classes requires less memory than for other classes. For example, for events with $U_{\max}$—$p = 1$ and $\beta \to -\infty$—ones needs no memory, since the class of maximum energy has only one sequence—the all-1s sequence. This can be generated by an IID process that emits only 1s. Generally, due to its IID character we do not need to remember or store the process' current state. In other words, the ϵ-machine $M(\mathcal{P}^U)$ that generates this class only has one state and so $C_\mu = 0$ bits there. For $U_{\min}$, occurring at $p = 0$ and $\beta \to \infty$, there are three "ground state" sequences—the three shifts of $\ldots 101101 \ldots$ and three equally probable states. Thus, $C_\mu(U_{\min}) = \log_2 3 \approx 1.585$ bits are necessary for generation.

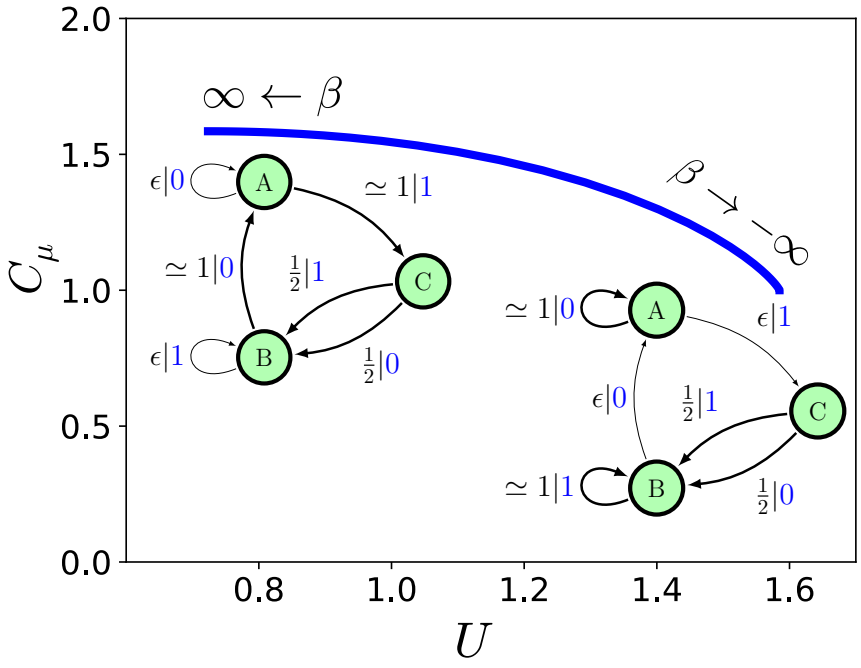
Third and finally, for a more complex example consider the process generated by the ϵ-machine with $p = 1/3$ given in Fig. 4.5a(top). Using the β-map and Thm. 3 we again find the processes $\mathcal{P}^U$ and calculate their C_μ, as shown in Fig. 4.5b(bottom). The difference between this process and IPP is that at no inverse temperature β do we have an IID process $\mathcal{P}_\beta$. As a consequence $C_\mu(\mathcal{P}^U)$ is nonzero for all allowed U.

The insets in Fig. 4.5b(bottom) highlight the details of the process' ϵ-machines for two limits of β. In the limit $\beta \to \infty$ the probability of B's self-transition vanishes and the probability of transiting from state B to A goes to one. Similarly, the probability of A's self-transition vanishes and the A-to-C transition probability goes to one. As a consequence, as shown in Fig. 4.5b, the extreme process generates 0 then 1, then flips a coin to decide the outcome and then repeats the same steps again and again. The physical interpretation is that this limit captures the process' "ground states" and they have positive entropy density and memory.

In the complementary limit $\beta \to -\infty$, an interesting property emerges. The process breaks into two distinct subprocesses that link to each other only arbitrarily weakly. The first process consists of state B with a deterministic self-transition that generates 1s. And, the second subprocess consists of state A with a deterministic self-transition that and generates 0s. In other words, the process has two phases that rarely switch



(a)



(b)

Figure 4.5: a Process ϵ-machine generator. b Statistical complexity C_μ versus energy U for the ϵ-machine generator. Insets (bottom) display ϵ-machines for the processes generating the fluctuation extremes at $\beta \to \infty$ and $\beta \to -\infty$.

between themselves. As a result, over moderate durations the process exhibits nonergodic behavior. We note that this has profound effects on predictability: substantial resources are required for predicting nonergodic processes [160], despite their requiring finite resources for generation.

4.8 Conclusions

To generate the rare behaviors of a stochastic process one can wait, if one wants, for exponentially long times for them to occur. Here, we introduced an alternative to rare-event generation from large deviation theory and its predecessors. Given a process, we first classified its events into those that are forbidden, typical, and atypical. And, then we refined the atypical class. For any chosen rare class we introduced an algorithm that constructs a new process, and its unifilar HMM, that typically generates those rare events. Appealing to the optimality of computational mechanics' ϵ-machines then allowed us to analyze the minimal memory costs of implementing rare-event generators. Depending on the goal—producing a single correct sample (sequential generation) or a large number of correct of samples (simultaneous generation) from the rare class of interest—memory cost differs. We studied both costs. Taken together the three examples analyzed give a complete survey of applying the method and how memory costs vary across classes of rare events.

There are two main types of algorithms for generating stochastic processes: Monte Carlo versus finite-state machine algorithms. Monte Carlo algorithms are appropriate if the process can be written as a probability distribution generated by a Hamiltonian system and if what we are interested are macroscopic statistics. For a given process, finding a compact Hamiltonian generator can be challenging. In addition, to generate long realizations using Monte Carlo algorithms one needs corrspondingly long initial data. This data, which changes during the algorithm, must be stored by the algorithm. And so, this approach can be memory intensive. These limitations do not exist for finite-state machine algorithms.

The introduction emphasized that we only focused on unifilar HMMs as process generators and then we constructed the minimal unifilar generator for a given class of rare events. The unifilar condition is necessary when using a process' past behavior to optimally predict its future [12]. However, one may not be interested in prediction, only generation for which unifilarity is not required. While removing unifilarity expands the space of HMMs, it greatly complicates finding minimal generators. For one, *nonunifilar*

HMMs can be more memory efficient than unifilar HMMs for a given process [4,57,161]. For another, constructing a minimal nonunifilar HMM for a general process is still an open and hard question [58,59,81].

The required memory $C_\mu(\mathcal{P})$ for (unifilarly) generating realizations of a given process $\mathcal{P}$ has been used as a measure of structural complexity for over two decades. It places a total order over stochastic-process space, ranking processes by the difficulty to generate them. The theorem introduced here extends the measure $C_\mu(\mathcal{P})$ to the full memory spectrum $C_\mu(\mathcal{P}^U)$ to generate fluctuations.

As one consequence, this structural accounting introduces the new phenomenon of the *ambiguity of simplicity* [15] to the domain of fluctuation theory. Say that process A is simpler than process B, since it requires less memory to generate: $C_\mu(A) < C_\mu(B)$. However, if instead we are interested in the rarest events at U, we showed that it is possible that A is more complex than process B since it requires more memory for that event class: $C_\mu(A^U) > C_\mu(B^U)$. As Ref. [15] notes, this fundamental ambiguity flies in the face of appeals to simplicity via Occam's Razor and practically impacts employing statistical model selection as it relies on a total order of model complexity.

The same fluctuation theory has recently been used to identify fluctuations in macroscopic thermodynamic functioning in Maxwellian Demons [162]. Moreover, the method can be applied to many stochastic systems to explore their rare behaviors, from natural processes observed in fluid turbulence [163], physiology [164,165], surface science [166,167], meteorological processes [168], to designed systems found in finance [169,170], and traffic [171,172]. It gives a full description of a process, from its typical to its rare behaviors. And, it determines how difficult it is to simulate a process' rare events.

Finally, there is another potentially important application domain. The rapid progress in quantum computation and information suggest that, perhaps soon even, one will be able to generate processes, both classical and quantum, using programmable quantum systems. The equivalent memory C_q for the simultaneous quantum simulation of processes also has already been introduced [8,10,12–14]. And so, a sequel will analyze quantum memory fluctuation spectra $C_q(U)$ and how they differ from the classical spectra introduced here.

4.9 Appendix

This appendix establishes the main theorem via a single lemma relying on a process' cryptic order.

Cryptic order is a recently introduced topological property of stochastic processes [173] that is bounded by, but is rather different in motivation from, the more familiar Markov order [174]. Formally, given a process' ϵ-machine, its *cryptic order* is $K = \inf\left\{l : \mathrm{H}[\mathcal{S}_l | X_0 X_1 \cdots] = 0,\ l \in \mathbb{Z}\right\}$. Informally, this means that if we observe an infinite length realization, we can be certain about in which state the ϵ-machine is in after the K^{th} symbol [175].

Lemma 1. *For any given process with finite states and cryptic order, for every U and $\beta \in \mathbb{R}/0$ we have:*

$$\Lambda_U^{\mathcal{P}} = \Lambda^{\mathcal{P}_\beta}_{\beta U - \log_2 \widehat{\lambda}_\beta} \ .$$

Proof. *Consider an arbitrary word $w = x_0 x_1 \ldots x_{n-1} \in \mathcal{A}^n$ generated by process $\mathcal{P}$ where $n \gg 1$. Since the ϵ-machine is unifilar, immediately after choosing the initial state, all the successor states are uniquely determined. Using this, we can decompose w to two parts: The first part w_K is the first K symbols and the second part is w's remainder. Knowing w, the state σ_K and all successor states following $\sigma_{K+1}, \sigma_{K+2}, \ldots$ are uniquely determined. As a consequence, the probability of process $\mathcal{P}$ generating w can be written as:*

$$\mathbb{P}(w) = \mathbb{P}(w_K) \prod_{i=K}^{n-1} \left(\boldsymbol{T}^{(x_i)} \right)_{\sigma_i \sigma_{i+1}} \ .$$

We can adapt the energy definition in Eq. (4.3) to finite-length sequences. Then, w's energy is:

$$\begin{aligned} \mathcal{U}(w) &= -\frac{\log_2 \mathbb{P}(w)}{n} \\ &= -\frac{\log_2 \mathbb{P}(w_K)}{n} - \frac{\log_2 \left(\prod_{i=K}^{n-1} \left(\boldsymbol{T}^{(x_i)} \right)_{\sigma_i \sigma_{i+1}} \right)}{n} \ . \end{aligned}$$

Now, consider the same word, but this time generated by the ϵ-machine $M(\mathcal{P}_\beta)$. Then, the probability of generating w is:

$$\begin{aligned}
\mathbb{P}_\beta(w) &= \mathbb{P}_\beta(w_K) \prod_{i=K}^{n-1} \left(\boldsymbol{S}_\beta^{(x_i)} \right)_{\sigma_i \sigma_{i+1}} \\
&= \mathbb{P}_\beta(w_K) \prod_{i=K}^{n-1} \frac{\left(\boldsymbol{T}_\beta^{(x_i)} \right)_{\sigma_i \sigma_{i+1}} (\widehat{\mathbf{r}}_\beta)_{\sigma_{i+1}}}{\widehat{\lambda}_\beta (\widehat{\mathbf{r}}_\beta)_{\sigma_i}} \\
&= \mathbb{P}_\beta(w_K) \frac{(\widehat{\mathbf{r}}_\beta)_{\sigma_n}}{(\widehat{\mathbf{r}}_\beta)_{\sigma_K}} \left(\widehat{\lambda}_\beta \right)^{n-K} \prod_{i=K}^{n-1} \left(\boldsymbol{T}_\beta^{(x_i)} \right)_{\sigma_i \sigma_{i+1}} \\
&= \mathbb{P}_\beta(w_K) \frac{(\widehat{\mathbf{r}}_\beta)_{\sigma_n}}{(\widehat{\mathbf{r}}_\beta)_{\sigma_K}} \left(\widehat{\lambda}_\beta \right)^{n-K} \left(\prod_{i=K}^{n-1} \left(\boldsymbol{T}^{(x_i)} \right)_{\sigma_i \sigma_{i+1}} \right)^\beta .
\end{aligned}$$

The new energy for the same word is:

$$\begin{aligned}
\mathcal{U}_\beta(w) &= -\frac{\log_2 \mathbb{P}(w)}{n} \\
&= -\frac{\log_2 \left(\mathbb{P}_\beta(w_K) \frac{(\widehat{\mathbf{r}}_\beta)_{\sigma_n}}{(\widehat{\mathbf{r}}_\beta)_{\sigma_K}} \right)}{n} - \frac{n-K}{n} \log_2 \widehat{\lambda}_\beta \\
&\quad - \beta \frac{\log_2 \left(\prod_{i=K}^{n-1} \left(\boldsymbol{T}^{(x_i)} \right)_{\sigma_i \sigma_{i+1}} \right)}{n} .
\end{aligned}$$

In the limit of large n the first terms in $\mathcal{U}(w)$ and $\mathcal{U}_\beta(w)$ vanish and we have $\mathcal{U}_\beta(w) = \beta \mathcal{U}(w) - \log_2 \widehat{\lambda}_\beta$. Thus, for any two long sequences $w_1, w_2 \in \mathcal{A}^n$, if $\mathcal{U}(w_1) = \mathcal{U}(w_2)$, then $\mathcal{U}_\beta(w_1) = \mathcal{U}_\beta(w_2)$. And, the partitions induced by Eq. (4.3) are invariant under the β-map. In other words, the energy of an arbitrary bubble after β-mapping changes from U to U_β, where:

$$U_\beta = \beta U - \log_2 \widehat{\lambda}_\beta \ .$$

This completes the lemma's proof.

This demonstrates how the β-map changes bubble energy: $U \to \beta U - \log_2 \widehat{\lambda}_\beta$. So, now we ask for the bubble (and its energy) that maps to the typical set of the new process $\mathcal{P}_\beta$. That is, we use the β-map to find the class $\Lambda_U^{\mathcal{P}}$ of rare sequences typically generated by $M(\mathcal{P}_\beta)$.

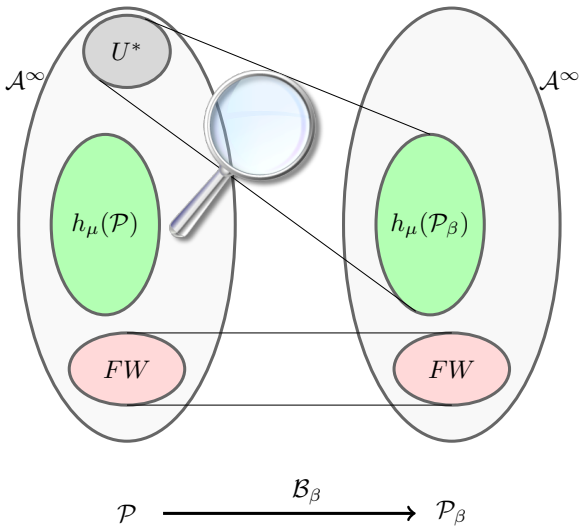


Figure 4.6: The β-map acts like a magnifier: In the parlance of large deviation theory, it "twists" or "tilts" the sequence distribution in a way that focuses on the probability of a chosen rare-event class. Fixing β, the β-map changes the energy U of a class to $U_\beta = \beta U - \log_2 \widehat{\lambda}_\beta$. In particular, a subset with energy U^* maps to the typical set of a new process that has energy $h_\mu(\mathcal{P}_\beta)$. The set FW of forbidden sequences is invariant under the β-map.

This sets up the theorem's proof. Using the fact that the process' metric entropy is the typical set's energy, the energy of $\mathcal{P}_\beta$'s typical set is $h_\mu(\mathcal{P}_\beta)$. (Refer to Fig. 4.6.) The lemma tells us how the β-map changes energy. Using this, we can identify the bubble with energy U^* that is typically generated by $M(\mathcal{P}_\beta)$, it has:

$$h_\mu(\mathcal{P}_\beta) = \beta U^* - \log_2 \widehat{\lambda}_\beta \ .$$

This completes the theorem's proof.

Chapter 5

Extreme Quantum Memory Advantage for Biased Sampling

5.1 Overview

We introduce a quantum algorithm for memory-efficient biased sampling of rare events generated by classical memoryful stochastic processes. Two efficiency metrics are used to compare quantum and classical resources for rare-event sampling. For a fixed stochastic process, the first is the classical-to-quantum ratio of required memory. We show for two example processes that there exists an infinite number of rare-event classes for which the memory ratio for sampling is larger than r, for any large real number r. Then, for a sequence of processes each labeled by an integer size N, we compare how the classical-to-quantum required memory ratio scales with N. In this setting, since both memories can diverge as $N \to \infty$, the efficiency metric tracks how fast they diverge. An *extreme quantum memory advantage* exists when the classical memory diverges in the limit $N \to \infty$, but the quantum memory has a finite bound. We then show that finite-state Markov processes and spin chains exhibit extreme memory advantage for sampling of almost all of their rare-event classes.

5.2 Introduction

From earthquakes to financial market crashes, rare events are associated with catastrophe—from decimated social infrastructure and the substantial loss of life to global economic

collapse. Though rare, their impact cannot be ignored. Prediction and modeling such rare events is essential to mitigating their effects. However, this is particularly challenging, often requiring huge datasets and massive computational resources, precisely because the events of interest are rare.

Ameliorating much of the challenge, *biased* or *extended sampling* [23, 24] is an effective and now widely-used method for efficient generation and analysis of rare events. The underlying idea is simple to state: transform a given distribution to a new one where previously-rare events are now typical. This concept was originally proposed in 1961 by Miller to probe the rare events generated by discrete-time, discrete-value Markov stochastic processes [154]. It has since been extended to address non-Markovian processes [128]. The approach was also eventually adapted to continuous-time first-order Markov processes [176–178]. Today, the statistical analysis of rare events is a highly developed toolkit with broad applications in sciences and engineering [179]. Given this, it is perhaps not surprising that the idea and its related methods appear under different appellations, depending on the research arena. For example, large deviation theory refers to the *s-ensemble method* [155, 156], the *exponential tilting algorithm* [158, 159], or as generating *twisted distributions*.

In 1997, building on biased sampling, Torrie and Valleau introduced *umbrella sampling* into Monte Carlo simulation of systems whose energy landscapes have high energy barriers and so suffer particularly from poor sampling [180]. Since then, stimulated by computational problems arising in statistical mechanics, the approach was generalized to *Ferrenberg-Swendsen reweighting*, later still to *weighted histogram analysis* [181], and more recently to *Wang-Landau sampling* [182].

When generating samples for a given stochastic process one can employ alternative types of algorithm. There are two main types—Monte Carlo or finite-state machine algorithms. Here, we consider finite-state machine algorithms based on Markov chains (MC) [76, 183] and hidden Markov models (HMM) [4–6]. For example, if the process is Markovian one uses MC generators and, in more general cases, one uses HMM generators.

When evaluating alternative approaches the key questions that arise concern algorithm

speed and memory efficiency. For example, it turns out there are HMMs that are always equally or more memory efficient than MCs. There are many finite-state HMMs for which the analogous MC is infinite-state [102]. And so, when comparing all HMMs that generate the same process, one is often interested in those that are most memory efficient. For a generic stochastic process, the most memory efficient classical HMM known currently is the ϵ-*machine* of computational mechanics [22]. The memory it requires is called the process' *statistical complexity* C_μ [56].

Today, we have come to appreciate that several important mathematical problems can be solved more efficiently using a quantum computer. Examples include quantum algorithms for integer factorization [49], search [50], eigen-decomposition [51], and solving linear systems [52]. Not long ago and for the first time, Ref. [8] provided a quantum algorithm that can perform stochastic process sample-generation using less memory than the best-known classical algorithms. Recently, using a stochastic process' higher-order correlations, a new quantum algorithm—the *q-machine*—substantially improved this efficiency and extended its applicability [12]. More detailed analysis and a derivation of the closed-form quantum advantage of the q-machine is given in a sequel [13]. Notably, the quantum advantage has been verified experimentally for a simple case [17].

The following brings together techniques from large deviation theory, classical algorithms for stochastic process generation, computational complexity theory, and the newly introduced quantum algorithm for stochastic process generation to propose a new, memory efficient quantum algorithm for the biased sampling problem. We show that there can be an extreme advantage in the quantum algorithm's required memory compared to the best known classical algorithm where the required memory for classical algorithm grows unboundedly with problem size, but is bounded from above for the quantum algorithm. Three examples are analyzed here. The first is the simple, but now well-studied *Perturbed Coin Process*. The second is a more physical example—a stochastic process that arises from the Ising next-nearest-neighbor spin system in contact with thermal reservoir. The third is a sequence of processes generated by a series of Ising N-nearest-neighbor Hamiltonians.

Today, we know of several different sampling problems for which their best quantum

algorithm has an advantage compared to the best classical algorithm. These sampling problems fall into two categories. First are those in which the problem is quantum in nature, such as boson sampling [184]. Second are the ones in which the target problem is classical. Function sampling [185] and mixing [186] are in this category.

On the one hand, the advantage for both boson sampling and mixing appears in shorter run times for the quantum algorithm. For the problem of rare-event sampling we study here the run times for both classical and quantum algorithms are the same. On the other hand, for function sampling advantage appears in smaller required memory; this is similar to our problem. In both boson sampling and function sampling, the advantage appears as an increasing function of problem size. While for mixing it is function of the spectral gap—that is, a property of the problem input—the Markov chain of interest, in that case. The quantum memory advantage we introduce here is both a function of problem size and a property of the input instance.

In the boson sampling problem, a linear system scatters N individual bosons into $M \gg N$ output modes. The goal then is to sample from the output distribution. It is known that for large N and M, the run time for the quantum algorithm is much smaller than for the classical algorithm, while both algorithms need memory on the order of the required sample size. In mixing, a Markov chain and an initial state are given and the goal is to sample from the stationary distribution over Markov chain's states with some acceptable error margin. Denoting the spectral gap for Markov chain's transition matrix by δ, the run time for the best known classical algorithm increases faster than the quantum algorithm when $\delta \to 0$ [187]. As a result, the notion of an advantage is captured by a function of δ. In function sampling, a function $f : X \times Y \to \{0, 1\}$ and a probability distribution $\Pr(X, Y)$ over $X \times Y$ are given. Alice and Bob start with no inputs. The goal then is to sample X, Y, and Z from the distribution $(\Pr, f(\Pr))$, where Alice end up with X and Bob with Y and Z. Algorithm efficiency is then defined by how much information Alice and Bob must communicate during the algorithm. It turns out that the best known quantum algorithm has markedly smaller communication costs than the classical. In the function sampling problem, as in many other similar problems, communication cost can

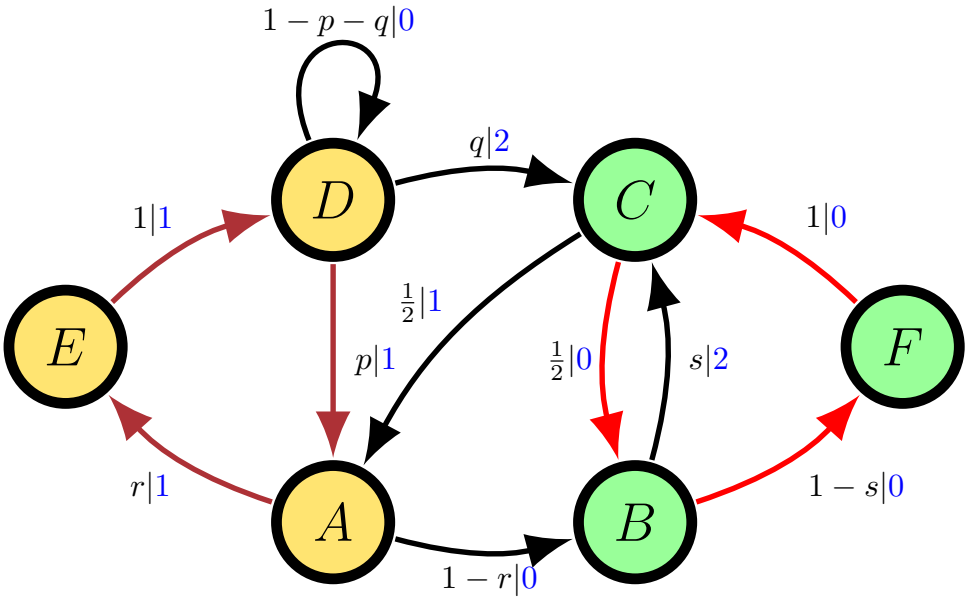


Figure 5.1: Hidden Markov model generator of a stochastic process with infinite-range statistical dependencies that requires an HMM with only six states. To generate the same process via a Markov chain requires one with an infinite number of states and so infinite memory.

be framed as a memory cost, since Alice can always write the message in a memory that Bob reads.

