Thermodynamics of random number generation

Cina Aghamohammadi and James P. Crutchfield

Phys. Rev. E **95**, 062139 — Published 30 June 2017

DOI: 10.1103/PhysRevE.95.062139

Santa Fe Institute Working Paper 2017-01-02
arxiv.org:1612.08459 [cond-mat.stat-mech]

Thermodynamics of Random Number Generation

Cina Aghamohammadi[*] and James P. Crutchfield[†]
Complexity Sciences Center and Department of Physics,
University of California at Davis, One Shields Avenue, Davis, CA 95616

(Dated: May 26, 2017)

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.

PACS numbers: 05.70.Ln 89.70.-a 05.20.-y 02.50.-r
Keywords: entropy rate, nonequilibrium steady state, Maxwell's Demon, Second Law of Thermodynamics, Landauer's Principle

I. INTRODUCTION

Random number generation is an essential tool these days in simulation and analysis. Applications range from statistical sampling [1], numerical simulation [2], cryptography [3], program validation [4], and numerical analysis [5] to machine learning [6] and decision making in games [7] and in politics [8]. More practically, a significant fraction of all the simulations done in physics [9] 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 [10]. A number of analog physical systems, such as reversed-biased Zener diodes [11] or even Lava® Lamps [12], 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 [13]. 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 [14]. Since then many new arithmetical methods have been introduced [15–20].

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 [21–26]. 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 [25].

Despite the formal challenges, many physical systems appear to behave randomly. Unstable nuclear decay processes obey Poisson statistics [27], thermal noise obeys Gaussian statistics [28], cosmic background radiation exhibits a probabilistically fluctuating temperature field [29], quantum state measurement leads to stochastic outcomes [30–32], and fluid turbulence is governed by an underlying chaotic dynamic [33]. When such physical systems are used to generate random numbers one speaks of *true random number generation* [34].

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 [35, 36]. 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" [37]. Nonetheless,

[*] caghamohammadi@ucdavis.edu
[†] chaos@ucdavis.edu

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) [38] 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) [39, 40].

Fortunately, there has been tremendous progress bridging information processing and the nonequilibrium thermodynamics required to support it [41, 42]. This information thermodynamics addresses processes that range from the very small scale, such as the operation nanoscale devices and molecular dynamics [43], to the cosmologically large, such the character and evolution of black holes [44, 45]. Recent technological innovations allowed many of the theoretical advances to be experimentally verified [46, 47]. The current state of knowledge in this rapidly evolving arena is reviewed in Refs. [48–50]. 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. II addresses random number generation, analyzing the thermodynamics of three algorithms, and discusses physical implementations. Second, removing the requirement of an input randomness source, Sec. III turns to analyze pseudorandom number generation and its costs. Third, Sec. IV 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.

II. 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 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 [39, 40]: 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.

Bounding the Energetics: The machine implementing the algorithm transforms symbols on an input string sampled from an information reservoir to an output symbol

[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 [51].

[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 [52–54].

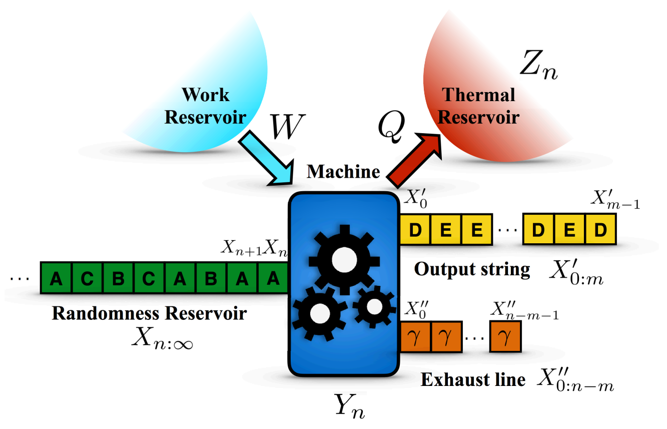

FIG. 1. Thermodynamically embedded finite-state machine implementing an algorithm that, from the source of randomness available on the input string, generates random numbers on the output string obeying a desired target distribution and an exhaust with zero entropy. Input string and output string symbols 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.

string and an exhaust string, using a finite-state machine that interacts with heat and work reservoirs; see Fig. 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''_i's 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. 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 [57, App. A]:

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

where $\mathrm{H}[\cdot]$ is the Shannon entropy [58]. Recalling the definition of mutual information $\mathrm{I}[\cdot : \cdot]$ [58], 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:

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

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

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

To use rates we divide both sides by n and decompose

[3] Several recent works [55–57] 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.

the first joint entropy:

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

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:

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

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:

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

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:

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

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:

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

This can be written as:

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

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:

$$\dot{Q} \geq k_{\mathrm{B}} T \ln 2 \left(h - \frac{h'}{\widehat{L}} \right) ~. \qquad (1)$$

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 $Q_{\mathrm{LB}} \equiv k_{\mathrm{B}} T \ln 2 \left(\widehat{L} h - h' \right)$ immediately leads to a Second Law adapted to RNG thermodynamics:

$$\Delta Q \geq Q_{\mathrm{LB}} ~. \qquad (2)$$

It can be shown that $\widehat{L}$ is always larger or equal to h'/h [58] and so $Q_{\mathrm{LB}} \geq 0$.[4] This tells us that RNG algorithms are always *heat dissipative* or, in other words, *work con-*

[4] This is not generally true for the setup shown in Fig. 1 interpreted most broadly. For computational tasks more general than RNG, Q_{LB} need not be positive.

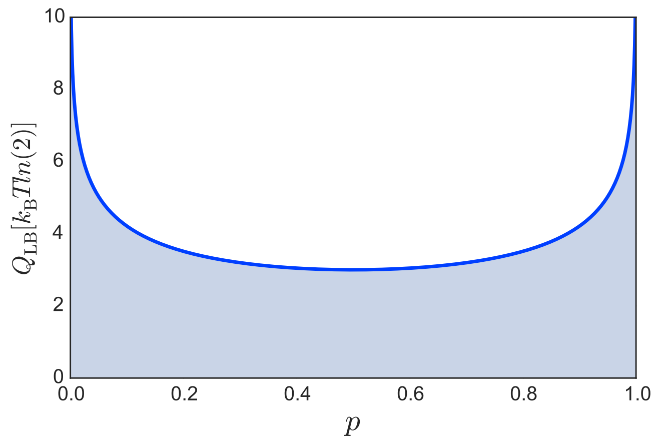

FIG. 2. Lower bound on heat dissipation during the process of single fair sample generation by von Neumann algorithm versus the input bias p.

suming 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 [57] one step closer to that of general Turing machines [59], which also take arbitrary time to produce an output. We now apply this generalized Second Law to various physically embedded RNG algorithms.

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 [37]. 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. (2) this costs:

$$Q_{\rm LB} = k_{\rm B}T \ln 2 \left(\frac{{\rm H}(p)}{p(1-p)} - 1 \right) ~, \qquad (3)$$

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

This minimum means that generating a fair coin from a fair coin has a heat cost of $3k_{\rm B}T \ln 2$. 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* [60], 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. (3) simplifies to $-k_{\rm B}T \ln p$, which means heat dissipation diverges at least as fast as $-\ln p$ in the limit $p \to 0$.

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 [61]. 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:

$$\beta_{k-1} \leq S_l < S_l + 2^{-l} \leq \beta_k ~, \qquad (4)$$

the output generated is symbol k. If not, we flip the coin

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

TABLE I. Most efficient map from inputs to outputs when using the DDG-tree RNG method.

again for one or more times until we find a k that satisfies the relation in Eq. (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 [58]. Noting that the input has entropy rate $h = 1$ and using Eq. (1) the heat dissipation is bounded by:

$$Q_{\rm LB} = k_{\rm B} T \ln 2 \left(\widehat{L} - {\rm H}[\{p_i\}] \right) . \tag{5}$$

Now, let's determine $\widehat{L}$ for the Knuth-Yao RNG. Ref. [61] showed that:

$${\rm H}[\{p_i\}] \leq \widehat{L} \leq {\rm H}[\{p_i\}] + 2 . \tag{6}$$

More modern proofs are found in Refs. [54] and [58]. 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 $\widehat{L} = {\rm H}[\{p_i\}]$.

Equations (2) and (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 ${\rm H}[\{p_i\}] \approx 1.567$ bits. Equation (6) tells us that $\widehat{L}$ should be larger than this. The DDG-tree method leads to the most efficient RNG. Table I 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). Now, using Eq. (5), we can bound the dissipated heat: $Q_{\rm LB} \approx 0.625 k_{\rm B} T$.

Roche and Hoshi RNG: A more sophisticated and more general RNG problem was posed by Roche in 1991 [62]: 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 [63] 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 [63]:

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

where:

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

with $p_{max} = \max_{i=1,\cdots,M} p_i$. Using this and Eq. (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. (2) there exist input and output distributions

[5] For details see Ref. [58].

Input	Output
0	0
10	0
11	1

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

with heat dissipation at least as large as $k_\mathrm{B}T \ln 2 f(\{p_i\})$.