5.3 Classical Algorithm

The object for which we wish to generate samples is a discrete-time, discrete-value *stochastic process* [3, 4]: a probability space $\mathcal{P} = \{\mathcal{A}^\infty, \Sigma, \mathbb{P}(\cdot)\}$, where $\mathbb{P}(\cdot)$ is a probability measure over the bi-infinite chain $\ldots X_{-2}X_{-1}X_0X_1X_2\ldots$, each random variable X_i takes values in a finite, discrete alphabet $\mathcal{A}$, and Σ is the σ-algebra generated by the cylinder sets in $\mathcal{A}^\infty$. For simplicity we consider only ergodic stationary processes: that is, $\mathbb{P}(\cdot)$ is invariant under time translation—$\mathbb{P}(X_{i_1}X_{i_2}\cdots X_{i_m}) = \mathbb{P}(X_{i_1+n}X_{i_2+n}\cdots X_{i_m+n})$ for all n, m—and over successive realizations.

Sampling or *generating* a given stochastic process refers to producing a finite realization that comes from the process' probability distribution. There are two main generation (sampling) problems: *sequential generation* and *simultaneous generation* [55]. In sequential generation or one-shot sampling the goal is to generate one long sample from the given

process. However, in simultaneous generation the goal is to generate $M \gg 1$ realizations of a process simultaneously, each of which is statistically independent of the others.

Generally, generating a process via its probability measure $\mathbb{P}(\cdot)$ is impossible due to the vast number of allowed realizations and, as a result, this prosaic approach requires an unbounded amount of memory. Fortunately, there are more compact ways than specifying in-full the probability measure on the sequence sigma algebra. This recalls the earlier remark that HMMs can be arbitrarily more compact than alternative algorithms for the task of generation.

An HMM is specified by a tuple $\left\{\boldsymbol{\mathcal{S}}, \mathcal{A}, \{T^{(x)}, x \in \mathcal{A}\}\right\}$. In this, $\boldsymbol{\mathcal{S}}$ is a finite set of states, $\mathcal{A}$ is a finite alphabet, and $\{T^{(x)}, x \in \mathcal{A}\}$ is a set of $|\boldsymbol{\mathcal{S}}| \times |\boldsymbol{\mathcal{S}}|$ substochastic symbol-labeled transition matrices whose sum $T = \sum_{x \in \mathcal{A}} T^{(x)}$ is a stochastic matrix.

As an example, consider the HMM state-transition diagram shown in Fig. 5.1, where $\boldsymbol{\mathcal{S}} = \{A, B, C, D, E, F\}$, $\mathcal{A} = \{0, 1, 2\}$, and we have three 6×6 substochastic matrices $T^{(0)}$, $T^{(1)}$, and $T^{(2)}$. Each edge is labeled $p|x$ denoting the transition probability p and a symbol $x \in \mathcal{A}$ which is emitted during the transition. In this HMM, of the two edges exiting state C, one enters state B and the other enters state A. The edges from C to A and C to B are labeled by $\frac{1}{2}|1$ and $\frac{1}{2}|0$. This simply means that if the HMM is in the state C, then with probability $\frac{1}{2}$ it goes to the state A and emits the symbol 1 and with probability $\frac{1}{2}$ it goes to state B and emits symbol 0. Following these transition rules in succession generates realizations in the HMM's process.

How does this generation method compare to generating realizations of the same process via a finite Markov chain? (Recall that states in a MC not hidden: $\mathcal{A} = \boldsymbol{\mathcal{S}}$) It turns out that this cannot be implemented, since generating a symbol can depend on the infinite history. That is, the process has infinite Markov order. As a result, to generate a realization using a Markov chain one needs an infinite number of Markovian states. In other words, implementing the Markov chain algorithm to generate process samples on a conventional computer requires an infinite amount of memory.

To appreciate the reason behind the process' infinite Markov order, refer to Fig. 5.1's HMM. There are two length-3 state-loops consisting of the edges colored red (right side

of state-transition diagram) and those colored maroon (left side). Note that if the HMM generates n 1s in a row, we will not know the HMM's current state, only that it is either A, D, or E. This state uncertainty (entropy) is bounded away from 0. The observation holds for the other loop and its sequences of symbol 0 and the consequent ambiguity among states B, C, and F. Thus, there exist process realizations from which we cannot determine the future statistics, independent of the number of symbols seen. This means that the process statistics depend on infinite past sequences—the process has infinite Markov order. To emphasize, implementing a MC algorithm for this requires infinite memory. The contrast with the finite HMM method is an important lesson: HMMs are strictly more powerful generators, as a class of algorithms, than Markov chain generators.

For any given process $\mathcal{P}$, there are an infinite number of HMMs that generate it. Therefore, one is compelled to ask, which algorithm requires the least memory for implementation? To appreciate the answer, let's first address how much state memory one needs to run an HMM.

Consider sequential generation in which the goal is to produce a very long realization of a process. For this, we use one computer with a code that runs the algorithm (HMM). At each step, the computer must memorize the current HMM state. Assuming the HMM has N states, this requires $\log_2(N)$ bits of memory. As a result, if one wishes to implement one-shot sampling using the minimum required memory then, over all the process' HMM generators, one needs to find that with the minimum number of states.

Here, though, we are interested in simultaneous generation for which the goal is to simultaneously generate $M \gg 1$ process realizations, each of which is statistically independent of the others. The effective implementation uses M computers each with the above code. Similar to the sequential problem, each computer must memorize the current state of its HMM. If each computer uses its own memory, each needs $\log_2(N)$ bits of memory as before. The total memory is then $M \log_2(N)$ bits. However, we can reduce required memory by using one large memory shared among the computers. Figure 5.2 depicts this schematically. In this way, according to Shannon's coding theorem [21], we can encode HMM states to reduce the amount of memory down to $M\,\mathrm{H}(\mathcal{S}) \leq M \log_2(N)$

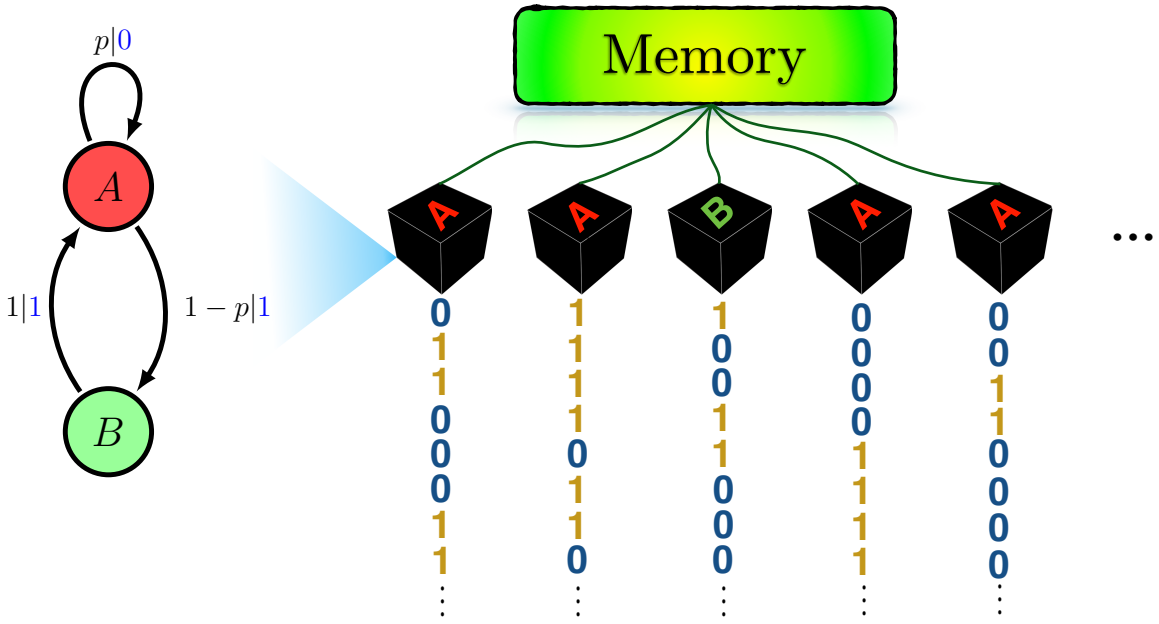


Figure 5.2: (Left) Even Process ϵ-machine. (Right) Schematic of simultaneous generation problem. Each black box contains an Even Process generator. They all share the same memory for tracking the individual generator states.

bits, where $\mathrm{H}(\mathcal{S})$ is the Shannon entropy of the probability distribution over HMM's states. The memory per sample is then just $\mathrm{H}(\mathcal{S})$. As a result, if one needs to do simultaneous sampling of a given process using minimum required memory, over all its HMM generators, one needs to find the HMM with the minimum Shannon state entropy.

For both one shot and simultaneous sampling, the best known implementation, and provably the optimal predictor, is known as the ϵ-machine M [22, 141]. It states are called *causal states*; we denote this set $\boldsymbol{\mathcal{S}}$. The average memory required for $\mathrm{M}(\mathcal{P})$ to sequentially sample process $\mathcal{P}$ is given by the process' *statistical complexity* $C_\mu(\mathcal{P})$ [56]. To calculate it:

1. Compute the stationary distribution π over causal states. π is the left eigenvector of the state-transition matrix T with eigenvalue 1: $\pi T = \pi$.

2. Calculate the state's Shannon entropy $\mathrm{H}[\boldsymbol{\mathcal{S}}] = -\sum_{\sigma \in \boldsymbol{\mathcal{S}}} \pi(\sigma) \log_2 \pi(\sigma)$.

$C_\mu = \mathrm{H}[\boldsymbol{\mathcal{S}}]$ measures the (ensemble average) memory required simultaneous sampling of

the process.

Another important, companion measure is h_μ, the process' *metric entropy* (or Shannon entropy rate) [150]:

$$h_\mu(\mathcal{P}) = -\lim_{n\to\infty} \frac{1}{n} \sum_{w\in\mathcal{A}^n} \mathbb{P}(w) \log_2 \mathbb{P}(w) \ .$$

Although sometimes confused, it is important to emphasize that h_μ describes randomness in realizations, while C_μ is the average memory required to generate them.

5.4 Quantum Memory Advantage

Recently, it was shown that a quantum algorithm for process generation can use less memory than the best known classical algorithm (ϵ-machine) [8]. By accounting for a process' higher-order correlations, a new quantum algorithm—the *q-machine*—was introduced that substantially improved the original and is, to date, the most memory-efficient quantum algorithm known for process generation [12]. We refer to the ratio of required classical memory C_μ to quantum memory C_q as the *quantum memory advantage*. Closed-form expressions for the quantum memory advantage are given in Ref. [13].

Importantly, the quantum advantage was recently verified experimentally for the simple *perturbed coins process* [17]. It was also discovered that the q-machine can confer an extreme quantum memory advantage. For example, when generating ground-state configurations (in a Dyson-type spin model with N-nearest-neighbor interactions at temperature T), the quantum advantage scales as $NT^2/\log_2 T$ [11, 14]. Another example of extreme quantum memory advantage has appeared recently in the simulation of continuous-time stochastic processes [9]. One consequence of quantum advantage arises in model selection where the basic question "Which process is simpler?" no longer has a well-defined answer [15]. This phenomenon recently has been experimentally observed [18]. Statistical inference of models for stochastic systems often involves controlling for model size or memory. And so, quantum statistical inference may see large improvements. Another consequence of this advantage in the context of simulations is that reduced memory can reduce the heat dissipation of simulation [188].

The following determines the quantum advantage in biased sampling of a process' rare events. In particular, we develop tools to determine how the memory requirements of classical and quantum algorithms vary over rare-event classes.

5.5 Quantum Algorithm

We define a stochastic process $\mathcal{P}$'s *quantum machine* by $\mathrm{Q}(\mathcal{P}) = \{\mathcal{H}, \mathcal{A}, \{K_x, x \in \mathcal{A}\}\}$, where $\mathcal{H}$ denotes the Hilbert space with dimension $|\boldsymbol{\mathcal{S}}|$ in which quantum states reside, $\mathcal{A}$ is the same alphabet as the given process', and $\{K_x, x \in \mathcal{A}\}$ is a set of Kraus operators we use to specify the measurement protocol for states [19].[1] Assume we have the *quantum state* (density matrix) $\rho_0 \in \mathcal{B}(H)$ in hand. We perform a measurement by applying Kraus operators and, as a result, measure X. The probability of yielding symbol $X = x$ is:

$$\mathbb{P}(X = x_0|\rho_0) = \mathrm{tr}\left(K_{x_0}\rho_0 K_{x_0}^\dagger\right) \ .$$

After measurement with outcome $X = x_0$, the new quantum state is:

$$\rho_1 = \frac{K_{x_0}\rho_0 K_{x_0}^\dagger}{\mathrm{tr}(K_{x_0}\rho_0 K_{x_0}^\dagger)} \ .$$

Repeating these measurements generates a stochastic process. The process potentially could be nonergodic, depending on the initial state ρ_0. However, starting the q-machine in the stationary state defined by:

$$\rho_s = \sum_{x\in\mathcal{A}} K_x \rho_s K_x^\dagger \ ,$$

and repeatedly measuring generates a stationary stochastic process over $x \in \mathcal{A}$. For any given process, ρ_s can be calculated by the method introduced in Ref. [13].

Our immediate goal is to design a quantum generator of a given classical process. (Section 5.7 will then take the given process to represent a rare-event class of another process.) For now, we start with the process' ϵ-machine. The construction consists of three steps, as follows.

[1] We adopt a particular form for the Kraus operators. In general, they are not unique.

First step Map every causal state $\sigma_i \in \boldsymbol{\mathcal{S}}$ to a pure quantum state $|\eta_i\rangle$. Each signal state $|\eta_i\rangle$ encodes the set of length-R sequences that may follow σ_i, as well as each corresponding conditional probability:

$$|\eta_i\rangle \equiv \sum_{w \in \mathcal{A}^R} \sqrt{\mathbb{P}(w|\sigma_i)} \ |w\rangle \ ,$$

where w denotes a length-R sequence, $\mathbb{P}(w|\sigma_i) = \mathbb{P}(X_0 \cdots X_{R-1} = w | \mathcal{S}_0 = \sigma_i)$, and R is the process' the Markov order. The resulting Hilbert space is $\mathcal{H}_w$ with size $|\mathcal{A}|^R$, the number of length-R sequences, with basis elements $|w\rangle = |x_0\rangle \otimes \cdots \otimes |x_{R-1}\rangle$. From here on out, assume all $|\eta_i\rangle$s are linearly independent.[2] As a result, one defines $|\boldsymbol{\mathcal{S}}|$ new states $|\xi_i\rangle$ that reside in a Hilbert space of size $|\boldsymbol{\mathcal{S}}|$. (This is much smaller than the $|\eta_i\rangle$'s Hilbert space, which has size $|\mathcal{A}|^R$.) Moreover, the $|\xi_i\rangle$s have the same pairwise overlaps as the $|\eta_i\rangle$s. That is, for all i, j:

$$\langle \xi_i | \xi_j \rangle = \langle \eta_i | \eta_j \rangle \ .$$

Reference [13] developed a method to calculate all the overlaps $\langle \eta_i | \eta_j \rangle$ for a given process in closed form. Since the $|\eta_i\rangle$s are linearly independent one can use the overlaps to construct the $|\xi_i\rangle$s.

Second step Form a matrix Ξ by assembling the signal states:

$$\Xi = \left[|\xi_0\rangle \quad |\xi_1\rangle \quad \cdots \quad |\xi_{|\boldsymbol{\mathcal{S}}|-1}\rangle \right] \ .$$

Define $|\boldsymbol{\mathcal{S}}|$ new bra states $|\widetilde{\xi_i}\rangle$:

$$\begin{bmatrix} \langle \widetilde{\xi_0} | \\ \langle \widetilde{\xi_1} | \\ \cdots \\ \langle \widetilde{\xi_{|\boldsymbol{\mathcal{S}}|-1}} | \end{bmatrix} = \Xi^{-1} \ .$$

[2] The procedure requires only slight modification for linearly dependent causal states and, in any case, does not affect the results.

That is, we design the new bra states such that we obtain the identity:

$$\begin{bmatrix} \langle\widetilde{\xi_0}| \\ \langle\widetilde{\xi_1}| \\ \cdots \\ \langle\widetilde{\xi_{|\boldsymbol{\mathcal{S}}|-1}}| \end{bmatrix} \begin{bmatrix} |\xi_0\rangle & |\xi_1\rangle & \cdots & |\xi_{|\boldsymbol{\mathcal{S}}|-1}\rangle \end{bmatrix} = \mathrm{I} \ .$$

Third step Define $|\mathcal{A}|$ Kraus operators K_x for all $x \in \mathcal{A}$ via:

$$K_x = \sum_{i,j} \sqrt{T_{ij}^x} \, |\xi_j\rangle \, \langle\widetilde{\xi_i}| \ .$$ [3]

By applying Kraus operators repeatedly one generates the target stochastic process. For example, consider the case in which the q-machine is in state $|\xi_i\rangle \langle\xi_i|$ and we apply the Kraus operators. Then, if we do not make a measurement, the next state is $\sum_{j,x} T_{ij}^x |\xi_j\rangle \langle\xi_j|$. Let us say, though, that we make a measurement and the result is x. The next state is $|\xi_j\rangle \langle\xi_j|$, where j is the unique index where T_{ij}^x is nonzero. Appendix 5.10.2 shows that the stationary state is:

$$\rho_s = \sum_i \pi_i \, |\xi_i\rangle \, \langle\xi_i| \ .$$

The Hilbert space in which the algorithm operates has dimension $|\boldsymbol{\mathcal{S}}|$. Since the operation is not unitary in this space and measurements are not projective, one may argue that $|\boldsymbol{\mathcal{S}}|$ is not the actual size of the Hilbert space needed to physically implement the algorithm. However, it was recently shown that the algorithm presented here can be implemented by unitary operations and projective measurements in a Hilbert space with dimension $|\boldsymbol{\mathcal{S}}||\mathcal{A}|$ [20]. This new result gives an experimental implementation of our algorithm.

Using the quantum generator $\mathrm{Q}(\mathcal{P})$, the required average memory for simultaneous generation of process $\mathcal{P}$ is $C_q(\mathcal{P}) = S(\rho_s)$, where $S(\rho) = -\mathrm{tr}(\rho \log \rho)$ denotes the *von Neumann entropy* [19].[3] Since the algorithm can be implemented by unitary operations

[2] Appendix 5.10.1 proves the completeness of these operators.

[3] Von Neumann entropy is a well-accepted measure of quantum advantage in communication games [189].

and projective measurements, the entropy of the quantum model stays constant at C_q at all times during the simulation process. This confirms that C_q is a valid measure of memory, while the minimal dimensions needed for sequential (one-shot) generation of $\mathcal{P}$ is at most $|\boldsymbol{\mathcal{S}}||\mathcal{A}|$.

Comparing memory efficiencies of classical and quantum algorithms requires an efficiency metric. Depending on the setting, there are two one can use. In the *single-process* case, $\mathcal{P}$ is given and memory efficiency is defined as a the ratio of required memory for the classical algorithm to the quantum algorithm. Here, since we only explore finite-size stochastic processes, both memories are finite and the ratio is a good quantitative efficiency measure.

In the *multi-process* case, a series of stochastic processes is given, with each process labeled by an integer N that measures process size. Then, memory efficiency is defined by how the memory scales in N for the classical algorithm compared to the quantum. This metric is closer to relative-complexity definitions familiar in computation complexity theory. In the present case, since both memories are allowed to diverge when $N \to \infty$, the quantitative measure of efficiency tracks how fast they diverge. We say we have *extreme quantum memory advantage* when the classical memory diverges as $N \to \infty$, but quantum memory converges to a finite quantity.

5.6 Typical Realizations

At this point, we established classical and quantum representations of processes and characterized their respective memory requirements. Using this, our purpose now shifts to monitor classical and quantum resources required to generate rare-event classes of a process' realizations. Our first task is to review the theory of typical events and their complement—rare events.

The concept of a stochastic process is quite general. Any physical system that exhibits stochastic dynamics in time or space may be thought of as *generating* a stochastic process. In the spatial setting one considers not time evolution, but rather the spatial "dynamic". For example, consider a one-dimensional noninteracting Ising spin-$\frac{1}{2}$ chain with classical

Hamiltonian $H = -\sum_{i=1}^{n} h\sigma_i$ in contact with a thermal reservoir at temperature T. After thermalizing, a spin configuration at one instant of time may be thought of as having been generated by a process that scans the lattice left-to-right (or, equivalently, right-to-left). The probability distribution over these spatial-translation invariant configurations defines a stationary stochastic process—a simple Markov random field.

For $n \gg 1$, one can ask for the probability of seeing k up spins. The Strong Law of Large Numbers [147] guarantees that for large n, the ratio k/n almost surely converges to $p_\uparrow = \frac{1}{2}\left(1 + \tanh(h/k_B T)\right)$. That is:

$$\mathbb{P}\left(\lim_{n\to\infty} \frac{k}{n} = p_\uparrow\right) = 1 \ .$$

Informally, a *typical sequence* is one that has close to $p_\uparrow n$ up-spins. However, this does not preclude seeing other kinds of long runs, e.g., all up-spins or all down-spins. It simply means that the latter are *rare events*, compared to the typical ones.

Now, let us formally define the concept of typical realizations and, consequently, rare ones. Consider a given process $\mathcal{P}$ and let $\mathcal{A}^n$ denote the set of of all possible length-n realizations. Then, for an arbitrary $0 < \epsilon \ll 1$ the process' *typical set* [21, 148, 149] is defined:

$$A_\epsilon^n \equiv \{w : 2^{-n(h_\mu+\epsilon)} \leq \mathbb{P}(w) \leq 2^{-n(h_\mu-\epsilon)}, w \in \mathcal{A}^n\}, \tag{5.1}$$

where h_μ is the process' Shannon entropy rate, introduced above.

According to the *Shannon-McMillan-Breiman theorem* [140, 151, 152], for a given $\epsilon \ll 1$, sufficiently large n^*, and $w \in \mathcal{A}^n$:

$$\mathbb{P}(w \notin A_\epsilon^n) \leq \epsilon \ , \tag{5.2}$$

for all $n \geq n^*$. There are two important lessons here. First, from Eq. (5.1) we see that all sequences in the typical set have approximately the same probability. More precisely, the probability of typical sequences decays at the *same exponential rate*. The following adapts this to use decay rates to identify distinct sets of rare events. Second, coming from Eq. (5.2), for large n the probability of sequences falling outside the typical set is close to zero—these are the sets of rare sequences.

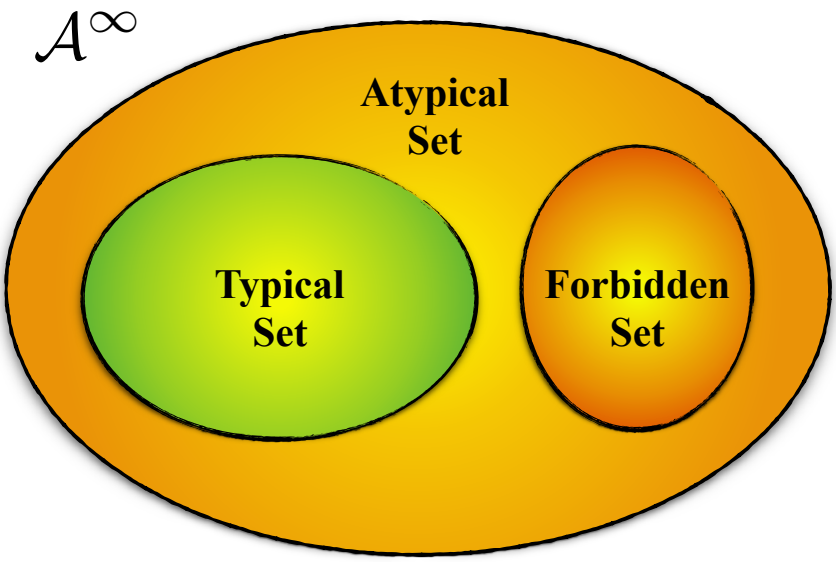


Figure 5.3: For a given process, the space $\mathcal{A}^\infty$ of all sequences is partitioned into forbidden sequences, sequences in the typical set, and sequences neither forbidden nor typical—the *atypical* or rare sequences.

Another important consequence of the theorem is that sequences generated by a stationary ergodic process fall into one of three partitions; see Fig. 5.3. The first contains sequences that are never generated; they fall in the the *forbidden set*. For example, the HMM in Fig. 5.1 never generates sequences that have consecutive 2s. The second partition consists of those in the typical set—the set with probability close to one, as in Eq. (5.1). And, the last contains sequences in a family of atypical sets—realizations that are rare to different degrees. We now refine this classification by dividing the atypical sequences into identifiable subsets, each with their own characteristic rarity.

Mirroring the familiar *Boltzmann weight* in statistical physics [153], in the $n \to \infty$ limit, we define the subsets $\Lambda_u^{\mathcal{P}} \subset \mathcal{A}^\infty$ for a process $\mathcal{P}$ as:

$$\Lambda_{u,n}^{\mathcal{P}} = \left\{ w : -\frac{\log_2 \mathbb{P}(w)}{n} = u, \ w \in \mathcal{A}^n \right\} \text{ and} \tag{5.3}$$

$$\Lambda_u^{\mathcal{P}} = \lim_{n\to\infty} \Lambda_{u,n}^{\mathcal{P}} \ .$$

This partitions $\mathcal{A}^\infty$ into disjoint subsets $\Lambda_u^{\mathcal{P}}$ in which all $w \in \Lambda_u^{\mathcal{P}}$ have the same probability decay rate u. Physics vernacular would speak of the sequences having the same *energy*

density u.[4] Figure 5.4 depicts these subsets as "bubbles" of equal energy. Equation (5.1) says the typical set is that bubble with energy equal to the process' Shannon entropy rate: $u = h_\mu$. All the other bubbles contain rare events, some rarer than others. They exhibit faster or slower probability decay rates.

Employing a process' HMM to generate realizations produces sequences in the typical set with probability close to one and, rarely, atypical sequences. Imagine that one is interested in a particular class $\Lambda_u^{\mathcal{P}}$ of rare sequences, say, those with energy u. (One might be concerned about the class of large-magnitude earthquakes or the emergence of major instabilities in the financial markets, for example.) How can one efficiently generate these rare sequences? We now show that there is a new process $\mathcal{P}^u$ whose typical set is $\Lambda_u^{\mathcal{P}}$ and this returns us directly to the challenge of biased sampling.

5.7 Biased Sampling

Consider a finite set of configurations $\{c_i\}$ with probabilities specified by distribution $\Pr(\cdot)$ and an associated set $\{\omega_i\}$ of *weighting factors*. Consider the procedure of reweighting that introduces a new distribution $\widetilde{\Pr}(\cdot)$ over configurations where:

$$\widetilde{\Pr}(c_i) = \frac{\Pr(c_i)\exp(\omega_i)}{\sum_i \Pr(c_i)\exp(\omega_i)} \ .$$

Given a process $\mathcal{P}$ and its ϵ-machine $M(\mathcal{P})$, How do we construct an ϵ-machine $M(\mathcal{P}^u)$ that generates $\mathcal{P}$'s atypical sequences at some energy density $u \neq h_\mu$ or, as we denoted it, the set $\Lambda_u^{\mathcal{P}}$? Here, we answer this question by constructing a map $\mathcal{B}_\beta : \mathcal{P} \to \mathcal{P}_\beta$ from the original process $\mathcal{P}$ to a new one $\mathcal{P}_\beta$. The map is parametrized by $\beta \in \mathbb{R}/\{0\}$ which indexes the rare set of interest. (We use β for convenience here, but it is related to u by a function introduced shortly.) Both processes $\mathcal{P} = \left\{\mathcal{A}^\infty, \Sigma, \Pr(\cdot)\right\}$ and $\mathcal{P}_\beta = \left\{\mathcal{A}^\infty, \Sigma, \Pr_\beta(\cdot)\right\}$ are defined on the same measurable sequence space. The measures differ, but their supports (allowed sequences) are the same. For simplicity we refer to $\mathcal{B}_\beta$ as the *β-map*.

Assume we are given $M(\mathcal{P}) = \left\{\boldsymbol{\mathcal{S}}, \mathcal{A}, \{T^{(x)}, x \in \mathcal{A}\}\right\}$. We showed that for every probability decay rate or energy density u, there exists a particular β such that $M(\mathcal{P}_\beta)$

[4] u, considered as a random variable, is sometimes called a *self process* [129].

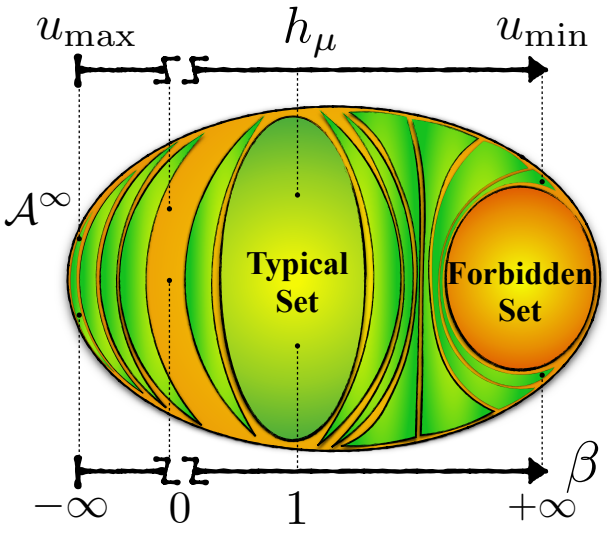


Figure 5.4: Space of all sequences $\mathcal{A}^\infty$ partitioned into Λ_us—isoenergy density or equal probability-decay-rate bubbles—in which all sequences in the same Λ_u have the same energy density u. The typical set is one such bubble with energy equal to Shannon entropy rate: $u = h_\mu$. Another important class is the forbidden set, in which all sequences do not occur. The forbidden set can also be interpreted as the subset of sequences with infinite positive energy. By applying the map $\mathcal{B}_\beta$ to the process and changing β continuously from $-\infty$ to $+\infty$ (excluding $\beta = 0$) one can generate any rare class of interest $\Lambda_u^{\mathcal{P}}$. $\beta \to -\infty$ corresponds to the most probable sequences with the largest energy density $u_{\max}$, $\beta = 1$ corresponds to the typical set and $\beta \to +\infty$ corresponds to the least probable sequences with the smallest energy density $u_{\min}$.

typically generates the words in $\Lambda_{u,n}^{\mathcal{P}}$ for large n [55]. The β-map which establishes this is calculated by a construction that relates $M(\mathcal{P})$ to $M(\mathcal{P}_\beta) = \{\boldsymbol{\mathcal{S}}, \mathcal{A}, \{\mathbf{S}_\beta^{(x)}, x \in \mathcal{A}\}\}$—the HMM that generates $\mathcal{P}_\beta$:

1. For each $x \in \mathcal{A}$, construct a new matrix $\mathbf{T}_\beta^{(x)}$ for which $\left(\mathbf{T}_\beta^{(x)}\right)_{ij} = \left(\mathbf{T}^{(x)}\right)_{ij}^{\beta}$.
2. Form the matrix $\mathbf{T}_\beta = \sum_{x \in \mathcal{A}} T_\beta^{(x)}$.
3. Calculate $\mathbf{T}_\beta$'s maximum eigenvalue $\widehat{\lambda}_\beta$ and corresponding right eigenvector $\widehat{\mathbf{r}}_\beta$.
4. For each $x \in \mathcal{A}$, construct new matrices $\mathbf{S}_\beta^{(x)}$ for which:

$$\left(\mathbf{S}_\beta^{(x)}\right)_{ij} = \frac{\left(\mathbf{T}_\beta^{(x)}\right)_{ij} (\widehat{\mathbf{r}}_\beta)_j}{\widehat{\lambda}_\beta (\widehat{\mathbf{r}}_\beta)_i} \ .$$

Having constructed the new process $\mathcal{P}_\beta$ by introducing its generator, we use the latter to produce some rare set of interest $\Lambda_{u,n}^{\mathcal{P}}$.

Theorem 3. *In the limit $n \to \infty$, within the new process $\mathcal{P}_\beta$ the probability of generating realizations from the set $\Lambda^{\mathcal{P}}_{u,n}$ converges to one:*

$$\lim_{n\to\infty} \Pr_\beta(\Lambda^{\mathcal{P}}_{u,n}) = 1 \ ,$$

where:

$$u = \beta^{-1}\big(h_\mu(\mathcal{P}_\beta) - \log_2 \widehat{\lambda}_\beta\big) \ . \tag{5.4}$$

In addition, in the same limit the process $\mathcal{P}_\beta$ assigns equal energy densities over all the members of the set $\Lambda^{\mathcal{P}}_{u,n}$.

Proof. *See Ref. [55].*

As a result, for large n the process $\mathcal{P}_\beta$ typically generates the set $\Lambda^{\mathcal{P}}_{u,n}$ with the specified energy u. The process $\mathcal{P}_\beta$ is sometimes called the *auxiliary*, *driven*, or *effective* process [157, 190, 191]. Examining the form of the energy in Eq. (5.4), one sees that there is a one-to-one relationship between β and u. And so, we can equivalently denote the process $\mathcal{P}_\beta$ by $\mathcal{P}^u$. More formally, every word in $\Lambda^{\mathcal{P}}_u$ with probability measure one is in the typical set of process $\mathcal{P}_\beta$.

The β-map construction guarantees that the HMMs $M(\mathcal{P})$ and $M(\mathcal{P}_\beta)$ have the same states and transition topology: $\left(\mathbf{T}^{(x)}_\beta\right)_{ij} \neq 0 \iff \left(\mathbf{S}^{(x)}_\beta\right)_{ij} \neq 0$. The only difference is in their transition probabilities. $M(\mathcal{P}_\beta)$ is not necessarily an ϵ-machine—the most memory-efficient classical algorithm that generates the process. Typically, though, $M(\mathcal{P}_\beta)$ is an ϵ-machine. Moreover, there are only finitely many βs for which it is not. (More detailed development along these lines will appear in a sequel.)

5.8 Quantum and Classical Memory For Biased Sampling

Having introduced the necessary background to compare classical versus quantum models and to appreciate typical versus rare realizations, we are ready to investigate the quantum memory advantage when generating a given process' rare events.

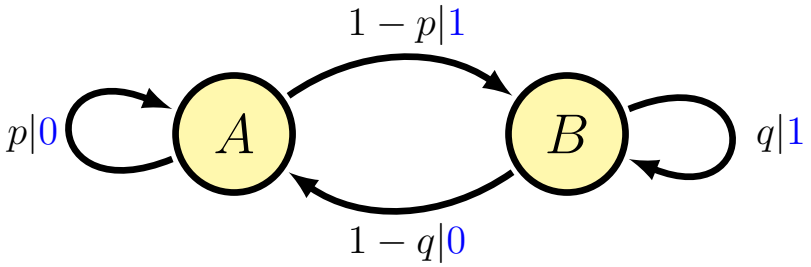


Figure 5.5: ϵ-Machine generator of the Perturbed Coins Process. Edges are labeled with conditional transition probabilities and emitted symbols. For example, for the self-loop on state A, $p|0$ indicates the transition is taken with probability $\Pr(0|A) = p$ and the symbol 0 is emitted.

The last section concluded that the memory required by the classical algorithm to generate $\mathcal{P}$'s rare sequences with energy density u is:

$$C_\mu(\mathcal{P}_\beta) = C_\mu(\mathcal{B}_\beta(\mathcal{P})) \ ,$$

where u and β are related via $u = \beta^{-1}\big(h_\mu(\mathcal{P}_\beta) - \log_2 \widehat{\lambda}_\beta\big)$. Similarly, the memory required by the quantum algorithm to generate the rare class with energy density u is:

$$C_q(\mathcal{B}_\beta(\mathcal{P})) \ .$$

For simplicity, we denote these two quantities by $C_\mu(\beta) \equiv C_\mu(\mathcal{P}_\beta)$ and $C_q(\beta) \equiv C_q(\mathcal{P}_\beta)$.

The following analyzes the advantage for three example processes—two in the *single-process* setting and one in the *multi-process* setting. For the first two, we consider particular given stochastic processes and study the advantage (memory ratio) as the metric of memory efficiency. In the third example, we consider a series of stochastic processes labeled by their size N and compare how both classical and quantum memories scale with N. The ratio of scaling then is the metric for memory efficiency.

5.8.1 Quantum Memory Advantage for a Simple Markov Process

Let's start in the single-process setting in which an individual stochastic process is given. Consider the case where we have two biased coins, call them A and B, and each has a different bias p and $1 - q$ both for Heads (symbol 0), respectively. When we flip a coin,

if the result is Heads, then on the next flip we choose coin A. If the result is Tails, we choose coin B. Flipping the coins over and over again results in a process $\mathcal{P}^{\text{pc}}$ called the *Perturbed Coins Process* [8]. Figure 5.5 shows the process' ϵ-machine generator $M(\mathcal{P}^{\text{pc}})$, where $\boldsymbol{\mathcal{S}} = \{A, B\}$ and $\mathcal{A} = \{0, 1\}$.

One can also generate this process with a quantum machine $\text{Q}(\mathcal{P}^{\text{pc}})$. Using the construction introduced in Sec. 5.5, it has two Kraus operators corresponding to symbols 0 and 1:

$$K_0 = \begin{bmatrix} \sqrt{p} & 0 \\ \sqrt{1-p} & 0 \end{bmatrix}$$
$$K_1 = \begin{bmatrix} 0 & \sqrt{1-q} \\ 0 & \sqrt{q} \end{bmatrix} .$$

For its stationary state distribution we have:

$$\rho_s = \frac{1}{2-p-q} \begin{bmatrix} 1-q & \alpha \\ \alpha & 1-p \end{bmatrix} ,$$

where $\alpha = (1-q)\sqrt{p(1-p)} + (1-p)\sqrt{q(1-q)}$. Calculation detail is given in App. 5.10.3.

Figure 5.6 shows the classical and quantum memory costs to generate rare realizations: $C_\mu(\beta)$ and $C_q(\beta)$ versus β for different β-classes. Surprisingly, the two costs exhibit completely different behaviors. For example, $\lim_{\beta \to 0} C_q = 0$, while $\lim_{\beta \to 0} C_\mu = 1$. More interestingly, as the inset demonstrates, even though both $C_\mu(\beta)$ and $C_q(\beta)$ vanish exponentially fast, in the limit of $\beta \to \infty$ $C_q(\beta)$ goes to zero noticeably faster.

We define the *memory efficiency* $\eta(\beta)$ of biased sampling in the single-process setting as the ratio of classical to quantum memory:

$$\eta(\beta) \equiv \frac{C_\mu(\beta)}{C_q(\beta)} .$$

Figure 5.7 graphs $\eta(\beta)$, revealing how it divides into three distinct scaling regimes.

First, for small $|\beta|$ the memory ratio scales as $\mathcal{O}(\beta^{-2})$. Second, for large positive β the memory ratio scales exponentially $\mathcal{O}(\exp(c\beta))$ as one increases β, where c is a function of p and q. Third, for large negative β, there is no quantum advantage. Since we are analyzing

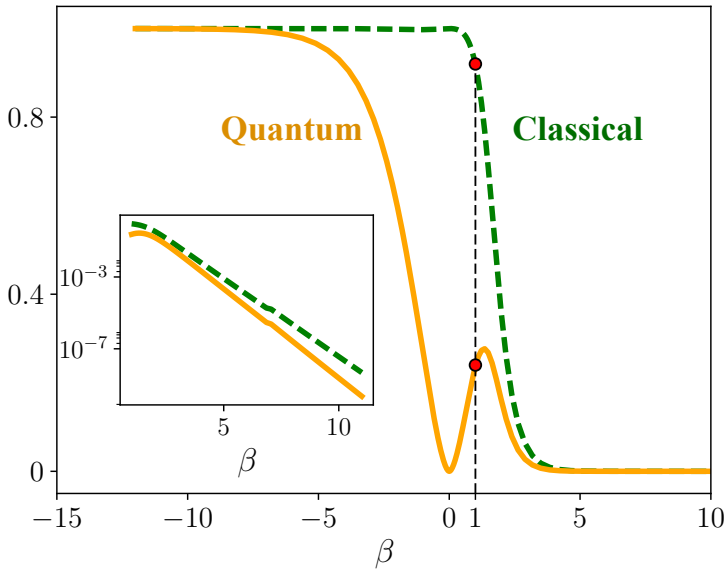


Figure 5.6: Classical memory $C_\mu(\beta)$ and quantum memory $C_q(\beta)$ versus β for biased sampling of Perturbed Coins Process' rare sequence classes: See Fig. 5.5, with $p = 0.6$ and $q = 0.8$. As the inset shows, for large β both classical and quantum memories decay exponentially with β, but the quantum memory decays faster. The vertical dashed black line and two red markers at $\beta = 1$ show the memory costs for generating typical sequences.

finite-state processes, this regime appears and is the analog of population inversion. And so, formally there are β-class events with negative β.

Such is the quantum advantage for the Perturbed Coins Process at $p = 0.6$ and $q = 0.8$. The features exhibited—the different scaling regimes—are generic for any $p > 1-q$, though. Moreover, for Perturbed Coins Processes with $p < 1 - q$, the positive and negative low temperature behaviors switch.

5.8.2 Quantum Memory Advantage for Next-Nearest-Neighbor Spin Systems

Again, consider the *single-process* setting in which an individual stochastic process is given. Let us, however, analyze the quantum advantage in a more familiar physical system. Consider a general one-dimensional ferromagnetic next-nearest-neighbor Ising spin-$\frac{1}{2}$ chain [67, 68] defined by the Hamiltonian:

$$\mathcal{H} = -\sum_i \left(s_i s_{i+1} + \tfrac{1}{4} s_i s_{i+2} \right) \ , \tag{5.5}$$

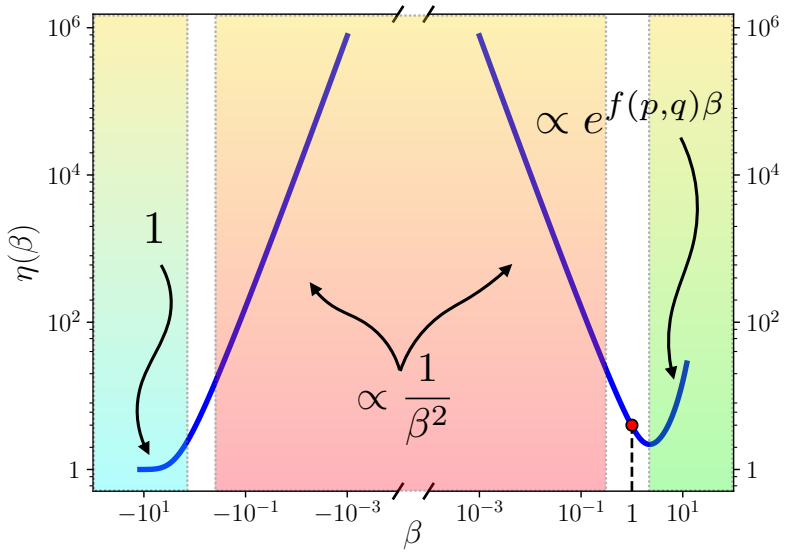


Figure 5.7: The classical to quantum memory ratio for generating rare realizations of the Perturbed Coins Process with $p = 0.6$ and $q = 0.8$ when employing its q-machine instead of its (classical) ϵ-machine. Three different advantages occur: (i) near $\beta = 0$ the ratio scales as $\mathcal{O}(\beta^{-2})$, (ii) for large positive β, it scales exponentially with β, $\mathcal{O}(\exp(f(q,p)\beta))$, and (iii) no advantage at large negative β. The vertical dashed black line and red marker at $\beta = 1$ show the memory cost for generating typical sequences.

in contact with thermal bath at temperature $k_B T = 1$. The spin s_i at site i takes on values $\{+1, -1\}$.

After thermalizing, a spin configuration at one instant of time may be thought of as having been generated left-to-right (or, equivalently, right-to-left) along the lattice. The probability distribution over these spatial-translation invariant configurations defines a stationary stochastic process. Reference [1, Eqs. (84) − (91)] showed that for any finite and nonzero temperature T, this process has Markov order $R = 2$. More to the point, the ϵ-machine that generates this process has four causal states and those states are in one-to-one correspondence with the set of length-2 spin configurations.

Figure 5.8 displays the parametrized ϵ-machine that generates this family of spin-configuration processes. To simulate the process, the generator need only remember the last two spins generated. This means the ϵ-machine has four states, $\downarrow\downarrow$, $\downarrow\uparrow$, $\uparrow\downarrow$, and $\uparrow\uparrow$. If the last two observed spins are $\uparrow\uparrow$ for example, then the current state is $\uparrow\uparrow$. We denote the probability of generating a $\downarrow$ spin given that the previous two spins were $\uparrow\uparrow$ by $p^{\downarrow}_{\uparrow\uparrow}$. If the

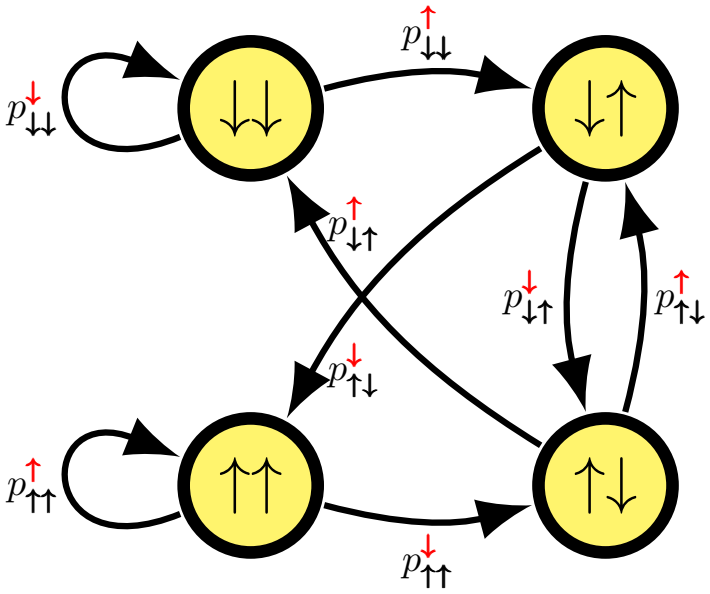


Figure 5.8: ϵ-Machine that generates the spin configurations occurring in the one-dimensional ferromagnetic next-nearest-neighbor Ising spin chain with the Hamiltonian in Eq. (5.5).

generator is in the $\uparrow\uparrow$ state and generates a $\downarrow$ spin, then the generator state changes to $\uparrow\downarrow$.

To determine the ϵ-machine transition probabilities $\{T^{(x)}\}_{x\in\mathcal{A}}$, we first compute the transfer matrix V for the Hamiltonian of Eq. (5.5) at temperature T and then extract conditional probabilities, following Ref. [1] and Ref. [14]'s appendix.

What are the classical and quantum memory costs for bias sampling of the rare spin-configuration class with decay rate u, as defined in Eq. (5.3)? First, note that u is not a configuration's actual energy density E. If we assume the system is in thermal equilibrium and thus exhibits a Boltzmann distribution over configurations, then u and E are related via:

$$u = \frac{\log_2(e)}{k_B T}(E - \mathcal{F}(T)) \ ,$$

where:

$$\mathcal{F}(T) = -k_B T \lim_{n\to\infty} \frac{1}{n} \ln \left(\sum_{\{w\in\mathcal{A}^n\}} e^{-\frac{nE(w)}{k_B T}} \right) \ .$$

This simply tells us that if a stochastic process describes thermalized configurations of a physical system with some given Hamiltonian, then every rare-event bubble in Fig. 5.4

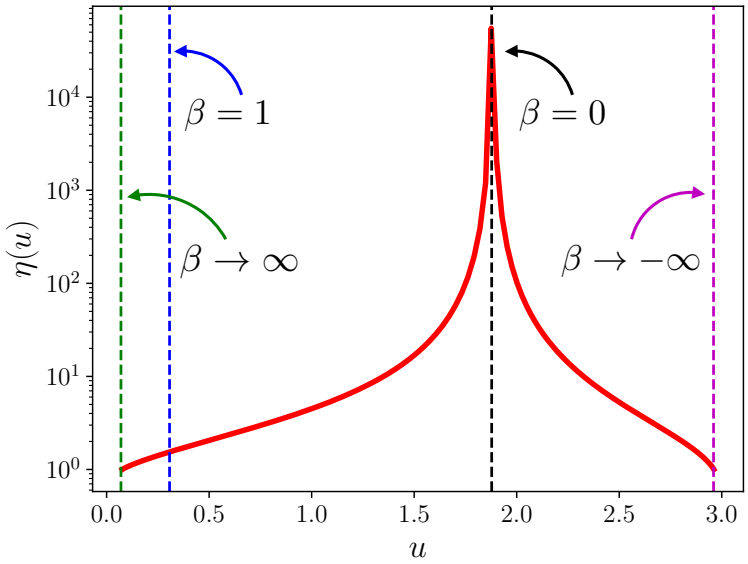


Figure 5.9: Classical to quantum memory ratio for biased sampling of Ising spin configurations: $\eta(u)$ versus decay rate u for bias sampling of equal-energy density spin configurations. Vertical lines locate βs corresponding to particular us. Note the memory ratio $\eta(u)$ diverges at $u = u_0 \approx 1.878$ corresponding to $\beta = 0$. Note that the quantum memory advantage is not limited to $\beta = 0$, but occurs in a large neighborhood around it. Quantitatively, for any arbitrary large number r there exist a number ϵ for which the rare class $\beta_0 = \epsilon$ has the memory ratio $\eta(\beta_0) > r$.

can be labeled either with β, u, or E. Moreover, there is a one-to-one mapping between every such variable pair.

Figure 5.9 plots $\eta(u)$ versus u—the memory ratio for generating rare configurations with decay rate u. To calculate $\eta(u)$ for a given process $\mathcal{P}$, first we determine the process' classical generator $M(\mathcal{P})$ using the method introduced in Ref. [141]. Second, for every $\beta \in \mathbb{R}/\{0\}$, using the map introduced in Sec. 5.7, we find the new classical generator $M(\mathcal{P}_\beta)$. Third, using the construction introduced in Sec. 5.4, we find $\mathrm{Q}(\mathcal{P}_\beta)$. Fourth, using Thm. 3 we find the corresponding u for the chosen β. Using these results gives $\eta(u(\beta)) = C_\mu(\beta)/C_q(\beta)$. By varying β in the range $\mathbb{R}/\{0\}$ we cover all the energy density us. Practically, to calculate $\eta(u)$ in Fig. (5.9), we chose 2000 $\beta \in [-10, 7.5]$.

As pointed out earlier, $\beta = 1$ always corresponds to the process itself. And, one obtains its typical sequences. As one sees in Fig. 5.9, the memory ratio $\eta(1) < 2$. This simply means that, though there is a quantum advantage generating typical sequences, it is not

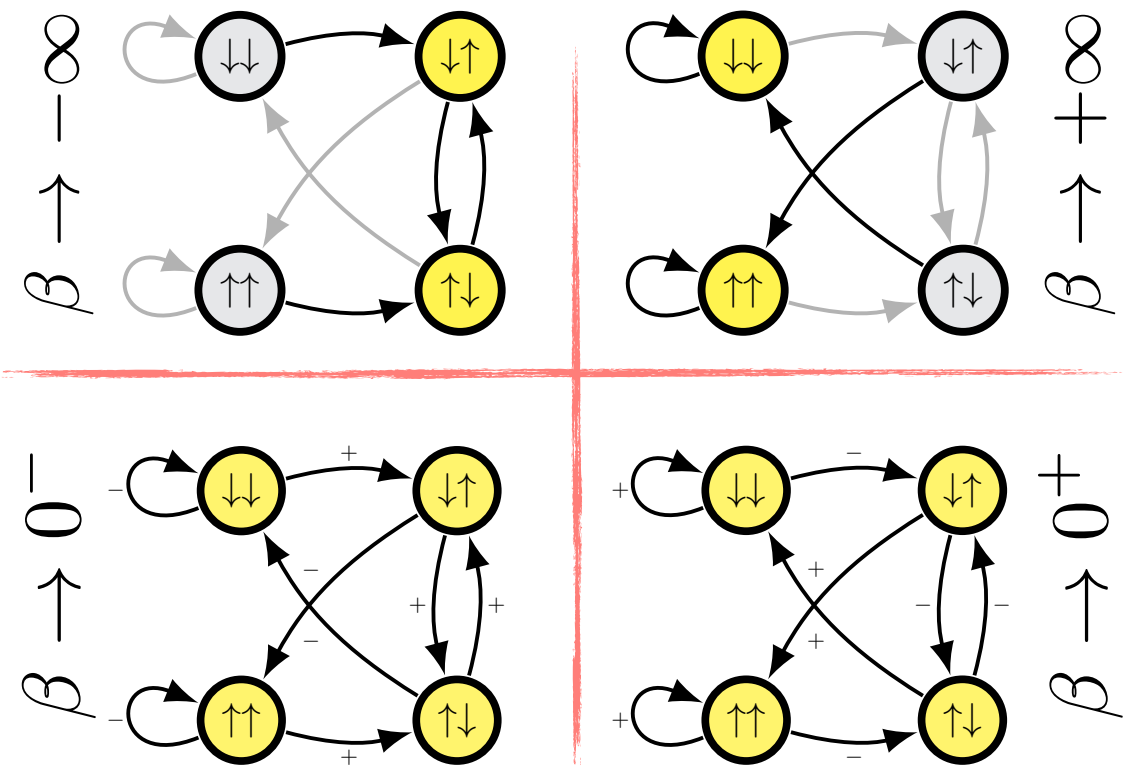


Figure 5.10: Classical generators of four important rare classes: (Top-left) Negative zero-temperature limit. (Top-right) positive zero temperature limit. (Bottom-left) Negative infinite temperature limit. (Bottom-right) positive temperature limit. Gray edges and states denotes them being rarely visited.

that notable. However, the figure highlights four other interesting regimes.

First, there is the $\beta \to \infty$ limit corresponding to the rare class with minimum decay rate equal to $u_{\min} = -\log_2(p^{\downarrow}_{\downarrow\downarrow}) = -\log_2(p^{\uparrow}_{\uparrow\uparrow})$. From Eq. (5.5) it is easy to see that this rare bubble only has two configurations as members: all up-spins or all down-spins. Let us consider finite but large $\beta \gg 1$ that corresponds to the rare class with a low energy density close to $u_{\min}$. Figure 5.10(top-left) shows a general ϵ-machine for this process. Low color intensity for both edges and states means that the process rarely visits them during generation. This means, in turn, that a typical realization consists of large blocks of all up-spins and all down-spins. These large blocks are joined by small segments.

Second, there is the $\beta \to -\infty$ limit that corresponds to the rare class with maximum

decay rate equal to $u_{\max} = -\frac{1}{2}\log_2(p^{\downarrow}_{\downarrow\uparrow} p^{\uparrow}_{\uparrow\downarrow})$. From Eq. (5.5) it is easy to see that this rare bubble only has one configuration as a member: a periodic repetition of spin down and spin up. Consider finite $\beta \ll 1$ corresponding to a rare class with a high energy density close to $u_{\max}$. Figure 5.10(top-right) shows the general ϵ-machine for the associated process. The typical configuration consists of large blocks tiled with spin-up and spin-down pairs that are connected by other short segments.

Third, there is the $\beta \to 0^+$ limit. In this limit we expect to see completely random spin-up/spin-down configurations. Figure 5.10(bottom-right) shows the ϵ-machine for this class labeled with nonzero small β. The transition probability for the edges labeled $+$ is $1/2 + \epsilon$ and for the edges labeled $-$ is $1/2 - \epsilon$, where ϵ is a small positive number. As one can see, even though each transition probability is close to one-half, the self-loops are slightly favored.

Fourth and finally, there is the $\beta \to 0^-$ limit. The generator here, Figure 5.10(bottom-left), is similar to that at positive infinite temperature, except that the edge-sign labels are reversed. This means that the self-loops are slightly less favored.

Appendix 5.10.4 explains the meaning of the β regimes and why each is important.

Remarkably, the memory ratio $\eta(u)$ diverges at $u = u_0 \approx 1.878$, where $u_0 = \lim_{\beta \to 0} u$—that is, at both the positive and negative high-temperature limit. Moreover, the memory ratio $\eta(u)$ diverges as $(u - u_0)^{-2}$ in both limits.

5.8.3 Extreme Quantum Memory Advantage for N-Nearest-Neighbor Spin Systems

Now, consider the *multi-process* setting in which we specify a series of stochastic processes labeled by an integer N that determines the size of each. In this, efficiency is defined by how the memory scales in N for the quantum algorithm compared to the classical.

We explore a general one-dimensional ferromagnetic N-nearest-neighbors Ising spin-$\frac{1}{2}$ chain defined by the Hamiltonian:

$$\mathcal{H}_N = -\sum_i \sum_{k=1}^{N} \frac{1}{k^\delta} s_i s_{i+k} \ ,$$

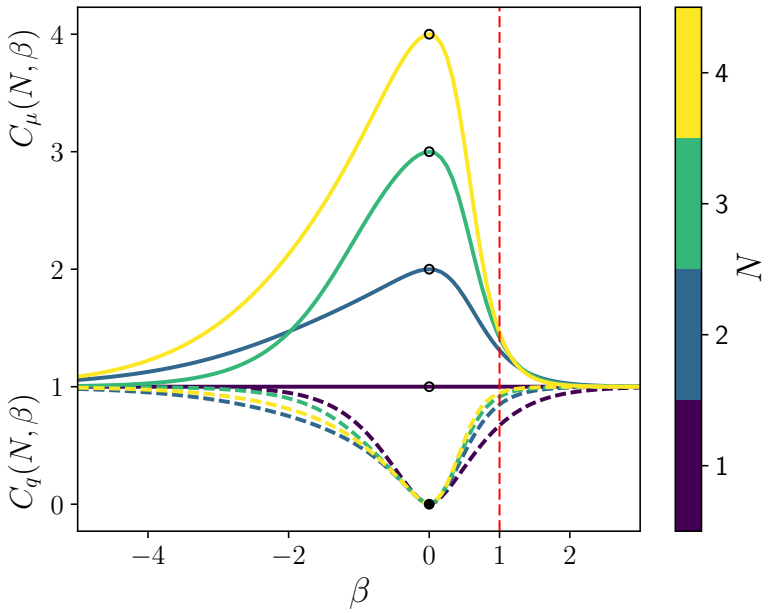


Figure 5.11: Classical memory $C_\mu(N,\beta)$ (solid lines) and quantum memory $C_q(N,\beta)$ (dashed lines) required for generating process $\mathcal{P}(N,\beta)$ for interaction ranges $N = 1,\ldots,4$, a range of $\beta \in [-5,3]$, and $\delta = 2$. At both limits $\beta \to 0^+$ and $\beta \to 0^-$, $C_\mu(N,\beta)$ scales linearly with N while $C_q(N,\beta)$ vanishes. For any finite β, for sufficiently large N, $C_\mu(N,\beta)$ is an increasing function of N, while $C_q(N,\beta)$ is bounded above by 1. The vertical dashed red line at $\beta = 1$ shows the memory costs for generating typical sequences.

in contact with a thermal bath at temperature $k_B T = 1$ and for which there is monopole-dipole coupling ($\delta = 2$).

As in the nearest-neighbor spin system, after thermalizing the probability over configurations at one instant of time defines a spatially-stationary stochastic process. We denote the process generated by this Hamiltonian by $\mathcal{P}(N)$. $\mathcal{P}(N)$ has Markov order $R = N$. More to the point, the ϵ-machine that generates this process has 2^N causal states and those states are in one-to-one correspondence with the set of length-N spin configurations. To determine the ϵ-machine transition probabilities $\{T^{(x)}\}_{x\in\mathcal{A}}$, one can use Ref. [1] and Ref. [14]'s appendix.

Let $\mathcal{P}(N,\beta)$ denote the process that typically generates the rare β-class of process $\mathcal{P}(N)$. Now, for an arbitrary fixed β, one can ask how the required classical memory $C_\mu(N,\beta)$ and quantum memory $C_q(N,\beta)$ for generating $\mathcal{P}(N,\beta)$ scales with N.

Figure 5.11 shows $C_\mu(N,\beta)$ and $C_q(N,\beta)$ versus β for different Ns. $C_\mu(.)$s are plotted

by solid-line and C_qs by dashed-line. To make them distinguishable, curves at different Ns are displayed with different colors. Interestingly, in both $\beta \to 0^+$ and $\beta \to 0^-$ limits, $C_\mu(N, \beta)$ scales linearly with N, while $C_q(N, \beta)$ goes to zero. More importantly, one can also check that for any finite nonzero β and sufficiently large N, $C_\mu(N, \beta)$ is an increasing function of N. Surprisingly, it can be shown that for any nonzero β and any N, $C_q(N, \beta)$ is bounded above by 1. The result is extreme quantum memory advantage for rare-event sampling of this series of stochastic processes.

5.9 Conclusions

We introduced a new quantum algorithm for simultaneous sampling rare events in classical stochastic processes. We showed that it confers a significant memory advantage when compared to the best known classical algorithm.

We explored two settings: single-process and multi-process sampling. For single processes an individual stochastic process is given and memory efficiency is defined as the ratio of memory required by the classical algorithm compared to that by the quantum one. For two example systems, we showed that for any large real number r there exist infinite classes of rare events for which the classical-quantum memory ratio for sampling is larger than r. In the multi-process setting, a series of stochastic processes each labeled by an integer size N is given. There, the memory efficiency is defined by how required memory scales in N for the classical algorithm compared to the quantum algorithm. In this setting we demonstrated an extreme quantum memory advantage in which the classical memory grows with N unboundedly, but the quantum memory is bounded.

5.10 Appendices

5.10.1 Completeness Condition

To establish Kraus operator completeness—$\sum_x K_x^\dagger K_x = I$— it is enough to show that, for all l and k:

$$\langle \xi_l | \left(\sum_x K_x^\dagger K_x \right) | \xi_k \rangle = \langle \xi_l | \xi_k \rangle \ .$$

Using the definition of Kraus operators we have:

$$\sum_x K_x^\dagger K_x = \sum_{\substack{i,j \\ m,n \\ x}} \sqrt{T_{mn}^{(x)}} \sqrt{T_{ij}^{(x)}} \, |\widetilde{\xi_m}\rangle \, \langle \xi_n | \xi_j \rangle \, \langle \widetilde{\xi_i}| \ .$$

As a result, for all l and k:

$$\begin{aligned} \langle \xi_l | \, (\sum_x K_x^\dagger K_x) \, |\xi_k\rangle &= \sum_{\substack{n,j \\ x}} \sqrt{T_{ln}^{(x)}} \sqrt{T_{kj}^{(x)}} \, \langle \xi_n | \xi_j \rangle \\ &= \langle \xi_l | \xi_k \rangle \ . \end{aligned}$$

This completes the proof.

5.10.2 Stationary Distribution

We calculate the stationary distribution over causal states. First, we have:

$$\begin{aligned} \sum_x K_x \rho_s K_x^\dagger &= \sum_{\substack{r,i,j \\ m,n \\ x}} \pi_r \sqrt{T_{mn}^x} \sqrt{T_{ij}^x} \, |\xi_j\rangle \, \langle \widetilde{\xi_i} | \xi_r \rangle \, \langle \xi_r | \widetilde{\xi_m} \rangle \, \langle \xi_n | \\ &= \sum_{\substack{n,j,r \\ x}} \pi_r \sqrt{T_{rn}^{(x)}} \sqrt{T_{rj}^{(x)}} \, |\xi_j\rangle \, \langle \xi_n | \ . \end{aligned}$$

Since ϵ-machines are unifilar [90], we always have: $\sqrt{T_{rn}^{(x)}} \sqrt{T_{rj}^{(x)}} = T_{rn}^x$. From this we find:

$$\sum_x K_x \rho_s K_x^\dagger = \sum_{\substack{n,r \\ x}} \pi_r T_{rn}^{(x)} \, |\xi_n\rangle \, \langle \xi_n | \ .$$

The stationary distribution over causal states satisfies $\pi T = \pi$ or, equivalently, $\pi_n = \sum_{x,r} \pi_r T_{rn}^{(x)}$. Replacing π_n in the above equation leads to:

$$\sum_x K_x \rho_s K_x^\dagger = \sum_n \pi_n \, |\xi_n\rangle \, \langle \xi_n | = \rho_s \ .$$

5.10.3 Perturbed Coins Process Quantum Machine

First step (Find the $|\eta_i\rangle$s):

Map causal states A and B to two pure quantum states $|\eta_A\rangle$ and $|\eta_B\rangle$:

$$\begin{aligned} |\eta_A\rangle &= \sqrt{p} \, |0\rangle + \sqrt{1-p} \, |1\rangle \\ |\eta_B\rangle &= \sqrt{1-q} \, |0\rangle + \sqrt{q} \, |1\rangle \ , \end{aligned}$$

where $|0\rangle$ and $|1\rangle$ form orthogonal bases on Hilbert spaces of size 2.

Second step (Find $|\xi_i\rangle$s and $|\widetilde{\xi_i}\rangle$s):

Since the size of alphabet $|\mathcal{A}| = 2$ and Markov order is $R = 1$ then $|\mathcal{A}|^R = 2$. The number of causal state is also $|\boldsymbol{\mathcal{S}}| = 2$. As a result $|\xi_A\rangle = |\eta_A\rangle$, $|\xi_B\rangle = |\eta_B\rangle$,

$$\Xi = \begin{bmatrix} |\xi_A\rangle & |\xi_B\rangle \end{bmatrix} = \begin{bmatrix} \sqrt{p} & \sqrt{1-q} \\ \sqrt{1-p} & \sqrt{q} \end{bmatrix} ,$$

and

$$\Xi^{-1} = \begin{bmatrix} \langle\widetilde{\xi_A}| \\ \langle\widetilde{\xi_B}| \end{bmatrix} = \frac{\begin{bmatrix} \sqrt{q} & -\sqrt{1-q} \\ -\sqrt{1-p} & \sqrt{p} \end{bmatrix}}{\sqrt{pq} - \sqrt{(1-p)(1-q)}} .$$

Third step (Find Kraus operators K_is):

$$\begin{aligned} K_0 &= \sqrt{p}\,|\xi_A\rangle\,\langle\widetilde{\xi_A}| + \sqrt{1-q}\,|\xi_A\rangle\,\langle\widetilde{\xi_B}| \\ &= \begin{bmatrix} \sqrt{p} & 0 \\ \sqrt{1-p} & 0 \end{bmatrix} \\ K_1 &= \sqrt{q}\,|\xi_B\rangle\,\langle\widetilde{\xi_B}| + \sqrt{1-p}\,|\xi_B\rangle\,\langle\widetilde{\xi_A}| \\ &= \begin{bmatrix} 0 & \sqrt{1-q} \\ 0 & \sqrt{q} \end{bmatrix} . \end{aligned}$$

We can easily check the completeness condition $K_0^\dagger K_0 + K_1^\dagger K_1 = I$.

Fourth step (Find stationary distribution ρ_s):

For the stationary state distribution over causal states we have $\pi T = \pi$, where:

$$T = \begin{bmatrix} p & 1-p \\ 1-q & q \end{bmatrix} .$$

As a result:

$$\pi = \begin{bmatrix} \pi_A \\ \pi_B \end{bmatrix} = \frac{1}{2-p-q} \begin{bmatrix} 1-q \\ 1-p \end{bmatrix} .$$

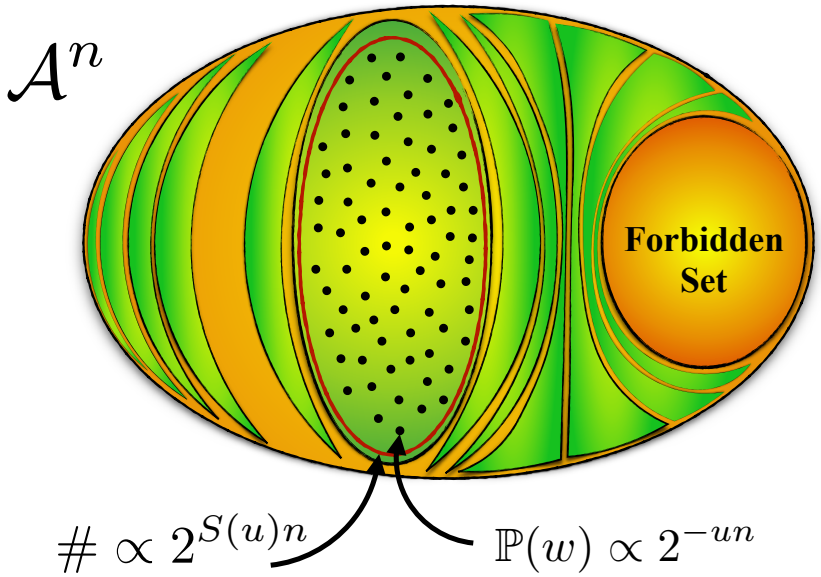


Figure 5.12: $\mathcal{A}^n$ the set of words with length n, can be decomposed to different bubbles $\Lambda^{\mathcal{P}}_{u,n}$ each labeled with different u. The number of words in the bubble with the energy density u is $\left|\Lambda^{\mathcal{P}}_{u,n}\right| = 2^{nS(u)}$ and the probability of each word in this bubble is 2^{-nu}.

Using $\rho_s = \pi_A \left|\xi_A\right\rangle \left\langle\xi_A\right| + \pi_B \left|\xi_B\right\rangle \left\langle\xi_B\right|$ we find:

$$\rho_s = \frac{1}{2-p-q} \begin{bmatrix} 1-q & \alpha \\ \alpha & 1-p \end{bmatrix},$$

where $\alpha = (1-q)\sqrt{p(1-p)} + (1-p)\sqrt{q(1-q)}$.

5.10.4 Meaning of the β Regimes

By Eq. (5.3), $\Lambda^{\mathcal{P}}_{u,n}$ is subset of all words with length n, $\Lambda^{\mathcal{P}}_{u,n} \subseteq \mathcal{A}^n$, whose probability decay rate $u = -\frac{1}{2}\log_2 \Pr(w)$. This partitions $\mathcal{A}^n$ into disjoint subsets $\Lambda^{\mathcal{P}}_{u,n}$ showed by green bubbles in Fig. 5.12. By definition, all the words in the bubble $\Lambda^{\mathcal{P}}_{u,n}$ have the same probabilities 2^{-nu}, as Fig. 5.12 shows.

One important question is how the probability of the whole bubble $\mathbb{P}(\Lambda^{\mathcal{P}}_{u,n})$ decays with n. This quantity can be written as a product of two factors: $\mathbb{P}(\Lambda^{\mathcal{P}}_{u,n}) = 2^{-nu}\left|\Lambda^{\mathcal{P}}_{u,n}\right|$. The first is the probability of each word in the bubble and the second the number of words in it.

Since $\left|\Lambda^{\mathcal{P}}_{u,n}\right|$ grows exponentially with n we define:

$$S(u) \equiv \lim_{n\to\infty} \frac{\log_2\left(\left|\Lambda^{\mathcal{P}}_{u,n}\right|\right)}{n} \ .$$

Now, for the probability of the bubble with energy density u one can write:

$$\mathbb{P}(\Lambda^{\mathcal{P}}_{u,n}) = 2^{-n(u-S(u))} \ .$$

The quantity $I(u) = u - S(u)$ often called the large deviation rate. $I(u)$ is always positive except for the typical set where it vanishes. This simply tells us that $\mathbb{P}(\Lambda^{\mathcal{P}}_{u,n})$, the probability of every bubble except the typical set, decays exponentially with n.

This provides the background needed to interpret the meaning behind different β regimes.

$\beta \ll -1$ **regime**: This regime includes bubbles with very high energy density u. Recalling that the probability of each word in the bubble with energy density u is 2^{-nu}, we see that this regime includes the bubble with the rarest words. Notice that this does not refer to the probability of an entire bubble, rather the probability of each word in the bubble.

$\beta \ll 1$ **regime**: This regimes includes bubbles with very low energy density u. Again, since the probability of each word in the bubble is 2^{-nu}, then this regime includes the bubble with the most probable words. To emphasize, this is the probability of each word in the bubble. For example, the bubble that has the most probable word can potentially have one member and still be very rare.

$\beta = 1$ **class**: This points to a particular bubble, the typical set.

$\beta \in (-\epsilon, \epsilon)$ **regime**: This includes bubbles with βs near zero. One can show that $\partial_u S(u(\beta)) = \beta$. As a result, the maximum of $S(u(\beta))$ happens at $\beta = 0$. Recalling the definition of $S(u)$, this means that the bubble with the maximum number of words is labeled by $\beta = 0$ or, equivalently, $u(\beta = 0)$). In other words, $\beta \in (-\epsilon, \epsilon)$ includes bubbles that have the largest number of words in them or, equivalently, largest $S(u)$.

Chapter 6

Thermodynamic Cost of Random Number Generation

6.1 Overview

We analyze the thermodynamic costs of the three main approaches to generating random numbers via the recently introduced Information Processing Second Law. Given access to a specified source of randomness, a random number generator (RNG) produces samples from a desired target probability distribution. This differs from pseudorandom number generators (PRNG) that use wholly deterministic algorithms and from true random number generators (TRNG) in which the randomness source is a physical system. For each class, we analyze the thermodynamics of generators based on algorithms implemented as finite-state machines, as these allow for direct bounds on the required physical resources. This establishes bounds on heat dissipation and work consumption during the operation of three main classes of RNG algorithms—including those of von Neumann, Knuth and Yao, and Roche and Hoshi—and for PRNG methods. We introduce a general TRNG and determine its thermodynamic costs exactly for arbitrary target distributions. The results highlight the significant differences between the three main approaches to random number generation: One is work producing, one is work consuming, and the other is potentially dissipation neutral. Notably, TRNGs can both generate random numbers and convert thermal energy to stored work. These thermodynamic costs on information creation complement Landauer's limit on the irreducible costs of information destruction.

6.2 Introduction

Random number generation is an essential tool these days in simulation and analysis. Applications range from statistical sampling [192], numerical simulation [193], cryptography [194], program validation [195], and numerical analysis [196] to machine learning [197] and decision making in games [198] and in politics [199]. More practically, a significant fraction of all the simulations done in physics [200] employ random numbers to greater or lesser extent.

Random number generation has a long history, full of deep design challenges and littered with pitfalls. Initially, printed tables of random digits were used for scientific work, first documented in 1927 [201]. A number of analog physical systems, such as reversed-biased Zener diodes [202] or even Lava® Lamps [203], were also employed as sources of randomness; the class of so-called *noise generators*. One of the first digital machines that generated random numbers was built in 1939 [204]. With the advent of digital computers, analog methods fell out of favor, displaced by a growing concentration on arithmetical methods that, running on deterministic digital computers, offered flexibility and reproducibility. An early popular approach to digital generation was the *linear congruential method* introduced in 1950 [205]. Since then many new arithmetical methods have been introduced [206–211].

The recurrent problem in all of these strategies is demonstrating that the numbers generated were, in fact, random. This concern eventually lead to Chaitin's and Kolmogorov's attempts to find an algorithmic foundation for probability theory [212–217]. Their answer was that an object is random if it cannot be compressed: random objects are their own minimal description. The theory exacts a heavy price, though: identifying randomness is uncomputable [216].

Despite the formal challenges, many physical systems appear to behave randomly. Unstable nuclear decay processes obey Poisson statistics [218], thermal noise obeys Gaussian statistics [219], cosmic background radiation exhibits a probabilistically fluctuating temperature field [220], quantum state measurement leads to stochastic outcomes [221–223], and fluid turbulence is governed by an underlying chaotic dynamic [224]. When such

physical systems are used to generate random numbers one speaks of *true random number generation* [225].

Generating random numbers without access to a source of randomness—that is, using arithmetical methods on a deterministic finite-state machine, whose logic is physically isolated—is referred to as *pseudorandom number generation*, since the numbers must eventually repeat and so, in principle, are not only not random, but are exactly predictable [226,227]. John von Neumann was rather decided about the pseudo-random distinction: "Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin" [228]. Nonetheless, these and related methods dominate today and perform well in many applications.

Sidestepping this concern by assuming a given source of randomness, *random number generation* (RNG) [229] is a complementary problem about the transformation of randomness: Given a specific randomness source, whose statistics are inadequate somehow, how can we convert it to a source that meets our needs? And, relatedly, how efficiently can this be done?

Our interest is not algorithmic efficiency, but thermodynamic efficiency, since any practical generation of random numbers must be physically embedded. What are the energetic costs—energy dissipation and power inputs—to harvest a given amount of information? This is a question, at root, about a particular kind of information processing—viz., information creation—and the demands it makes on its physical substrate. In this light, it should be seen as exactly complementary to Landauer's well known limit on the thermodynamic costs of information destruction (or erasure) [26,230].

Fortunately, there has been tremendous progress bridging information processing and the nonequilibrium thermodynamics required to support it [231,232]. This information thermodynamics addresses processes that range from the very small scale, such as the operation nanoscale devices and molecular dynamics [233], to the cosmologically large, such the character and evolution of black holes [234,235]. Recent technological innovations allowed many of the theoretical advances to be experimentally verified [236,237]. The current state of knowledge in this rapidly evolving arena is reviewed in Refs. [238–240].

Here, we use information thermodynamics to describe the physical limits on random number generation. Though the latter is often only treated as a purely abstract mathematical subject, practicing scientists and engineers know how essential random number generation is in their daily work. The following explores the underlying necessary thermodynamic resources.

First, Sec. 6.3 addresses random number generation, analyzing the thermodynamics of three algorithms, and discusses physical implementations. Second, removing the requirement of an input randomness source, Sec. 6.5 turns to analyze pseudorandom number generation and its costs. Third, Sec. 6.6 analyzes the thermodynamics of true random number generation. Finally, the conclusion compares the RNG strategies and their costs and suggests future problems and energy use.

6.3 Random Number Generation

Take a fair coin as our source of randomness.[1] Each flip results in a Head or a Tail with $50\% - 50\%$ probabilities. However, we need a coin that $1/4$ of the time generates Heads and $3/4$ of the time Tails. Can the series of fair coin flips be transformed? One strategy is to flip the coin twice. If the result is Head-Head, we report Heads. Else, we report Tails. The reported sequence is equivalent to flipping a coin with a bias $1/4$ for Heads and $3/4$ for Tails.

Each time we ask for a sample from the biased distribution we must flip the fair coin twice. Can we do better? The answer is yes. If the first flip results in a Tail, independent of the second flip's result, we should report Tail. We can take advantage of this by slightly modifying the original strategy. If the first flip results in a Tail, stop. Do not flip a second time, simply report a Tail, and start over. With this modification, $1/2$ of the time we need a single flip and $1/2$ the time we need two flips. And so, on average we need 1.5 flips to generate the distribution of interest. This strategy reduces the use of the fair coin "resource" by 25%.

Let's generalize. Assume we have access to a source of randomness that generates the

[1] Experiments reveal this assumption is difficult if not impossible to satisfy. Worse, if one takes the full dynamics into account, a flipped physical coin is quite predictable [241].

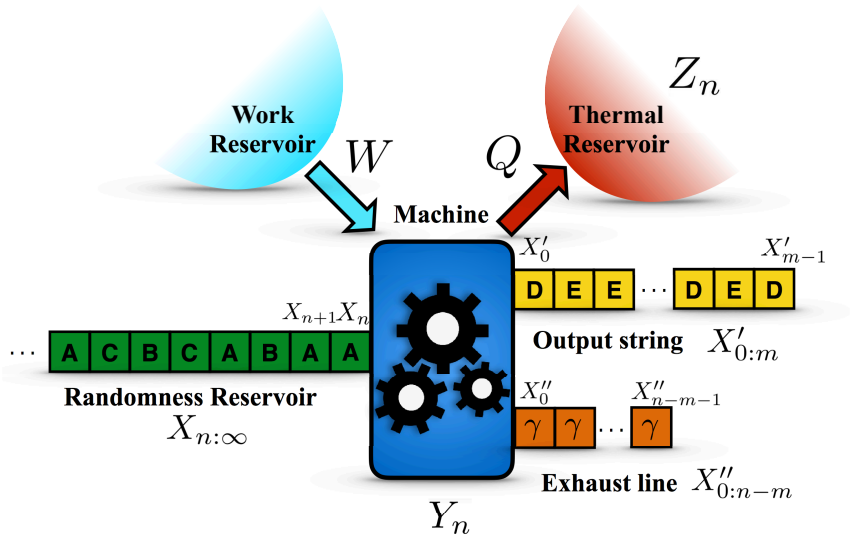


Figure 6.1: Thermodynamically embedded finite-state machine implementing an algorithm that, from the source of randomness available on the input string, generates random numbers on the output string obeying a desired target distribution and an exhaust with zero entropy. Input string and output string symbols can come from different alphabet sets. For example, here the input symbols come from the set $\{A, B, C\}$ and the outputs from $\{D, E\}$. Exhaust line symbols all are the same symbols γ.

distribution $\{p_i : i \in \mathcal{A}\}$ over discrete alphabet $\mathcal{A}$. We want an algorithm that generates another target distribution $\{q_j : j \in \mathcal{B}\}$ from samples of the given source. (Generally, the source of randomness $\{p_i\}$ can be known or unknown to us.) In this, we ask for a single correct sample from the target distribution. This is the *immediate random number generation problem*: Find an algorithm that minimizes the expected number of necessary samples of the given source to generate one sample of the target.[2]

The goal in the following is to analyze the thermodynamic costs when these algorithmically efficient algorithms are implemented in a physical substrate. This question parallels that posed by Landauer [26, 230]: What is the minimum thermodynamic cost to erase a bit of information? That is, rather than destroying information, we analyze the costs of creating information with desired statistical properties given a source of randomness.

[2] A companion is the *batch random number generation problem*: Instead of a single sample, generate a large number of inputs and outputs. The challenge is to find an algorithm minimizing the ratio of the number of inputs to outputs [242–244].

6.3.1 Bounding the Energetics:

The machine implementing the algorithm transforms symbols on an input string sampled from an information reservoir to an output symbol string and an exhaust string, using a finite-state machine that interacts with heat and work reservoirs; see Fig. 6.1. The input *Randomness Reservoir* is the given, specified source of randomness available to the RNG. The states and transition structure of the finite-state machine implement the RNG algorithm. The output string then consists of samples of the distribution of interest. The exhaust string is included to preserve state space.

Here, we assume inputs X_n are independent, identically distributed (IID) samples from the randomness reservoir with discrete alphabet $\mathcal{A}$. The output includes two strings, one with samples from the target distribution X'_m over alphabet $\mathcal{B}$ and another, the exhaust string. At each step one symbol, associated with variable X_n, enters the machine. After analyzing that symbol and, depending on its value and that of previous input symbols, the machine either writes a symbol to the output string or to the exhaust string. Y_n denotes the machine's state at step n after reading input symbol X_n. The last symbol in the output string after the input X_n is read is denoted X'_m, where $m \leq n$ is not necessarily equal to n. The last symbol in the exhaust string is X''_{n-m}. As a result, the number of input symbols read by the machine equals the number of symbols written to either the output string or the exhaust string. To guarantee that the exhaust makes no thermodynamic contribution, all symbols written to X''_is are the same—denoted γ. Without loss of generality we assume both the input and output sample space is $\mathcal{A} \cup \mathcal{B} \cup \{\gamma\}$. In the following we refer to the random-variable input chain as $X_{n:\infty} = X_n X_{n+1} \cdots X_\infty$, output chain as $X'_{0:m} = X'_0 X'_1 \cdots X'_{m-1}$, and exhaust chain as $X''_{0:n-m} = X''_0 X''_1 \cdots X''_{n-m-1}$.

The machine also interacts with an environment consisting of a *Thermal Reservoir* at temperature T and a *Work Reservoir*. The thermal reservoir is that part of the environment which contributes or absorbs heat, exchanging thermodynamic entropy and changing its state Z_n. The work reservoir is that part which contributes or absorbs energy by changing its state, but without an exchange of entropy. All transformations are performed isothermally at temperature T. As in Fig. 6.1, we denote heat that flows

to the thermal reservoir by Q. To emphasize, Q is positive if heat flows into the thermal reservoir. Similarly, W denotes the work done on the machine and not the work done by the machine.[3]

After n steps the machine has read n input symbols and generated m output symbols and $n - m$ exhaust symbols. The thermodynamic entropy change of the entire system is [246, App. A]:

$$\Delta S \equiv k_B \ln 2 \big(\mathrm{H}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty}, Y_n, Z_n] - \mathrm{H}[X_{0:\infty}, Y_0, Z_0] \big) \ ,$$

where $\mathrm{H}[\cdot]$ is the Shannon entropy [21]. Recalling the definition of mutual information $\mathrm{I}[\cdot : \cdot]$ [21], we rewrite the change in Shannon entropy on the righthand side as:

$$\begin{aligned} \Delta \mathrm{H} = & (\mathrm{H}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty}, Y_n] - \mathrm{H}[X_{0:\infty}, Y_0]) \\ & + (\mathrm{H}[Z_n] - \mathrm{H}[Z_0]) \\ & - (\mathrm{I}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty}, Y_n : Z_n] - \mathrm{I}[X_{0:\infty}, Y_0 : Z_0]) \ . \end{aligned}$$

By definition, a heat bath is not correlated with other subsystems, in particular, with portions of the environment. As a result, both mutual informations vanish. The term $\mathrm{H}[Z_n] - \mathrm{H}[Z_0]$ is the heat bath's entropy change, which can be written in terms of the dissipated heat Q:

$$\mathrm{H}[Z_n] - \mathrm{H}[Z_0] = \frac{Q}{k_B T \ln 2} \ .$$

Since by assumption the entire system is closed, the Second Law of Thermodynamics says that $\Delta S \geq 0$. Using these relations gives:

$$Q \geq -k_B T \ln 2 \big(\mathrm{H}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty}, Y_n] - \mathrm{H}[X_{0:\infty}, Y_0] \big) \ .$$

To use rates we divide both sides by n and decompose the first joint entropy:

$$\begin{aligned} \frac{Q}{n} \geq - \frac{k_B T \ln 2}{n} \big(& \mathrm{H}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty}] - \mathrm{H}[X_{0:\infty}] \\ & + \mathrm{H}[Y_n] - \mathrm{H}[Y_0] - \mathrm{I}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty} : Y_n] \\ & + \mathrm{I}[X_{0:\infty} : Y_0] \big) \ . \end{aligned}$$

[3] Several recent works [25, 245, 246] use the same convention for Q, but W is defined as the work done by the machine. This makes sense in those settings, since the machine is intended to do work.

Appealing to basic information identities, several terms on the right-hand side vanish, simplifying the overall bound. First, since the Shannon entropy of a random variable Y is bounded by logarithm of the size $|\mathcal{A}_Y|$ of its state space, we have for the machine's states:

$$\begin{aligned}\lim_{n\to\infty}\frac{1}{n}\,\mathrm{H}[Y_n] &= \lim_{n\to\infty}\frac{1}{n}\,\mathrm{H}[Y_0] \\ &\leq \lim_{n\to\infty}\frac{1}{n}\log_2|\mathcal{A}_Y| \\ &= 0\ ,\end{aligned}$$

Second, recalling that the two-variable mutual information is nonnegative and bounded above by the Shannon entropy of the individual random variables, in the limit $n\to\infty$ we can write:

$$\begin{aligned}\lim_{n\to\infty}\frac{1}{n}\,\mathrm{I}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty} : Y_n] &\leq \lim_{n\to\infty}\frac{1}{n}\,\mathrm{H}[Y_0] \\ &= 0\ .\end{aligned}$$

Similarly, $\lim_{n\to\infty}\frac{1}{n}\,\mathrm{I}[X_{0:\infty} : Y_0] = 0$. As a result, we have:

$$\lim_{n\to\infty}\frac{Q}{n} \geq -\frac{k_B T\ln 2}{n}\big(\,\mathrm{H}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty}] - \mathrm{H}[X_{0:\infty}]\big)\ .$$

We can also rewrite the joint entropy as:

$$\begin{aligned}\mathrm{H}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty}] &= \mathrm{H}[X'_{0:m}, X_{n:\infty}] + \mathrm{H}[X''_{0:n-m}] \\ &\quad - \mathrm{I}[X'_{0:m}, X_{n:\infty} : X''_{0:n-m}]\ .\end{aligned}$$

Since the entropy of the exhaust vanishes, $\mathrm{H}[X''_{0:n-m}] = 0$. Also, since $\mathrm{I}[X'_{0:m}, X_{n:\infty} : X''_{0:n-m}]$ is bounded above by it, $\mathrm{I}[X'_{0:m}, X_{n:\infty} : X''_{0:n-m}]$ also vanishes. This leads to:

$$\mathrm{H}[X''_{0:n-m}, X'_{0:m}, X_{n:\infty}] = \mathrm{H}[X'_{0:m}, X_{n:\infty}]\ .$$

This simplifies the lower bound on the heat to:

$$\lim_{n\to\infty}\frac{Q}{n} \geq -\frac{k_B T\ln 2}{n}\big(\,\mathrm{H}[X'_{0:m}, X_{n:\infty}] - \mathrm{H}[X_{0:\infty}]\big)\ .$$

Rewriting the righthand terms, we have:

$$\mathrm{H}[X_{0:\infty}] = \mathrm{H}[X_{0:n}] + \mathrm{H}[X_{n:\infty}] - \mathrm{I}[X_{0:n} : X_{n:\infty}]$$

and

$$\mathrm{H}[X'_{0:m}, X_{n:\infty}] = \mathrm{H}[X'_{0:m}] + \mathrm{H}[X_{n:\infty}] - \mathrm{I}[X'_{0:m} : X_{n:\infty}] \ .$$

These lead to:

$$\begin{aligned}\lim_{n\to\infty} \frac{Q}{n} \geq -\frac{k_B T \ln 2}{n} \big(& \mathrm{H}[X'_{0:m}] - \mathrm{H}[X_{0:n}] \\ & + \mathrm{I}[X_{0:n} : X_{n:\infty}] - \mathrm{I}[X'_{0:m} : X_{n:\infty}] \big) \ .\end{aligned}$$

Since the inputs are IID, $\mathrm{I}[X_{0:n} : X_{n:\infty}]$ vanishes. Finally, $\mathrm{I}[X'_{0:m} : X_{n:\infty}]$ is bounded above by $\mathrm{I}[X_{0:n} : X_{n:\infty}]$, meaning that $\mathrm{I}[X'_{0:m} : X_{n:\infty}] = 0$. Using these we have:

$$\lim_{n\to\infty} \frac{Q}{n} \geq \frac{k_B T \ln 2}{n} \big(\mathrm{H}[X_{0:n}] - \mathrm{H}[X'_{0:m}] \big) \ .$$

This can be written as:

$$\lim_{n\to\infty} \frac{Q}{n} \geq k_B T \ln 2 \left(\frac{\mathrm{H}[X_{0:n}]}{n} - \frac{\mathrm{H}[X'_{0:m}]}{m} \left(\frac{m}{n} \right) \right) \ .$$

As $n \to \infty$, $\mathrm{H}[X_{0:n}]/n$ converges to the randomness reservoir's Shannon entropy rate h and $\mathrm{H}[X'_{0:m}]/m$ converges to the output's entropy rate h'. The tapes' relative velocity term m/n also converges and we denote the limit as $1/\widehat{L}$. As a result, we have the rate $\dot{Q} \equiv \lim_{n\to\infty} (Q/n)$ of heat flow from the RNG machine to the heat bath:

$$\dot{Q} \geq k_B T \ln 2 \left(h - \frac{h'}{\widehat{L}} \right) \ . \tag{6.1}$$

Since the machine is finite state, its energy is bounded. In turn, this means the average energy entering the machine, above and beyond the constant amount that can be stored, is dissipated as heat. In other words, the average work rate $\dot{W}$ and average heat dissipation rate $\dot{Q}$ per input are equal: $\dot{W} = \dot{Q}$.

This already says something interesting. To generate one random number the average change ΔW in work done on the machine and the average change ΔQ in heat dissipation by the machine are directly related: $\Delta W = \Delta Q = \widehat{L}\dot{Q}$. More to the point, denoting the lower bound by $Q_{\mathrm{LB}} \equiv k_B T \ln 2 \left(\widehat{L} h - h' \right)$ immediately leads to a Second Law adapted to RNG thermodynamics:

$$\Delta Q \geq Q_{\mathrm{LB}} \ . \tag{6.2}$$

It can be shown that $\widehat{L}$ is always larger or equal to h'/h [21] and so $Q_{\rm LB} \geq 0$.[4] This tells us that RNG algorithms are always *heat dissipative* or, in other words, *work consuming* processes. Random numbers generated by RNGs cost energy. This new RNG Second Law allows the machine to take whatever time it needs to respond to and process an input. The generalization moves the information ratchet architecture [246] one step closer to that of general Turing machines [247], which also take arbitrary time to produce an output. We now apply this generalized Second Law to various physically embedded RNG algorithms.

6.3.2 von Neumann RNG:

Consider the case where the randomness resource is a biased coin with *unknown* probability $p \neq 1/2$ for Heads. How can we use this imperfect source to generate fair (unbiased $p = 1/2$) coin tosses using the minimum number of samples from the input? This problem was first posed by von Neumann [228]. The answer is simple but clever. What we need is a symmetry to undo the source's bias asymmetry. The strategy is to flip the biased coin twice. If the result is Heads-Tails we report a Head; if it is Tails-Heads we report Tails. If it is one of the two other cases, we neglect the flips and simply repeat from the beginning. A moment's reflection reveals that using any source of randomness that generates independent, identically distributed (IID) samples can be used in this way to produce a statistically uniform sample, even if we do not know the source's bias.

Note that we must flip the biased coin more than twice, perhaps many more, to generate an output. More troublesome, there is no bound on how many times we must flip to get a useful output.

So, what are the thermodynamic costs of this RNG scheme? With probability $2p(1-p)$ the first two flips lead to an output; with probability $(1-2p(1-p))(2p(1-p))$ the two flips do not, but the next two flips will; and so on. The expected number of flips to generate a fair coin output is $\widehat{L} = \frac{1}{p(1-p)}$. Using Eq. (6.2) this costs:

$$Q_{\rm LB} = k_B T \ln 2 \left(\frac{\mathrm{H}(p)}{p(1-p)} - 1 \right) \ , \tag{6.3}$$

[4] This is not generally true for the setup shown in Fig. 6.1 interpreted most broadly. For computational tasks more general than RNG, Q_{LB} need not be positive.

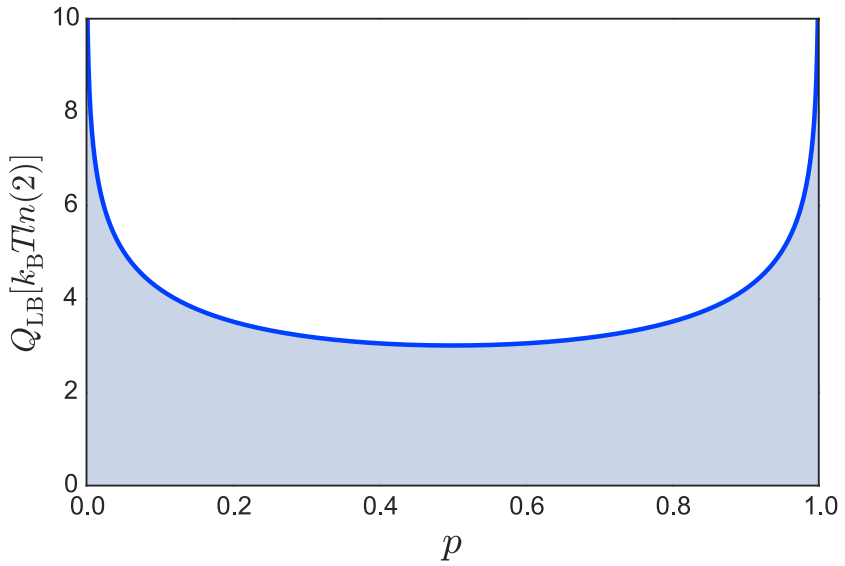


Figure 6.2: Lower bound on heat dissipation during the process of single fair sample generation by von Neumann algorithm versus the input bias p.

where $\mathrm{H}(p) = -p\log_2(p) - (1-p)\log_2(1-p)$. Figure 6.2 shows Q_{LB} versus source bias p. It is always positive with a minimum $3k_BT\ln 2$ at $p = 1/2$.

This minimum means that generating a fair coin from a fair coin has a heat cost of $3k_BT\ln 2$. At first glance, this seems wrong. Simply pass the fair coin through. The reason it is correct is that the von Neumann RNG does not know the input bias and, in particular, that it is fair. In turn, this means we may flip the coin many times, depending on the result of the flips, costing energy.

Notably, the bound diverges as $p \to 0$ and as $p \to 1$, since the RNG must flip an increasingly large number of times. As with all RNG methods, the positive lower bound implies that generating an unbiased sample via the von Neumann method is a *heat dissipative* process. We must put energy in to get randomness out.

Consider the *randomness extractor* [248], a variation on von Neumann RNG at extreme p, that uses a weakly random physical source but still generates a highly random output. (Examples of weakly random sources include radioactive decay, thermal noise, shot noise, radio noise, avalanche noise in Zener diodes, and the like. We return to physical randomness sources shortly.) For a weakly random source $p \ll 1$, the bound in Eq. (6.3) simplifies to $-k_BT\ln p$, which means heat dissipation diverges at least as fast as $-\ln p$ in the limit

$p \to 0$.

6.3.3 Knuth and Yao RNG:

Consider a scenario opposite von Neumann's where we have a fair coin and can flip it an unlimited number of times. How can we use it to generate samples from any desired distribution over a finite alphabet using the minimum number of samples from the input? Knuth and Yao were among the first to attempt an answer [249]. They proposed the *discrete distribution generation tree* (DDD-tree) algorithm.

The algorithm operates as follows. Say the target distribution is $\{p_j\}$ with probabilities p_j ordered from large to small. Define the partial sum $\beta_k = \sum_{j=1}^{k} p_j$, with $\beta_0 = 0$. This partitions the unit interval $(0, 1)$ into the subintervals (β_{k-1}, β_k) with lengths p_k. Now, start flipping the coin, denoting the outcomes $X_1, X_2, \ldots$. Let $S_l = \sum_{m=1}^{l} X_m 2^{-m}$. It can be easily shown that S_∞ has the uniform distribution over the unit interval. At any step l, when we flip the coin, we examine S_l. If there exists a k such that:

$$\beta_{k-1} \leq S_l < S_l + 2^{-l} \leq \beta_k \ , \tag{6.4}$$

the output generated is symbol k. If not, we flip the coin again for one or more times until we find a k that satisfies the relation in Eq. (6.4) and report that k as the output.

This turns on realizing that if the condition is satisfied, then the value of future flips does not matter since, for $r > l$, S_r always falls in the subinterval (β_{k-1}, β_k). Recalling that S_∞ is uniformly distributed over $(0, 1)$ establishes that the algorithm generates the desired distribution $\{p_j\}$. The algorithm can be also interpreted as walking a binary tree,[5] a view related to arithmetic coding [21]. Noting that the input has entropy rate $h = 1$ and using Eq. (6.1) the heat dissipation is bounded by:

$$Q_{\rm LB} = k_B T \ln 2 \left(\widehat{L} - \mathrm{H}[\{p_i\}] \right) \ . \tag{6.5}$$

Now, let's determine $\widehat{L}$ for the Knuth-Yao RNG. Ref. [249] showed that:

$$\mathrm{H}[\{p_i\}] \leq \widehat{L} \leq \mathrm{H}[\{p_i\}] + 2 \ . \tag{6.6}$$

[5] For details see Ref. [21].

Input	Output
00	A
01	B
10	C
110	B
1110	A
11110	A
111110	B
111111	C

Table 6.1: Most efficient map from inputs to outputs when using the DDG-tree RNG method.

More modern proofs are found in Refs. [244] and [21]. Given a general target distribution the Knuth-Yao RNG's $\widehat{L}$ can be estimated more accurately. However, it cannot be calculated in closed form, only bounded. Notably, there are distributions $\{p_j\}$ for which $\widehat{L}$ can be calculated exactly. These include the *dyadic distributions* whose probabilities can be written as 2^{-n} with n an integer. For these target distributions, the DDG-tree RNG has $\widehat{L} = \mathrm{H}[\{p_i\}]$.

Equations (6.2) and (6.6) lead one to conclude that the heat dissipation for generating one random sample is always a strictly positive quantity, except for the dyadic distributions which lead to vanishing or positive dissipation. Embedding the DDG-tree RNG into a physical machine, this means one must inject work to generate a random sample. The actual amount of work depends on the target distribution given.

Let us look at a particular example. Consider the case that our source of randomness is a fair coin with half and half probability over symbols 0 and 1 and we want to generate the target distribution $\{{}^{11}/_{32}, {}^{25}/_{64}, {}^{17}/_{64}\}$ over symbols A, B, and C. The target distribution has Shannon entropy $\mathrm{H}[\{p_i\}] \approx 1.567$ bits. Equation (6.6) tells us that $\widehat{L}$ should be larger than this. The DDG-tree method leads to the most efficient RNG. Table 6.1 gives the mapping from binary inputs to three-symbol outputs. $\widehat{L}$ can be calculated using the table:

$\widehat{L} \approx 2.469$. This is approximately 1 bit larger than the entropy consistent with Eq. (6.6). Now, using Eq. (6.5), we can bound the dissipated heat: $Q_{\rm LB} \approx 0.625 k_B T$.

6.3.4 Roche and Hoshi RNG:

A more sophisticated and more general RNG problem was posed by Roche in 1991 [250]: What if we have a so-called M-coin that generates the distribution $\{p_i : i = 1, \ldots, M\}$ and we want to use it to generate a different target distribution $\{q_j\}$? Roche's algorithm was probabilistic. And so, since we assume the only source of randomness to which we have access is the input samples themselves, Roche's approach will not be discussed here.

However, in 1995 Hoshi introduced a deterministic algorithm [251] from which we can determine the thermodynamic cost of this general RNG problem. Assume the p_is and q_js are ordered from large to small. Define $\alpha_t = \sum_{i=1}^{t} p_i$ and $\beta_k = \sum_{j=1}^{k} q_j$, with $\alpha_0 = \beta_0 = 0$. These quantities partition $(0, 1)$ into subintervals $[\alpha_{t-1}, \alpha_t)$ and $B_k \equiv [\beta_{k-1}, \beta_k)$ with lengths p_t and q_k, respectively. Consider now the operator $\mathcal{D}$ that takes two arguments—an interval and an integer—and outputs another interval:

$$\mathcal{D}([a, b), t) = [a + (b - a)\alpha_{t-1}, a + (b - a)\alpha_t) \ .$$

Hoshi's algorithm works as follows. Set $n = 0$ and $R_0 = [0, 1)$. Flip the M-coin, call the result x_n. Increase n by one and set $R_n = \mathcal{D}(R_{n-1}, x_n)$. If there is a k such that $R_n \subseteq B_k$, then report k, else flip the M-coin again.

Han and Hoshi showed that [251]:

$$\frac{\mathrm{H}[\{q_j\}]}{\mathrm{H}[\{p_i\}]} \leq \widehat{L} \leq \frac{\mathrm{H}[\{q_j\}] + f(\{p_i\})}{\mathrm{H}[\{p_i\}]} \ ,$$

where:

$$f(\{p_i\}) = \ln(2(M - 1)) + \frac{\mathrm{H}[\{p_{max}, 1 - p_{max}\}]}{1 - p_{max}} \ ,$$

with $p_{max} = \max_{i=1,\cdots,M} p_i$. Using this and Eq. (6.2) we see that the heat dissipation per sample is always positive except for measure-zero cases for which the dissipation may be zero or not. This means one must do work on the system independent of input and output distributions to generate the target sample. Again, using this result and Eq. (6.2) there exist input and output distributions with heat dissipation at least as large as $k_B T \ln 2 f(\{p_i\})$.

Input	Output
0	0
10	0
11	1

Table 6.2: Immediate random number generation: The most efficient map from inputs to output to transform fair coin inputs to biased coin outputs with bias 1/4.

6.4 RNG Physical Implementations:

Recall the first RNG we described. The input distribution is a fair coin and the output target distribution is a biased coin with bias 1/4. Table 6.2 summarizes the optimal algorithm. Generally, optimal algorithms require the input length to differ from the output length—larger than or equal, respectively.

This is the main challenge to designing physical implementations. Note that for some inputs, after they are read, the machine should wait for additional inputs until it receives the correct input and then transfers it deterministically to the output. For example, in our problem if input 0 is read, the output would be 0. However, if 1 is read, the machine should wait for the next input and then generate an output. How to implement these delays? Let's explore a chemical implementation of the algorithm.

Chemical reaction networks (CRNs) [252,253] have been widely considered as substrates for physical information processing [254] and as a programming model for engineering artificial systems [255,256]. Moreover, CRN chemical implementations have been studied in detail [257,258]. CRNs are also efficiently Turing-universal [259], which makes them appealing. One of their main applications is deterministic function computation [260,261], which is what our RNGs need.

Consider five particle types—0, 1, A, B, and γ—and a machine consisting of a box that can contain them. Particles 0 and 1 can be inputs to or outputs from the machine and particle γ can be an output from the machine. "Machine" particles A and B always stay in the machine's box and are in contact with a thermal reservoir. Figure 6.3 shows that the left wall is designed so that only input particles (0 and 1) can enter, but no particles

can exit. The right wall is designed so that only output particles (0, 1, and γ) can exit.

To get started, assume there is only a single machine particle A in the box. Every τ seconds a new input particle, 0 or 1, enters from the left. Now, the particles react in the following way:

$$\begin{aligned} 0 + A &\Rightarrow A + 0 \ , \\ 1 + A &\Rightarrow B \ , \\ 0 + B &\Rightarrow A + 0 + \gamma \ , \\ 1 + B &\Rightarrow A + 1 + \gamma \ . \end{aligned}$$

The time period of each chemical reaction is also τ. With this assumption it is not hard to show that if the distribution of input particles 0 and 1 is $\{1/2, 1/2\}$ then the distribution of output particles 0 and 1 would be $\{3/4, 1/4\}$, respectively. Thus, this CRN gives a physical implementation of our original RNG.

Using Eqs. (6.2) and (6.5) we can put a lower bound on the average heat dissipation per output: $Q_{\text{LB}} \approx 0.478 k_B T$. Since deriving the bound does not invoke any constraints over input or output particles, the bound is a universal lower bound over all possible reaction energetics. That is, if we find any four particles (molecules) obeying the four reactions above then the bound holds. Naturally, depending on the reactions' energetics, the CRN-RNG's ΔQ can be close to or far from the bound. Since CRNs are Turing-universal [259] they can implement all of the RNGs studied up to this point. The details of designing CRNs for a given RNG algorithm can be gleaned from the general procedures given in Ref. [260].

6.5 Pseudorandom Number Generation

So far, we abstained from von Neumann's sin by assuming a source of randomness—a fair coin, a biased coin, or any general IID process. Nevertheless, modern digital computers generate random numbers using purely deterministic arithmetical methods. This is *pseudorandom number generation* (PRNG). Can these methods be implemented by finite-state machines? Most certainly. The effective memory in these machines is very large,

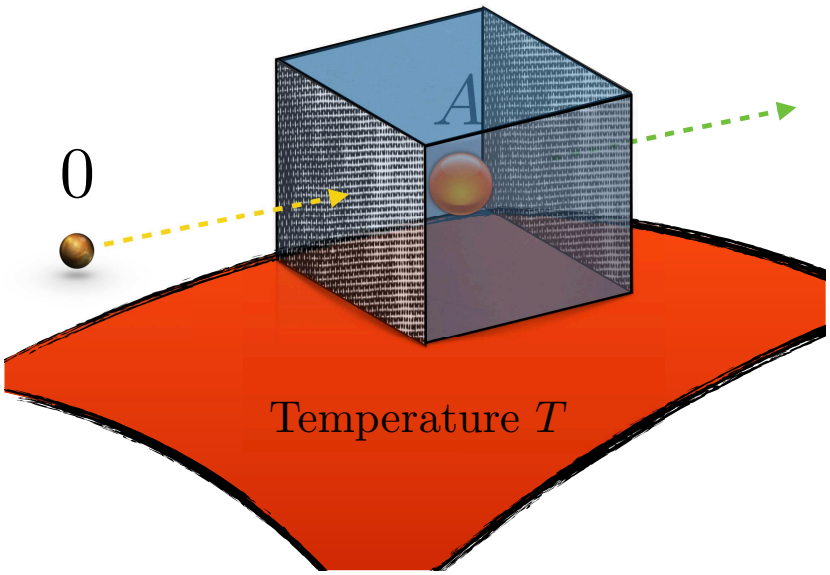


Figure 6.3: Chemical Reaction Network (CRN) implementation of an RNG machine consisting of a box and a particle in it. The left wall acts as a membrane filter such that only input particles, 0 and 1, can enter, but no particles can exit through the wall. The right wall is also a membrane designed such that only output particles, 0, 1 and γ, can exit. At the beginning the only particle in the box is "machine particle" A, which is confined to stay in the box. Every τ seconds a new input particle enters the box from the left and, depending on the reaction between the input particle and the machine particle, an output particle may or may not be generated that exists through the right wall.

with the algorithms typically allowing the user to specify the amount of state information used [262]. Indeed, they encourage the use of large amounts of state information, promising better quality random numbers in the sense that the recurrence time (generator period) is astronomically large. Our concern, though, is not analyzing their implementations. See Ref. [201] for a discussion of design methods. We can simply assume they can be implemented or, at least, there exist ones that have been, such as the Unix C-library `random()` function just cited.

The PRNG setting forces us to forego accessing a source of randomness. The input randomness reservoir is not random at all. Rather, it is simply a pulse that indicates that an output should be generated. Thus, $h = 0$ and $\widehat{L} = 1$. In our analysis, we can take the outputs to be samples of any desired IID process.

Even though a PRNG is supposed to generate a random number, in reality after setting the seed [226, 227] it, in fact, generates an exactly periodic sequence of outputs. Thus, as just noted, to be a good PRNG algorithm that period should be relatively long compared to the sample size of interest. Also, the sample statistics should be close to those of the desired distribution. This means that if we estimate h' from the sample it should be close to the Shannon entropy rate of the target distribution. However, in reality $h' = 0$ since h' is a measure over infinite-length samples, which in this case are completely nonrandom due to their periodicity.

This is a key point. When we use PRNGs we are only concerned about samples with comparatively short lengths compared to the PRNG period. However, when determining PRNG thermodynamics we average over asymptotically large samples. As a result, we have $Q_{\text{LB}} = 0$ or, equivalently, $\Delta Q \geq 0$. And so, PRNGs are potentially heat dissipative processes. Depending on the PRNG algorithm, it may be possible to find machinery that achieves the lower bound (zero) or not. To date, no such PRNG implementations have been introduced.

Indeed, the relevant energetic cost bounds are dominated by the number of logically irreversible computation steps in the PRNG algorithm, following Landauer [26]. This, from a perusal of open source code for modern PRNGs, is quite high. However, this takes us far afield, given our focus on input-output thermodynamic processing costs.

6.6 True Random Number Generation

Consider situations in which no random information source is explicitly given as with RNGs and none is approximated algorithmically as with PRNGs. This places us in the domain of *true random number generator*s (TRNGs): randomness is naturally embedded in their substrate physics. For example, a spin one-half quantum particle oriented in the z^+ direction, but measured in x^+ and x^- directions, gives x^+ and x^- outcomes with $1/2$ and $1/2$ probabilities. More sophisticated random stochastic process generators employing quantum physics have been introduced recently [8, 10, 12–15]. TRNGs have also been based on chaotic lasers [263, 264], metastability in electronic circuits [265, 266], and electronic

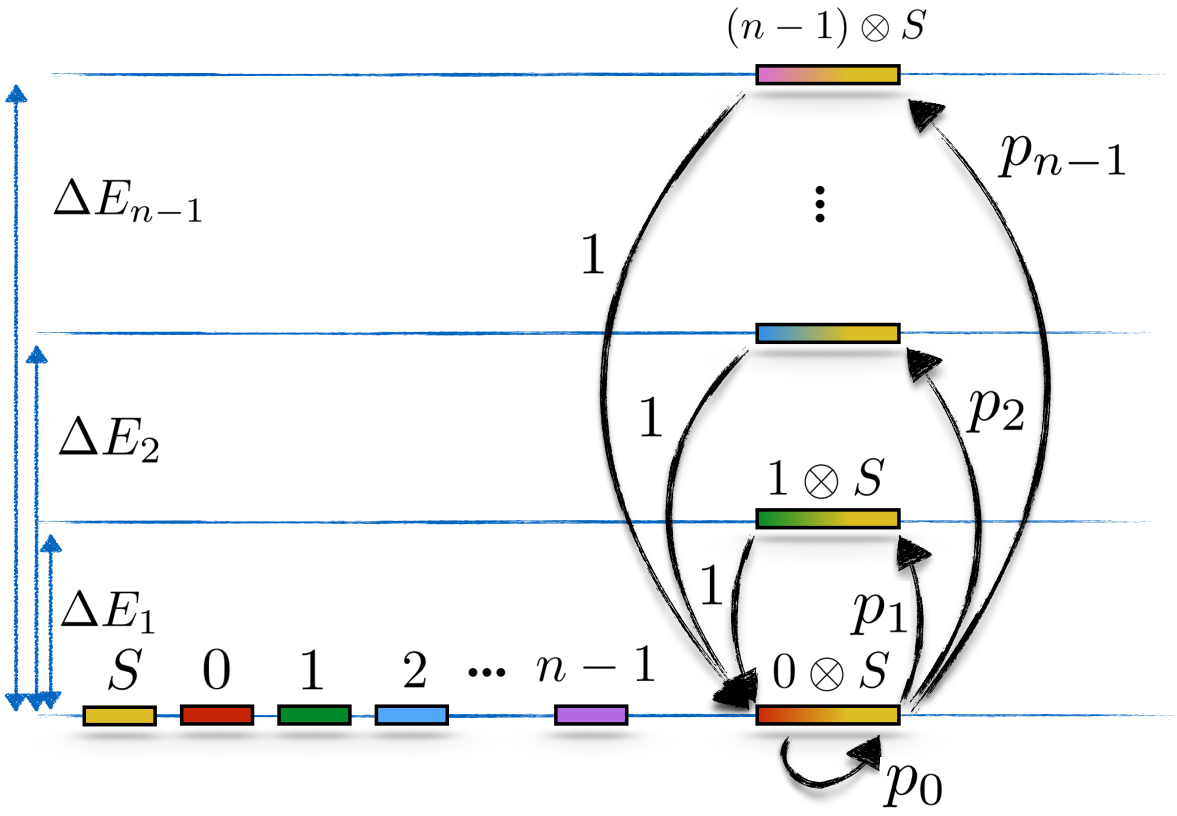


Figure 6.4: True general-distribution generator: Emit random samples from an arbitrary probability distribution $\{p_i\}$, $i = 0, \ldots, n-1$ where p_1 to p_{n-1} sorted from large to small. It has one internal state S and inputs and outputs can be 0, 1, ..., $n-1$. All states have energy zero. The joint states $i \otimes S$ for $i \neq 0$ have nonzero energies ΔE_i. Heat is transferred only during the transition from state $0 \otimes S$ to states $i \otimes S$. Work is transferred only during coupling the input bit to the machine's state and decoupling the output bit from the machine's state.

noise [267]. What thermodynamic resources do these TRNGs require? We address this here via one general construction.

6.6.1 True General-Distribution Generator:

Consider the general case where we want to generate a sample from an arbitrary probability distribution $\{p_i\}$. Each time we need a random sample, we feed in 0 and the TRNG returns a random sample. Again, the input is a long sequence of 0s and, as a consequence, $h = 0$. We also have $h' = \mathrm{H}[\{p_i\}]$ and $\widehat{L} = 1$. Equation (6.2) puts a bound on the dissipated heat

and input work: $Q_{\text{LB}} = -k_B T \ln 2 \, \text{H}[\{p_i\}]$. Notice here that Q_{LB} is a negative quantity. This is something that, as we showed above, can never happen for RNG algorithms since they all are heat-dissipation positive: $Q_{\text{LB}} > 0$. Of course, Q_{LB} is only a lower bound and ΔQ may still be positive. However, negative Q_{LB} opens the door to producing work from heat instead of turning heat to dissipated work—a functioning not possible for RNG algorithms.

Figure 6.4 shows one example of a physical implementation. The machine has a single state S and the inputs and outputs come from the symbol set $\{0, 1, \cdots, n-1\}$, all with zero energies. The system is designed so that the joint state $0 \otimes S$ has zero energy and the joint states $i \otimes S$, $i > 0$, have energy ΔE_i. Recall that every time we need a random sample we feed a 0 to the TRNG machine. Feeding 0 has no energy cost, since the sum of energies of states 0 and S is zero and equal to the energy of the state $0 \otimes S$. Then, putting the system into contact with a thermal reservoir, we have stochastic transitions between state $0 \otimes S$ and the other states $i \otimes S$. Tuning the $i \otimes S \to 0 \otimes S$ transition probabilities in a fixed time τ to 1 and assuming detailed balance, all the other transition probabilities are specified by the ΔE_is and, consequently, for all $i \in \{1, 2, \cdots, n-1\}$, we have $p_i = \exp(-\beta \Delta E_i)$.

The design has the system start in the joint state $0 \otimes S$ and after time τ with probability p_i it transitions to state $i \otimes S$. Then the average heat transferred from the system to the thermal reservoir is $-\sum_{i=1}^{n-1} p_i \Delta E_i$. Now, independent of the current state $i \otimes S$, we decouple the machine state S from the target state i. The average work we must pump into the system for this to occur is:

$$\Delta W = -\sum_{i=1}^{n-1} p_i \Delta E_i \ .$$

This completes the TRNG specification. In summary, the average heat ΔQ and the average work ΔW are the same and equal to $\sum_{i=1}^{n-1} p_i \Delta E_i$.

Replacing ΔE_i by $-k_B T \ln p_i$ we have:

$$\Delta Q = k_B T \sum_{i=1}^{n-1} p_i \ln p_i < 0 \ , \tag{6.7}$$

which is consistent with the lower bound $-k_B T \ln 2\, \mathrm{H}[\{p_i\}]$ given above. Though, as noted there, a negative lower bound does not mean that we can actually construct a machine with negative ΔQ, in fact, here is one example of such a machine. Negative ΔQ leads to an important physical consequence. The operation of a TRNG is a heat-consuming and work-producing process, in contrast to the operation of an RNG. This means not only are the random numbers we need being generated, but we also have an engine that absorbs heat from thermal reservoir and converts it to work. Of course, the amount of work depends on the distribution of interest. Thus, TRNGs are a potential win-win strategy. Imagine that at the end of charging a battery, one also had a fresh store of random numbers.

Let's pursue this further. For a given target distribution with n elements, we operate n such TRNG machines, all generating the distribution of interest. Any of the n elements of the given distribution can be assigned to the self-transition p_0. This gives freedom in our design to choose any of the elements. After choosing one, all the others are uniquely assigned to p_1 to p_{n-1} from largest to smallest. Now, if our goal is to pump-in less heat per sample, which of these machines is the most efficient? Looking closely at Eq. (6.7), we see that the amount of heat needed by machine j is proportional to $\mathrm{H}(\{p_i\}) - |p_j \log_2 p_j|$. And so, over all the machines, that with the maximum $|p_j \log_2 p_j|$ is the minimum-heat consumer and that with minimum $|p_j \log_2 p_j|$ is the maximum-work producer.

Naturally, there are alternatives to the thermodynamic transformations used in Fig. 6.4. One can use a method based on spontaneous irreversible relaxation. Or, one can use the approach of changing the Hamiltonian instantaneously and changing it back quasistatically and isothermally [232].

Let's close with a challenge. Now that a machine with negative ΔQ can be identified, we can go further and ask if there is a machine that actually achieves the lower bound Q_{LB}. If the answer is yes, then what is that machine? We leave the answer for the future.

6.7 Conclusion

Historically, three major approaches have been employed for immediate random number generation: RNG, PRNG, and TRNG. RNG itself divides into three interesting problems. First, when we have an IID source, but we have no knowledge of the source and the goal is to design machinery that generates an unbiased random number—the von Neumann RNG. Second, when we have a known IID source generating a uniform distribution and the goal is to invent a machine that can generate any distribution of interest—the Knuth and Yao RNG. Third, we have the general case of the second, when the randomness source is known but arbitrary and the goal is to devise a machine that generates another arbitrary distribution—the Roche and Hoshi RNG. For all these RNGs the overarching concern is to use the minimum number of samples from the input source. These approaches to random number generation may seem rather similar and to differ only in mathematical strategy and cleverness. However, the thermodynamic analyses show that they make rather different demands on their physical substrates, on the thermodynamic resources required.

We showed that all RNG algorithms are heat-consuming, work-consuming processes. In contrast, we showed that TRNG algorithms are heat-consuming, work-producing processes. And, PRNGs lie in between, dissipation neutral ($\Delta Q = 0$) in general and so the physical implementation determines the detailed thermodynamics. Depending on available resources and what costs we want to pay, the designer can choose between these three approaches.

The most thermodynamically efficient approach is TRNG since it generates both the random numbers of interest and converts heat that comes from the thermal reservoir to work. Implementing a TRNG, however, also needs a physical system with inherent stochastic dynamics that, on their own, can be inefficient depending on the resources needed. PRNG is the most unreliable method since it ultimately produces periodic sequences instead of real random numbers, but thermodynamically it potentially can be efficient. The RNG approach, though, can only be used given access to a randomness source. It is particularly useful if it has access to a nearly free randomness source. Thermodynamically, though, it is inefficient since the work reservoir must do work to run the machine, but the

resulting random numbers are reliable in contrast to those generated vis a PRNG.

To see how different the RNG and TRNG approaches can be, let's examine a particular example assuming access to a weakly random IID source with bias $p \ll 1$ and we want to generate an unbiased sample. We can ignore the randomness source and instead use the TRNG method with the machine in Fig. 6.4. Using Eq. (6.7) on average to produce one sample, the machine absorbs $|k_B T \; p \ln p| \approx 0$ heat from the heat reservoir and turns it into work. Since the required work is very small, this approach is resource neutral, meaning that there is no energy transfer between reservoir and machine. Now, consider the case when we use the RNG approach—the von Neumann algorithm. To run the machine and generate one symbol, on average the work reservoir needs to provide work energy to the machine. This thermodynamic cost can be infinitely large depending on how small p is. This comparison highlights how different the random number generation approaches can be and how their usefulness depends on available resources.

The thermodynamic analysis of the main RNG strategies suggests a number of challenges. Let's close with several brief questions that hint at several future directions in the thermodynamics of random number generation. Given that random number generation is such a critical and vital task in modern computing, following up on these strike us as quite important.

First, is Szilard's Engine [268] a TRNG? What are the thermodynamic costs in harvesting randomness? A recent analysis appears to have provided the answers [269] and anticipates TRNG's win-win property. Second, the randomness sources and target distributions considered were rather limited compared to the wide range of stochastic processes that arise in contemporary experiment and theory. For example, what about the thermodynamics of generating $1/f$ noise [270]? Nominally, this and other complex distributions are associated with infinite memory processes [271]. What are the associated thermodynamic cost bounds? Suggestively, it was recently shown that infinite-memory devices can actually achieve thermodynamic bounds [272]. Third, the random number generation strategies considered here are not secure. However, cryptographically secure random number generators have been developed [273]. What type of physical systems

can be used for secure TRNG and which are thermodynamically the most efficient? One possibility is to use superconducting nanowires and Josephson junctions tuned near where they generate superconducting critical currents [274]. Fourth, what are the additional thermodynamic costs of adding security to RNGs? Finally, there is a substantial advantage when employing quantum channels to compress classical random processes [12]. What are the thermodynamic consequences of using such quantum implementations for RNGs?

Let's close with several reflections on the results' practical impact. They could very well provide significant guidance in the near future, as we reduce the power consumption of computation for an energy-sustainable society. One can even argue they are significant now, as current technology strives to design ultra low-power devices and as the sciences attempt to understand information processing in biological process.

Consider the first—the total energy dissipated annually worldwide for computation. Total energy is directly related to the number of raw bit manipulations. The energy dissipated per bit manipulation arises from different sources, such as the operation of logic circuits, memory arrays, and communication interfaces. Currently for mainstream technology (e.g., CMOS), the average energy per one bit manipulation is close to $10^{-14}J$, which is referred as the *benchmark* [275]. It is also known that the computation volume (number of bit manipulations) increases exponentially every year [276]. These observations lead one to conclude that at the current benchmark energy dissipated per bit, global computing will not be sustainable by 2040, when the energy required for computing is projected to exceed the world's estimated energy production.

The conclusion is rather direct. We need a radical improvement in the energy efficiency of computing and, in particular, in random number generation which is a significant component in general computing. Random number generation is used heavily for many different tasks, much of it outside of the sciences and technology is found in security validation and secure communication and storage. Here, in analyzing the thermodynamic costs for alternative methods of random number generation, we showed that one method is work producing, one is work consuming, and the other is potentially dissipation neutral. In this way, the results highlight the basic physical trade-offs when implementing energy-

efficient random number generation. Hopefully, these will be useful guideposts when designing future computing infrastructure.

References

[1] D. P. Feldman and J. P. Crutchfield. Discovering non-critical organization: Statistical mechanical, information theoretic, and computational views of patterns in simple one-dimensional spin systems. Santa Fe Institute, 1998. Santa Fe Institute Paper 98-04-026. vii, 16, 19, 20, 29, 36, 87, 88, 92

[2] Whei Yeap Suen, Jayne Thompson, Andrew JP Garner, Vlatko Vedral, and Mile Gu. The classical-quantum divergence of complexity in modelling spin chains. *Quantum*, 1:25, 2017. vii, 6, 21, 22, 31, 34, 36

[3] N. F. Travers. *Bounds on Convergence of Entropy Rate Approximations in Hidden Markov Processes*. PhD thesis, University of California, Davis, 2013. 2, 16, 47, 70

[4] D. R. Upper. *Theory and Algorithms for Hidden Markov Models and Generalized Hidden Markov Models*. PhD thesis, University of California, Berkeley, 1997. 2, 3, 16, 47, 48, 62, 67, 70

[5] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, January, 1986. 3, 16, 48, 67

[6] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–286, 1989. 3, 16, 48, 67

[7] Ashley Montanaro. Quantum algorithms: an overview. *npj. Quantum. Inf.*, 2:15023, 2016. 6

[8] M. Gu, K. Wiesner, E. Rieper, and V. Vedral. Quantum mechanics can reduce the complexity of classical models. *Nat. Comm.*, 3:762, 2012. 6, 13, 20, 31, 33, 34, 38, 62, 68, 74, 85, 115

[9] Thomas J Elliott and Mile Gu. Superior memory efficiency of quantum devices for the simulation of continuous-time stochastic processes. *npj Quantum Information*, 4(1):18, 2018. 6, 74

[10] R. Tan, D. R. Terno, J. Thompson, V. Vedral, and M. Gu. Towards quantifying complexity with quantum mechanics. *Eur. Phys. J. Plus*, 129(9):1–12, 2014. 6, 22, 62, 115

[11] A. J. P. Garner, Q. Liu, J. Thompson, V. Vedral, and M. Gu. Provably unbounded memory advantage in stochastic simulation using quantum mechanics. *New J. Phys*, 2017. 6, 31, 74

[12] J. R. Mahoney, C. Aghamohammadi, and J. P. Crutchfield. Occam's quantum strop: Synchronizing and compressing classical cryptic processes via a quantum channel. *Sci. Rep.*, 6, 2016. 6, 8, 13, 21, 22, 31, 33, 61, 62, 68, 74, 115, 121

[13] P. M. Riechers, J. R. Mahoney, C. Aghamohammadi, and J. P. Crutchfield. Minimized state complexity of quantum-encoded cryptic processes. *Phys. Rev. A*, 93(5):052317, 2016. 6, 8, 13, 22, 31, 33, 62, 68, 74, 75, 76, 115

[14] C. Aghamohammdi, J. R. Mahoney, and J. P. Crutchfield. Extreme quantum advantage when simulating strongly coupled classical systems. *Sci. Rep.*, 7(6735):1–11, 2017. 6, 31, 62, 74, 88, 92, 115

[15] C. Aghamohammadi, J. R. Mahoney, and J. P. Crutchfield. The ambiguity of simplicity in quantum and classical simulation. *Phys. Lett. A.*, 381(14):1223–1227, 2017. 6, 22, 62, 74, 115

[16] C. Aghamohammadi, S. P. Loomis, J. R. Mahoney, and J. P. Crutchfield. Extreme quantum memory advantage for rare-event sampling. *Physical Review X*, 8(1):011025, 2018. 6

[17] M. S. Palsson, M. Gu, J. Ho, H. M. Wiseman, and G. J. Pryde. Experimentally modeling stochastic processes with less memory by the use of a quantum processor. *Sci. Adv.*, 3(2):e1601302, 2017. 6, 13, 22, 31, 68, 74

[18] F. G. Jouneghani, M. Gu, J. Ho, J. Thompson, W. Y. Suen, H. M. Wiseman, and G. J. Pryde. Observing the ambiguity of simplicity via quantum simulations of an ising spin chain. *arXiv preprint arXiv:1711.03661*, 2017. 6, 74

[19] J. Preskill. *Lecture notes for physics 229: Quantum information and computation*, volume 16. California Institute of Technology, Pasadena, California, 1998. 6, 35, 75, 77

[20] F. C. Binder, J. Thompson, and M. Gu. A practical, unitary simulator for non-Markovian complex processes. *Phys. Rev. Lett.*, 120(24):240502, 2018. 7, 8, 77

[21] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, second edition, 2006. 7, 17, 19, 32, 46, 50, 51, 52, 72, 79, 104, 107, 109, 110

[22] J. P. Crutchfield. Between order and chaos. *Nat. Phys.*, 8(1):17–24, 2012. 7, 18, 32, 51, 68, 73

[23] A. R. Leach. *Molecular modelling: Principles and applications*. Pearson Education, Boston, Massachusetts, 2001. 8, 67

[24] D. Frenkel and B. Smit. *Understanding Molecular Simulation: From Algorithms to Applications*. Academic Press, New York, second edition, 2007. 8, 67

[25] D. Mandal and C. Jarzynski. Work and information processing in a solvable model of Maxwell's demon. *Proc. Natl. Acad. Sci. USA*, 109(29):11641–11645, 2012. 9, 104

[26] R. Landauer. Irreversibility and heat generation in the computing process. *IBM J. Res. Develop.*, 5(3):183–191, 1961. 9, 100, 102, 115

[27] R. P. Feynman. Simulating physics with computers. *Intl. J. Theo. Phys.*, 21(6):467–488, 1982. 11

[28] K. Kihwan, M. S. Chang, S. Korenblit, R. Islam, E. E. Edwards, J. K. Freericks, G. D. Lin, L. M. Duan, and C. Monroe. Quantum simulation of frustrated Ising spins with trapped ions. *Nature*, 465(7298):590–593, 2010. 12

[29] R. Blatt and C. F. Roos. Quantum simulations with trapped ions. *Nat. Phys.*, 8(4):277–284, 2012. 12

[30] D. Jaksch and P. Zoller. The cold atom Hubbard toolbox. *Ann. Phys.*, 315(1):52–79, 2005. 12

[31] I. Bloch, J. Dalibard, and S. Nascimbene. Quantum simulations with ultracold quantum gases. *Nat. Phys.*, 8(4):267–276, 2012. 12

[32] J. Clarke and F. K. Wilhelm. Superconducting quantum bits. *Nature*, 453(7198):1031–1042, 2008. 12

[33] A. A. Houck, H. E. Türeci, and J. Koch. On-chip quantum simulation with superconducting circuits. *Nat. Phys.*, 8(4):292–299, 2012. 12

[34] B. P. Lanyon, J. D. Whitfield, G. G. Gillett, M. E. Goggin, M. P. Almeida, I. Kassal, J. D. Biamonte, M. Mohseni, B. J. Powell, and M. Barbieri. Towards quantum chemistry on a quantum computer. *Nat. Chem.*, 2(2):106–111, 2010. 12

[35] A. Aspuru-Guzik and P. Walther. Photonic quantum simulators. *Nat. Phys.*, 8(4):285–291, 2012. 12

[36] J. Du, N. Xu, X. Peng, P. Wang, S. Wu, and D. Lu. NMR implementation of a molecular hydrogen quantum simulation with adiabatic state preparation. *Phys. Rev. Lett.*, 104(3):030502, 2010. 12

[37] J. Zhang, M. H. Yung, R. Laflamme, A. Aspuru-Guzik, and J. Baugh. Digital quantum simulation of the statistical mechanics of a frustrated magnet. *Nat. Comm.*, 3:880, 2012. 12

[38] T. Byrnes, N. Y. Kim, K. Kusudo, and Y. Yamamoto. Quantum simulation of Fermi-Hubbard models in semiconductor quantum-dot arrays. *Phys. Rev. B.*, 78(7):075320, 2008. 12

[39] I. M. Georgescu, S. Ashhab, and F. Nori. Quantum simulation. *Rev. Mod. Phys.*, 86(1):153, 2014. 12

[40] D. A. Meyer. Quantum computing classical physics. *Phil. Trans. Roy. Soc. London A*, 360(1792):395–405, 2002. 12

[41] M. H. Yung, D. Nagaj, J. D. Whitfield, and A. Aspuru-Guzik. Simulation of classical thermal states on a quantum computer: A transfer-matrix approach. *Phys. Rev. A*, 82(6):060302, 2010. 12

[42] J. Yepez. Quantum lattice-gas model for computational fluid dynamics. *Phys. Rev. E*, 63(4):046702, 2001. 12

[43] J. Yepez. Quantum computation of fluid dynamics. In *Quantum Computing and Quantum Communications*, pages 34–60. Springer, 1999. 12

[44] S. Sinha and P. Russer. Quantum computing algorithm for electromagnetic field simulation. *Quant. Info. Proc.*, 9(3):385–404, 2010. 12

[45] J. Yepez. Quantum lattice-gas model for the diffusion equation. *Intl. J. Mod. Phys. C*, 12(09):1285–1303, 2001. 12

[46] G. P. Berman, A. A. Ezhov, D. I. Kamenev, and J. Yepez. Simulation of the diffusion equation on a type-ii quantum computer. *Phys. Rev. A*, 66(1):012310, 2002. 12

[47] J. Yepez. Quantum lattice-gas model for the burgers equation. *J. Stat. Phys*, 107(1-2):203–224, 2002. 12

[48] S. A. Harris and V. M. Kendon. Quantum-assisted biomolecular modelling. *Phil. Trans. Roy. Soc. London A*, 368(1924):3581–3592, 2010. 12

[49] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41(2):303–332, 1999. 12, 68

[50] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pages 212–219. ACM, 1996. 12, 68

[51] D. S. Abrams and S. Lloyd. Quantum algorithm providing exponential speed increase for finding eigenvalues and eigenvectors. *Phys. Rev. Lett.*, 83(24):5162, 1999. 12, 68

[52] A. W. Harrow, A. Hassidim, and S. Lloyd. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103(15):150502, 2009. 12, 68

[53] C. Pomerance. Fast, rigorous factorization and discrete logarithm algorithms. *Discrete Algo. Complexity*, pages 119–143, 1987. 12

[54] A. Bouland. Establishing quantum advantage. *XRDS: Crossroads, The ACM Magazine for Students*, 23(1):40–44, 2016. 12

[55] C. Aghamohammadi and J. P. Crutchfield. Minimum memory for generating rare events. *Phys. Rev. E.*, 95(3):032101, 2017. 12, 70, 82, 83

[56] J. P. Crutchfield and K. Young. Inferring statistical complexity. *Phys. Rev. Lett.*, 63:105–108, 1989. 12, 18, 31, 51, 68, 73

[57] W. Löhr and N. Ay. Non-sufficient memories that are sufficient for prediction. In *International Conference on Complex Sciences*, pages 265–276. Springer, 2009. 12, 19, 62

[58] W. Löhr and N. Ay. On the generative nature of prediction. *Adv. Complex Sys.*, 12(02):169–194, 2009. 12, 19, 62

[59] W. Löhr. Predictive models and generative complexity. *J. Sys. Sci. Complex.*, 25(1):30–45, 2012. 12, 19, 62

[60] A. Monras, A. Beige, and K. Wiesner. Hidden quantum Markov models and non-adaptive read-out of many-body states. *arXiv:1002.2337*, 2010. 12, 21, 31

[61] A. Monras and A. Winter. Quantum learning of classical stochastic processes: The completely positive realization problem. *J. Math. Phys.*, 57(1):015219, 2016. 12, 21

[62] C. J. Stark and A. W. Harrow. Compressibility of positive semidefinite factorizations and quantum models. *IEEE Trans. Info. Th.*, 62(5):2867–2880, 2016. 12

[63] F. J. Dyson. Existence of a phase-transition in a one-dimensional Ising ferromagnet. *Comm. Math. Phys.*, 12(2):91–107, 1969. 13, 14, 15, 27

[64] M. E. Fisher, S. K. Ma, and B. G. Nickel. Critical exponents for long-range interactions. *Phys. Rev. Lett.*, 29(14):917, 1972. 13, 15

[65] J. Fröhlich and T. Spencer. The phase transition in the one-dimensional Ising model with $1/r^2$ interaction energy. *Comm. Math. Phys.*, 84(1):87–101, 1982. 13, 15

[66] T. Blanchard, M. Picco, and M. A. Rajabpour. Influence of long-range interactions on the critical behavior of the Ising model. *Eur. Lett.*, 101(5):56003, 2013. 13, 15

[67] R. J. Baxter. *Exactly Solved Models in Statistical Mechanics*. Academic Press, 1989. 14, 40, 86

[68] A. Aghamohammadi, C. Aghamohammadi, and M. Khorrami. Externally driven one-dimensional Ising model. *J. Stat. Mech.*, 2012(02):P02004, 2012. 14, 86

[69] G. S. Rushbrooke and H. D. Ursell. On one-dimensional regular assemblies. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 44, pages 263–271. Cambridge University Press, 1948. 15

[70] J. Zhang, P. W. Hess, A. Kyprianidis, P. Becker, A. Lee, J. Smith, G. Pagano, I. D. Potirniche, A. C. Potter, A. Vishwanath, et al. Observation of a discrete time crystal. *Nature*, 543(7644):217, 2017. 15

[71] J. W. Britton, B. C. Sawyer, A. C. Keith, C. C. J. Wang, J. K. Freericks, H. Uys, M. J. Biercuk, and J. J. Bollinger. Engineered two-dimensional Ising interactions in a trapped-ion quantum simulator with hundreds of spins. *Nature*, 484(7395):489, 2012. 15

[72] P. Jurcevic, B. P. Lanyon, P. Hauke, C. Hempel, P. Zoller, R. Blatt, and C. F. Roos. Quasiparticle engineering and entanglement propagation in a quantum many-body system. *Nature*, 511(7508):202, 2014. 15

[73] R. Islam, C. Senko, W. C. Campbell, S. Korenblit, J. Smith, A. Lee, E. E. Edwards, C. C. J. Wang, J. K. Freericks, and C. Monroe. Emergence and frustration of magnetism with variable-range interactions in a quantum simulator. *Science*, 340(6132):583–587, 2013. 15

[74] J. Smith, Aaron. Lee, P. Richerme, B. Neyenhuis, P. W Hess, P. Hauke, M. Heyl, D. A. Huse, and C. Monroe. Many-body localization in a quantum simulator with programmable random disorder. *Nat. Phys.*, 12(10):907, 2016. 15

[75] J. P. Crutchfield and D. P. Feldman. Statistical complexity of simple one-dimensional spin systems. *Phys. Rev. E*, 55(2):R1239–R1243, 1997. 16

[76] J. R. Norris. *Markov Chains*, volume 2. Cambridge University Press, 1998. 16, 48, 67

[77] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2009. 16, 19, 47, 48

[78] B. Weiss. Subshifts of finite type and sofic systems. *Monastsh. Math.*, 77:462, 1973. 16

[79] J. P. Crutchfield. Semantics and thermodynamics. In *Nonlinear Modeling and Forecasting*, volume XII of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 317 – 359, Reading, Massachusetts, 1992. Addison-Wesley. 16

[80] J. P. Crutchfield, P. Riechers, and C. J. Ellison. Exact complexity: Spectral decomposition of intrinsic computation. *Phys. Lett. A*, 380(9-10):998–1002, 2016. 18, 19

[81] P. Gmeiner. Equality conditions for internal entropies of certain classical and quantum models. *arXiv:1108.5303*, 2011. 19, 21, 62

[82] N. Perry and P.-M. Binder. Finite statistical complexity for sofic systems. *Phys. Rev. E*, 60:459–463, 1999. 19

[83] J. Delgado and R. V. Solé. Collective-induced computation. *Phys. Rev. E*, 55:2338–2344, 1997. 19

[84] D. Nerukh, C. H. Jensen, and R. C. Glen. Identifying and correcting non-Markov states in peptide conformational dynamics. *J. Chem. Phys.*, 132(8):084104, 2010. 19

[85] D. Nerukh. Non-Markov state model of peptide dynamics. *J. Mole. Liquids*, 176:65–70, 2012. 19

[86] D. Kelly, M. Dillingham, A. Hudson, and K. Wiesner. A new method for inferring hidden Markov models from noisy time sequences. *PLoS One*, 7(1):e29703, 01 2012. 19, 51

[87] C. B. Li and T. Komatsuzaki. Aggregated Markov model using time series of a single molecule dwell times with a minimum of excessive information. *Phys. Rev. Lett.*, 111:058301, 2013. 19, 51

[88] D. P. Varn and J. P. Crutchfield. Chaotic crystallography: How the physics of information reveals structural order in materials. *Current Opinion Chem. Engin.*, 7:47–56, 2015. 19, 41

[89] J. R. Mahoney, C. J. Ellison, and J. P. Crutchfield. Information accessibility and cryptic processes. *J. Phys. A*, 42:362002, 2009. 19

[90] J. R. Mahoney, C. J. Ellison, R. G. James, and J. P. Crutchfield. How hidden are hidden processes? A primer on crypticity and entropy convergence. *CHAOS*, 21(3):037112, 2011. 19, 22, 94

[91] H. Dale, D. Jennings, and T. Rudolph. Provable quantum advantage in randomness processing. *Nat. Comm.*, 6, 2015. 27

[92] G. De las Cuevas and T. S. Cubitt. Simple universal models capture all classical spin physics. *Science*, 351(6278):1180–1183, 2016. 28

[93] J. F. Dobson. Many-neighbored Ising chain. *J. Math. Phys.*, 10(1):40–45, 1969. 28

[94] J. Preskill. Quantum computing and the entanglement frontier. *arXiv preprint arXiv:1203.5813*, 2012. 31, 40

[95] I. Boettcher, L. Bayha, D. Kedar, P. A. Murthy, M. Neidig, M. G. Ries, A. N. Wenz, G. Zurn, S. Jochim, and T. Enss. Equation of state of ultracold Fermions in the 2D BEC-BCS crossover region. *Phys. Rev. Lett*, 116:045303, 2016. 31

[96] K. Fenech, P. Dyke, T. Peppler, M. G. Lingham, S. Hoinka, H. Hu, , and C. J. Vale. Thermodynamics of an attractive 2D Fermi gas. *Phys. Rev. Lett.*, 116:045302, 2016. 31

[97] D. P. Feldman, C. S. McTague, and J. P. Crutchfield. The organization of intrinsic computation: Complexity-entropy diagrams and the diversity of natural information processing. *CHAOS*, 18(4):043106, 2008. 31

[98] P. W. Shor. *Algorithms for quantum computation: Discrete logarithms and factoring*. IEEE, 1994. 31

[99] D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. *Proc. Roy. Soc. Ser. A*, 439:553, 1992. 31

[100] C. Nayak, S. H. Simon, A. Stern, M. Freedman, and S. D. Sarma. Non-abelian anyons and topological quantum computation. *Rev. Mod. Phys*, 80(3):1083, 2008. 31

[101] A. N. Kolmogorov. A new metric invariant of transient dynamical systems and automorphisms in Lebesgue spaces. *Dokl. Akad. Nauk. SSSR*, 119:861, 1958. (Russian) Math. Rev. vol. 21, no. 2035a. 32

[102] J. P. Crutchfield and D. P. Feldman. Regularities unseen, randomness observed: Levels of entropy convergence. *CHAOS*, 13(1):25–54, 2003. 32, 48, 68

[103] N. F. Travers and J. P. Crutchfield. Exact synchronization for finite-state sources. *J. Stat. Phys.*, 145(5):1181–1201, 2011. 32

[104] R. G. James, J. R. Mahoney, C. J. Ellison, and J. P. Crutchfield. Many roads to synchrony: Natural time scales and their algorithms. *Phys. Rev. E*, 89:042135, Apr 2014. 33

[105] A. S. Holevo. Bounds for the quantity of information transmitted by a quantum communication channel. *Problems Inform. Trans.*, 9:177–183, 1973. 34, 38

[106] D. P. Feldman. *Computational Mechanics of Classical Spin Systems*. PhD thesis, University of California, Davis, 1998. Published by University Microfilms Intl, Ann Arbor, Michigan. 34, 42, 43

[107] M. A. Nielsen and I. Chuang. Quantum computation and quantum information, 2002. 35

[108] K. P. Burnham and D. R. Anderson. *Model selection and multimodel inference: A practical information-theoretic approach*. Springer Science & Business Media, 2003. 39

[109] C. C. Strelioff and J. P Crutchfield. Bayesian structural inference for hidden processes. *Phys. Rev. E*, 89(4):042119, 2014. 39

[110] H. Akaike. A new look at the statistical model identification. *IEEE Trans. Auto. Control*, 19(6):716–723, 1974. 39

[111] G. Schwarz. Estimating the dimension of a model. *Ann. Stat.*, 6(2):461–464, 1978. 39

[112] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*, volume 15. World Scientific, Singapore, 1998. 39

[113] C. S. Wallace and F. P. Freeman. Estimation and inference by compact coding. *J. R. Stat. Soc. B*, 49:240, 1987. 40

[114] D. A. Tennant, R. A. Cowley, S. E. Nagler, and A. M. Tsvelik. Measurement of the spin-excitation continuum in one-dimensional $KCuF_3$ using neutron scattering. *Phys. Rev. B*, 52:13368–13380, 1995. 41

[115] D. A. Tennant, S. E. Nagler, D. Welz, G. Shirane, and K. Yamada. Effects of coupling between chains on the magnetic excitation spectrum of $KCuF_3$. *Phys. Rev. B*, 52:13381–13389, 1995. 41

[116] J. M. Maillet. Heisenberg spin chains: from quantum groups to neutron scattering experiments. *Seminaire Poincare*, X:139–177, 2007. 41

[117] S. G. Brush. History of the Lenz-Ising model. *Rev. Mod. Phys.*, 39(4):883, 1967. 41

[118] R. Peierls. On Ising's model of ferromagnetism. *Math. Proc. Cambridge Phil. Soc.*, 32(03):477–481, 1936. 41

[119] S. F. Edwards and P. W. Anderson. Theory of spin glasses. *J. Phys. F: Metal Phys.*, 5(5):965, 1975. 41

[120] T. D. Lee and C. N. Yang. Statistical theory of equations of state and phase transitions. II. Lattice gas and Ising model. *Phys. Rev.*, 87(3):410, 1952. 41

[121] A. E. Noble, J. Machta, and A. Hastings. Emergent long-range synchronization of oscillating ecological populations without external forcing described by ising universality. *Nat. Comm.*, 6:6664, 2015. 41

[122] D. Sornette. Physics and financial economics (1776–2014): Puzzles, Ising, and agent-based models. *Rep. Prog. Phys.*, 77(6):062001, 2014. 41

[123] E. Schneidman, M. J. Berry, R. Segev, and W. Bialek. Weak pairwise correlations imply strongly correlated network states in a neural population. *Nature*, 440(7087):1007–1012, 2006. 41

[124] D. P. Feldman and J. P. Crutchfield. Structural information in two-dimensional patterns: Entropy convergence and excess entropy. *Phys. Rev. E*, 67(5):051103, 2003. 43

[125] J. P. Crutchfield. The hidden fragility of complex systems: Consequences of change, changing consequences. In P. Alsina and J. Perello, editors, *Cultures of Change — Changing Cultures*, pages 98–111, Barcelona, Spain, 2009. ACTAR Publishers. 46

[126] J. D. Deuschel and D. W. Stroock. *Large deviations*. Academic Press, New York, New York, 1989. 46

[127] J. A. Bucklew. *Large Deviation Techniques in Decision, Simulation, and Estimation*. Wiley-Interscience, New York, New York, 1990. 46

[128] K. Young and J. P. Crutchfield. Fluctuation spectroscopy. *Chaos Solitons Frac*, 4:5 – 39, 1994. 46, 57, 67

[129] H. Touchette. The large deviation approach to statistical mechanics. *Phys. Rep.*, 478:1–69, 2009. 46, 53, 81

[130] A. Dembo and O. Zeitouni. *Large Deviations Techniques and Applications*, volume 38 of *Stochastic Modelling and Applied Probability*. Springer, New York, New York, second edition, 2009. 46

[131] R. S. Ellis. *Entropy, Large Deviations, and Statistical Mechanics*, volume 271 of *A Series of Comprehensive Studies in Mathematics*. Springer, New York, New York, 2012. 46, 47

[132] P. H. Algoet and T. M. Cover. A sandwich proof of the Shannon-McMillan-Breiman theorem. *Ann. Prob.*, 16(2):899–909, 1988. 46

[133] M. D. Donsker and S. R. S. Varadhan. Asymptotic evaluation of certain Markov process expectations for large time. *Comm. Pure Appl. Math.*, 28:1–47, 1975. 47

[134] W. Feller. *An Introduction to Probability Theory and its Applications*. Wiley, New York, third, revised edition, 1970. 47

[135] A. Einstein. On the movement of small particles suspended in stationary liquids required by the molecular-kinetic theory of heat. *Ann. d. Phys.*, 17:549–560, 1905. 47

[136] A. Einstein. On the theory of Brownian motion. *Ann. d. Phys.*, 19:371–381, 1906. 47

[137] Y. Oono. Large deviation and statistical physics. *Prog. Theo. Phys.*, 99:165, 1989. 47

[138] O. E. Lanford. Entropy and equilibrium states in classical statistical mechanics. In *Statistical Mechanics and Mathematical Problems*, pages 1–113. Springer, 1973. 47

[139] D. Ruelle. *Thermodynamic Formalism*. Addison-Wesley, Reading, 1978. 47

[140] C. E. Shannon. A mathematical theory of communication. *Bell Sys. Tech. J.*, 27:379–423, 623–656, 1948. 51, 52, 79

[141] C. R. Shalizi and J. P. Crutchfield. Computational mechanics: Pattern and prediction, structure and simplicity. *J. Stat. Phys.*, 104:817–879, 2001. 51, 73, 89

[142] A. Witt, A. Neiman, and J. Kurths. Characterizing the dynamics of stochastic bistable systems by measures of complexity. *Phys. Rev. E*, 55:5050–5059, 1997. 51

[143] W. M. Gonçalves, R. D. Pinto, J. C. Sartorelli, and M. J. de Oliveira. Inferring statistical complexity in the dripping faucet experiment. *Physica A*, 257(1-4):385–389, 1998. 51

[144] A. J. Palmer, C. W. Fairall, and W. A. Brewer. Complexity in the atmosphere. *IEEE Trans. Geosci. Remote Sens.*, 38:2056–2063, 2000. 51

[145] R. W. Clarke, M. P. Freeman, and N. W. Watkins. The application of computational mechanics to the analysis of geomagnetic data. *Phys. Rev. E*, 67:160–203, 2003. 51

[146] D. Nerukh, C. H. Jensen, and R. C. Glen. Identifying and correcting non-Markov states in peptide conformational dynamics. *J. Chem. Phys.*, 132(8):084104, 2010. 51

[147] R. Durrett. *Probability: Theory and examples*. Cambridge University Press, Cambridge, UK, 2010. 52, 79

[148] S. Kullback. *Information Theory and Statistics*. Dover, New York, 1968. 52, 79

[149] R. W. Yeung. *Information Theory and Network Coding*. Springer, New York, 2008. 52, 79

[150] G. Han and B. Marcus. Analyticity of entropy rate of hidden Markov chains. *IEEE Trans. Info. Th.*, 52(12):5251–5266, 2006. 52, 74

[151] B. McMillan. The basic theorems of information theory. *Ann. Math. Stat.*, 24:196–219, 1953. 52, 79

[152] L. Breiman. The individual ergodic theorem of information theory. *Ann. Math. Stat.*, 28(3):809–811, 1957. 52, 79

[153] L. Boltzmann. *Lectures on Gas Theory*. Dover Publications, New York, 2013. 53, 80

[154] H. D. Miller. A convexity property in the theory of random variables defined on a finite Markov chain. *An. Math. Stat.*, 32(4):1260–1270, 1961. 57, 67

[155] J. P. Garrahan, R. L. Jack, V. Lecomte, E. Pitard, K. van Duijvendijk, and F. van Wijland. First-order dynamical phase transition in models of glasses: an approach based on ensembles of histories. *J. Phys. A: Math. Theo.*, 42(7):075007, 2009. 57, 67

[156] L. O. Hedges, R. L. Jack, J. P. Garrahan, and D. Chandler. Dynamic order-disorder in atomistic models of structural glass formers. *Science*, 323(5919):1309–1313, 2009. 57, 67

[157] R. L. Jack and P. Sollich. Large deviations and ensembles of trajectories in stochastic models. *Prog. Theo. Phys. Supp.*, 184:304–317, 2010. 57, 83

[158] J. Van Campenhout and T. M. Cover. Maximum entropy and conditional probability. *IEEE Trans. Info. Th.*, 27(4):483–489, 1981. 57, 67

[159] I. Csiszár. Sanov property, generalized I-projection and a conditional limit theorem. *Ann. Prob.*, 12(3):768–793, 1984. 57, 67

[160] J. P. Crutchfield and S. Marzen. Signatures of infinity: Nonergodicity and resource scaling in prediction, complexity, and learning. *Phys. Rev. E*, 91:050106(R), 2015. 60

[161] J. P. Crutchfield. The calculi of emergence: Computation, dynamics, and induction. *Physica D*, 75:11–54, 1994. 62

[162] J. P. Crutchfield and C. Aghamohammadi. Not all fluctuations are created equal: Spontaneous variations in thermodynamic function. *arXiv:1609.02519*, 2016. 62

[163] Y. Tsuji and T. Ishihara. Similarity scaling of pressure fluctuation in turbulence. *Phys. Rev. E*, 68:026309, 2003. 62

[164] T. Kuusela. Stochastic heart-rate model can reveal pathologic cardiac dynamics. *Phys. Rev. E*, 69:031916, 2004. 62

[165] J. Prusseit and K. Lehnertz. Stochastic qualifiers of epileptic brain dynamics. *Phys. Rev. Lett.*, 98:138103, 2007. 62

[166] M. Waechter, F. Riess, T. Schimmel, U. Wendt, and J. Peinke. Stochastic analysis of different rough surfaces. *Eur. Phys. J. B*, 41(2):259–277, 2004. 62

[167] A. L. S. Chua, C. A. Haselwandter, C. Baggio, and D. D. Vvedensky. Langevin equations for fluctuating surfaces. *Phys. Rev. E*, 72:051103, 2005. 62

[168] P. Sura. Stochastic analysis of Southern and Pacific Ocean sea surface winds. *J. Atmos. Sci.*, 60:654–666, 2003. 62

[169] C. Aghamohammadi, M. Ebrahimian, and H. Tahmooresi. Permutation approach, high frequency trading and variety of micro patterns in financial time series. *Physica A: Stat. Mech. App.*, 413:25–30, 2014. 62

[170] J. P. Huang. Experimental econophysics: Complexity, self-organization, and emergent properties. *Phys. Rep.*, 564:1–56, 2015. 62

[171] S. Kriso, J. Peinke, R. Friedrich, and P. Wagner. Reconstruction of dynamical equations for traffic flow. *Phys. Lett. A*, 299(2-3):287–291, 2002. 62

[172] T. Nagatani. Traffic jams induced by fluctuation of a leading car. *Phys. Rev. E*, 61:3534–3540, 2000. 62

[173] J. P. Crutchfield, C. J. Ellison, and J. R. Mahoney. Time's barbed arrow: Irreversibility, crypticity, and stored information. *Phys. Rev. Lett.*, 103(9):094101, 2009. 63

[174] N. Merhav, M. Gutman, and J. Ziv. On the estimation of the order of a Markov chain and universal data compression. *IEEE Trans. Info. Theo.*, 35(5):1014–1019, 1989. 63

[175] C. J. Ellison, J. R. Mahoney, and J. P. Crutchfield. Prediction, retrodiction, and the amount of information stored in the present. *J. Stat. Phys.*, 136(6):1005–1034, 2009. 63

[176] V. Lecomte, C. Appert-Rolland, and F. van Wijland. Chaotic properties of systems with Markov dynamics. *Phys. Rev. Lett.*, 95(1):010601, 2005. 67

[177] V. Lecomte, C. Appert-Rolland, and F. Van Wijland. Thermodynamic formalism for systems with markov dynamics. *J. Stat. Phys.*, 127(1):51–106, 2007. 67

[178] R. Chetrite and H. Touchette. Nonequilibrium microcanonical and canonical ensembles and their equivalence. *Phys. Rev. Lett.*, 111(12):120601, 2013. 67

[179] S. R. S. Varadhan. *Large deviations and applications*. SIAM, Philadelphia, Pennsylvannia, 1984. 67

[180] G. M. Torrie and J. P. Valleau. Nonphysical sampling distributions in monte carlo free-energy estimation: Umbrella sampling. *J. Comp. Phys.*, 23(2):187–199, 1977. 67

[181] S. Kumar, J. M. Rosenberg, D. Bouzida, R. H. Swendsen, and P. A. Kollman. The weighted histogram analysis method for free energy calculations on biomolecules. I. The method. *J. Comp. Chem.*, 13(8):1011–1021, 1992. 67

[182] F. Wang and D. P. Landau. Efficient, multiple-range random walk algorithm to calculate the density of states. *Phys. Rev. Lett.*, 86(10):2050, 2001. 67

[183] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, Providence, Rhode Island, 2009. 67

[184] S. Aaronson and A. Arkhipov. The computational complexity of linear optics. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 333–342. ACM, 2011. 69

[185] A. Ambainis, L. J. Schulman, A. Ta-Shma, U. Vazirani, and A. Wigderson. The quantum communication complexity of sampling. *SIAM Journal on Computing*, 32(6):1570–15857, 2003. 69

[186] M. Szegedy. Quantum speed-up of Markovchain based algorithms. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 32–41. IEEE, 2004. 69

[187] P. Wocjan and A. Abeyesinghe. Speedup via quantum sampling. *Phys. Rev. A*, 78(4):042336, 2008. 69

[188] A. J. P. Garner, J. Thompson, V. Vedral, and M. Gu. Thermodynamics of complexity and pattern manipulation. *Phys. Rev. E.*, 95(4):042140, 2017. 74

[189] C. Perry, R. Jain, and J. Oppenheim. Communication tasks with infinite quantum-classical separation. *Phys. Rev. Lett.*, 115(3):030504, 2015. 77

[190] J. P. Garrahan and I. Lesanovsky. Thermodynamics of quantum jump trajectories. *Phys. Rev. Lett.*, 104(16):160601, 2010. 83

[191] R. Chetrite and H. Touchette. Nonequilibrium Markov processes conditioned on large deviations. *Annales Henri Poincaré*, 16(9):2005–2057, 2015. 83

[192] W. G. Cochran. *Sampling Techniques*. John Wiley & Sons, 2007. 99

[193] B. Jerry. *Discrete-event System Simulation*. Pearson Education India, 1984. 99

[194] D. R. Stinson. *Cryptography: Theory and Practice*. CRC press, 2005. 99

[195] R. G. Sargent. Verification and validation of simulation models. In *Proceedings of the 37th conference on Winter simulation*, pages 130–143. winter simulation conference, 2005. 99

[196] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*, volume 12. Springer Science & Business Media, 2013. 99

[197] E. Alpaydin. *Introduction to Machine Learning*. MIT press, 2014. 99

[198] J. H. Conway. *On Numbers and Games*, volume 6. IMA, 1976. 99

[199] O. Dowlen. *The Political Potential of Sortition: A study of the random selection of citizens for public office*, volume 4. Andrews UK Limited, 2015. 99

[200] R. Y. Rubinstein and D. P. Kroese. *Simulation and the Monte Carlo method*, volume 707. John Wiley & Sons, 2011. 99

[201] D. E. Knuth. *The Art of Computer Programming: Semi-Numerical Algorithms*, volume 2. Addison-Wesley, Reading, Massachusetts, second edition, 1981. 99, 114

[202] C. D. Motchenbacher and F. C. Fitchen. *Low-Noise Electronic Design*. John Wiley & Sons, New York, 1973. 99

[203] B. Mende, L. C. Noll, and S. Sisodiya. SGI classic lavarand™. US Patent #5,732,138, 1996. 99

[204] M. G. Kendall and B. B. Smith. Randomness and random sampling numbers. *J. Roy. Stat. Soc.*, 101(1):147–166, 1938. 99

[205] D. H. Lehmer. Mathematical methods in large-scale computing units. In *Proc. 2nd Symp. on Large-Scale Digital Calculating Machinery*, pages 141–146. Harvard University Press, Cambridge, MA, 1951. 99

[206] B. A. Wichmann and I. D. Hill. Algorithm AS 183: An efficient and portable pseudo-random number generator. *J. Roy. Stat. Soc. Series C (Applied Statistics)*, 31(2):188–190, 1982. 99

[207] L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudo-random number generator. *SIAM J. Comput.*, 15(2):364–383, 1986. 99

[208] M. Mascagni, S. A. Cuccaro, D. V. Pryor, and M. L. Robinson. A fast, high quality, and reproducible parallel lagged-Fibonacci pseudorandom number generator. *J. Comput. Phys.*, 119(2):211–219, 1995. 99

[209] J. Kelsey, B. Schneier, and N. Ferguson. Yarrow-160: Notes on the design and analysis of the yarrow cryptographic pseudorandom number generator. In *International Workshop on Selected Areas in Cryptography*, pages 13–33. Springer, 1999. 99

[210] G. Marsaglia. Xorshift RNGs. *J. Stat. Soft.*, 8(14):1–6, 2003. 99

[211] J. K. Salmon, M. A. Moraes, R. O. Dror, and D. E. Shaw. Parallel random numbers: As easy as 1, 2, 3. In *2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, pages 1–12. IEEE, 2011. 99

[212] A. N. Kolmogorov. Three approaches to the concept of the amount of information. *Prob. Info. Trans.*, 1:1, 1965. 99

[213] G. Chaitin. On the length of programs for computing finite binary sequences. *J. ACM*, 13:145, 1966. 99

[214] P. Martin-Lof. The definition of random sequences. *Info. Control*, 9:602–619, 1966. 99

[215] L. A. Levin. Laws of information conservation (nongrowth) and aspects of the foundation of probability theory. *Problemy Peredachi Informatsii*, 10:30–35, 1974. Translation: Problems of Information Transmission **10** (1974) 206-210. 99

[216] M. Li and P. M. B. Vitanyi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer-Verlag, New York, 1993. 99

[217] A. N. Kolmogorov. Combinatorial foundations of information theory and the calculus of probabilities. *Russ. Math. Surveys*, 38:29–40, 1983. 99

[218] G. F. Knoll. *Radiation Detection and Measurement*. John Wiley & Sons, 2010. 99

[219] W. B. Davenport and W. L. Root. *Random Signals and Noise*. McGraw-Hill New York, 1958. 99

[220] N. Yoshida, R. K. Sheth, and A. Diaferio. Non-Gaussian cosmic microwave background temperature fluctuations from peculiar velocities of clusters. *Monthly Notices Roy. Astro. Soc.*, 328(2):669–677, 2001. 99

[221] T. Jennewein, U. Achleitner, G. Weihs, H. Weinfurter, and A. Zeilinger. A fast and compact quantum random number generator. *Rev. Sci. Instr.*, 71(4):1675–1680, 2000. 99

[222] A. Stefanov, N. Gisin, O. Guinnard, L. Guinnard, and H. Zbinden. Optical quantum random number generator. *J. Mod. Optics*, 47(4):595–598, 2000. 99

[223] A. Acin and L. Masanes. Certified randomness in quantum physics. *Nature*, 540:213–219, 2016. 99

[224] A. Brandstater, J. Swift, Harry L. Swinney, A. Wolf, J. D. Farmer, E. Jen, and J. P. Crutchfield. Low-dimensional chaos in a hydrodynamic system. *Phys. Rev. Lett.*, 51:1442, 1983. 99

[225] M. Stipčević and K. Ç. Koç. True random number generators. In *Open Problems in Mathematics and Computational Science*, pages 275–315. Springer, 2014. 100

[226] J. E. Gentle. *Random Number Generation and Monte Carlo Methods*. Springer Science & Business Media, New York, 2013. 100, 115

[227] R. Y. Rubinstein and B. Melamed. *Modern Simulation and Modeling*, volume 7. Wiley New York, 1998. 100, 115

[228] J. V. Neumann. Various techniques used in connection with random digits. In *Notes by G. E. Forsythe*, volume 12, pages 36–38. National Bureau of Standards Applied Math Series, 1963. 100, 107

[229] L. Devroye. Sample-based non-uniform random variate generation. In *Proceedings of the 18th conference on Winter simulation*, pages 260–265. ACM, 1986. 100

[230] C. H. Bennett. Thermodynamics of computation - a review. *Intl. J. Theo. Phys.*, 21:905, 1982. 100, 102

[231] C. Jarzynski. Equalities and inequalities: irreversibility and the second law of thermodynamics at the nanoscale. *Ann. Rev. Cond. Matter Physics*, 2(1):329–351, 2011. 100

[232] J. M. R. Parrondo, J .M. Horowitz, and T. Sagawa. Thermodynamics of information. *Nat. Phys.*, 11(2):131–139, 2015. 100, 118

[233] T. Sagawa. Thermodynamics of information processing in small systems. *Prog. Theo. Phys*, 127:1–56, 2012. 100

[234] T. M. Fiola, J. Preskill, A. Strominger, and S. P. Trivedi. Black hole thermodynamics and information loss in two dimensions. *Phys. Rev. D*, 50(6):3987, 1994. 100

[235] S. Das. Black-hole thermodynamics: Entropy, information and beyond. *Pramana*, 63(4):797–815, 2004. 100

[236] A. Berut, A. Arakelyan, A. Petrosyan, S. Ciliberto, R. Dillenschneider, and E. Lutz. Experimental verification of Landauer's principle linking information and thermodynamics. *Nature*, 483:187, 2012. 100

[237] S. Toyabe, T. Sagawa, M. Ueda, E. Muneyuki, and M. Sano. Experimental demonstration of information-to-energy conversion and validation of the generalized Jarzynski equality. *Nat. Phys.*, 6:988–992, 2010. 100

[238] K. Maruyama, F. Nori, and V. Vedral. Colloquium: The physics of Maxwell's demon and information. *Rev. Mod. Phys.*, 81:1, 2009. 100

[239] K. Sekimoto. *Stochastic Energetics*, volume 799. Springer, New York, 2010. 100

[240] U. Seifert. Stochastic thermodynamics, fluctuation theorems and molecular machines. *Rep. Prog. Phys.*, 75(12):126001, 2012. 100

[241] P. Diaconis, S. Holmes, and R. Montgomery. Dynamical bias in the coin toss. *SIAM review*, 49(2):211–235, 2007. 101

[242] P. Elias. The efficient construction of an unbiased random sequence. *Ann. Math. Stat.*, pages 865–870, 1972. 102

[243] Y. Peres. Iterating von Neumann's procedure for extracting random bits. *Ann. Stat.*, 20(1):590–597, 1992. 102

[244] D. Romik. Sharp entropy bounds for discrete statistical simulation. *Stat. Prob. Lett.*, 42(3):219–227, 1999. 102, 110

[245] A. C. Barato and U. Seifert. An autonomous and reversible Maxwell's demon. *Eur. Lett.*, 101:60001, 2013. 104

[246] A. B. Boyd, D. Mandal, and J. P. Crutchfield. Identifying functional thermodynamics in autonomous Maxwellian ratchets. *New. J. Phys.*, 18:023049, 2016. 104, 107

[247] H. R. Lewis and C. H. Papadimitriou. *Elements of the Theory of Computation*. Prentice-Hall, Englewood Cliffs, N.J., second edition, 1998. 107

[248] L. Trevisan and S. Vadhan. Extracting randomness from samplable distributions. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium*, pages 32–42. IEEE, 2000. 108

[249] D. E. Knuth and A. C. Yao. The complexity of nonuniform random number generation. *Algorithms and Complexity: New directions and recent results*, pages 357–428, 1976. 109

[250] J. R. Roche. Efficient generation of random variables from biased coins. In *Information Theory, Proc. 1991 IEEE Intl. Symp.*, pages 169–169, 1991. 111

[251] M. Hoshi. Interval algorithm for random number generation. *IEEE Trans. Info. Th.*, 43(2):599–611, 1997. 111

[252] O. N. Temkin, A. V. Zeigarnik, and D. G. Bonchev. *Chemical reaction networks: A graph-theoretical approach*. CRC Press, 1996. 112

[253] M. Cook, D. Soloveichik, E. Winfree, and J. Bruck. Programmability of chemical reaction networks. In *Algorithmic Bioprocesses*, pages 543–584. Springer, 2009. 112

[254] H. Jiang, M. D. Riedel, and K. K. Parhi. Digital signal processing with molecular reactions. *IEEE Design and Testing of Computers*, 29(3):21–31, 2012. 112

[255] M. O. Magnasco. Chemical kinetics is Turing universal. *Phys. Rev. Lett.*, 78(6):1190, 1997. 112

[256] A. Hjelmfelt, E. D. Weinberger, and J. Ross. Chemical implementation of neural networks and Turing machines. *Proc. Natl. Acad. Sci. USA*, 88(24):10983–10987, 1991. 112

[257] L. Cardelli. Strand algebras for DNA computing. *Nat. Comput.*, 10(1):407–428, 2011. 112

[258] D. Soloveichik, G. Seelig, and E. Winfree. DNA as a universal substrate for chemical kinetics. *Proc. Natl. Acad. Sci.*, 107(12):5393–5398, 2010. 112

[259] D. Soloveichik, M. Cook, E. Winfree, and J. Bruck. Computation with finite stochastic chemical reaction networks. *Nat. Comput.*, 7(4):615–633, 2008. 112, 113

[260] H. Chen, D. Doty, and D. Soloveichik. Deterministic function computation with chemical reaction networks. *Nat. Comput.*, 13(4):517–534, 2014. 112, 113

[261] D. Doty and M. Hajiaghayi. Leaderless deterministic chemical reaction networks. *Nat. Comput.*, 14(2):213–223, 2015. 112

[262] Unix Berkeley Software Distribution. Random(3). BSD Library Functions Manual, 2016. 114

[263] A. Uchida, K. Amano, M. Inoue, K. Hirano, S. Naito, Hiroyuki Someya, Isao Oowada, T. Kurashige, M. Shiki, and S. Yoshimori. Fast physical random bit generation with chaotic semiconductor lasers. *Nat. Photonics.*, 2(12):728–732, 2008. 115

[264] I. Kanter, Y. Aviad, I. Reidler, E. Cohen, and M. Rosenbluh. An optical ultrafast random bit generator. *Nat. Photonics.*, 4(1):58–61, 2010. 115

[265] D. J. Kinniment and E. G. Chester. Design of an on-chip random number generator using metastability. In *Solid-State Circuits Conference, 2002. ESSCIRC 2002. Proceedings of the 28th European*, pages 595–598. IEEE, 2002. 115

[266] C. Tokunaga, D. Blaauw, and T. Mudge. True random number generator with a metastability-based quality control. *IEEE J. Solid-State Circuits*, 43(1):78–85, 2008. 115

[267] M. Epstein, L. Hars, R. Krasinski, M. Rosner, and H. Zheng. Design and implementation of a true random number generator based on digital circuit artifacts. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 152–165. Springer, 2003. 116

[268] L. Szilard. On the decrease of entropy in a thermodynamic system by the intervention of intelligent beings. *Z. Phys.*, 53:840–856, 1929. 120

[269] A. B. Boyd and J. P. Crutchfield. Maxwell demon dynamics: Deterministic chaos, the Szilard map, and the intelligence of thermodynamic systems. *Phys. Rev. Lett.*, 116:190601, 2016. 120

[270] W. H. Press. Flicker noises in astronomy and elsewhere. *Comments Astrophys*, 7(4):103–119, 1978. 120

[271] S. Marzen and J. P. Crutchfield. Statistical signatures of structural organization: The case of long memory in renewal processes. *Phys. Lett. A*, 380(17):1517–1525, 2016. 120

[272] A. B. Boyd, D. Mandal, and J. P. Crutchfield. Leveraging environmental correlations: The thermodynamics of requisite variety. *J. Stat. Phys.*, 167(6):1555–1585, 2016. 120

[273] C. Easttom. *Modern Cryptography: Applied Mathematics for Encryption and Information Security*. McGraw-Hill Education, New York, 2015. 120

[274] M. Foltyn and M. Zgirski. Gambling with superconducting fluctuations. *Phys. Rev. App.*, 4(2):024002, 2015. 121

[275] V. Zhirnov, R. Cavin, and L. Gammaitoni. *Minimum energy of computing, fundamental considerations*, chapter 7. InTech, 2014. 121

[276] M. Hilbert and P. López. The world's technological capacity to store, communicate, and compute information. *Science*, 332(6025):60–65, 2011. 121