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 II 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) [64, 65] have been widely considered as substrates for physical information processing [66] and as a programming model for engineering artificial systems [67, 68]. Moreover, CRN chemical implementations have been studied in detail [69, 70]. CRNs are also efficiently Turing-universal [71], which makes them appealing. One of their main applications is deterministic function computation [72, 73], 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 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

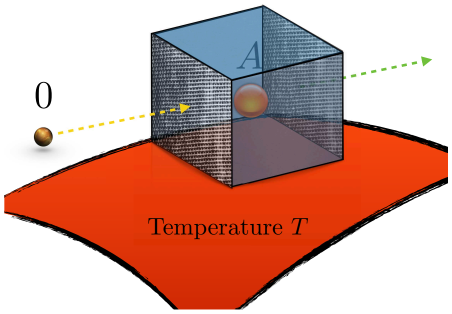

FIG. 3. Chemical Reaction Network (CRN) implementation of an RNG machine consisting of a box and a particle in it. The left wall acts as a membrane filter such that only input particles, 0 and 1, can enter, but no particles can exit through the wall. The right wall is also a membrane designed such that 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.

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. (2) and (5) we can put a lower bound on the average heat dissipation per output: $Q_\mathrm{LB} \approx 0.478 k_\mathrm{B}T$. 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 [71] 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. [72].

III. 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, with the algorithms typically allowing the user to specify the amount of state information used [74]. 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. [10] 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 [35, 36] 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 $Q_{\rm LB} = 0$ 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 [39]. 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.

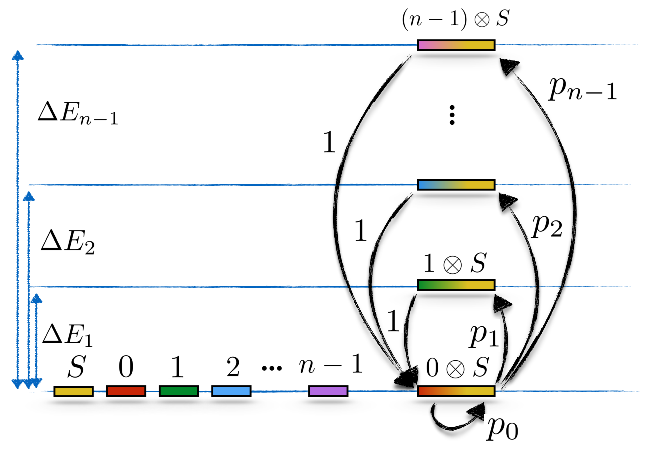

FIG. 4. True general-distribution generator: Emit random samples from an arbitrary probability distribution $\{p_i\}$, $i = 0, \ldots, n-1$ where p_1 to p_{n-1} sorted from large to small. It has one internal state S and inputs and outputs can be 0, 1, ..., $n-1$. All states have energy zero. The joint 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.

IV. 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 generators* (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 [75–80]. TRNGs have also been based on chaotic lasers [81, 82], metastability in electronic circuits [83, 84], and electronic noise [85]. What thermodynamic resources do these TRNGs require? We address this here via one general construction.

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 (2) puts a bound on the dissipated heat and input work: $Q_{\mathrm{LB}} = -k_{\mathrm{B}} T \ln 2\, \mathrm{H}[\{p_i\}]$. 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: $Q_{\mathrm{LB}} > 0$. 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 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 $-k_{\mathrm{B}} T \ln p_i$ we have:

$$\Delta Q = k_{\mathrm{B}} T \sum_{i=1}^{n-1} p_i \ln p_i < 0 \ , \qquad (7)$$

which is consistent with the lower bound $-k_{\mathrm{B}} T \ln 2\, \mathrm{H}[\{p_i\}]$ 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. (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. 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 [42].

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.

V. 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. 4. Using Eq. (7) on average to produce one sample, the machine absorbs $|k_{\mathrm{B}}T \; p \ln p| \approx 0$ 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 [86] a TRNG? What are the thermodynamic costs in harvesting randomness? A recent analysis appears to have provided the answers [87] 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 [88]? Nominally, this and other complex distributions are associated with infinite memory processes [89]. What are the associated thermodynamic cost bounds? Suggestively, it was recently shown that infinite-memory devices can actually achieve thermodynamic bounds [90]. Third, the random number generation strategies considered here are not secure. However, cryptographically secure random number generators have been developed [91]. 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 [92]. 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 [75]. 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* [93]. It is also known that the computation volume (number of bit manipulations) increases exponentially every year [94]. 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.

ACKNOWLEDGMENTS

We thank A. Aghamohammadi, M. Anvari, A. B. Boyd, R. G. James, M. Khorrami, J. R. Mahoney, and P. M. Riechers for helpful discussions. JPC thanks the Santa Fe Institute for its hospitality during visits as an External Faculty member. This material is based upon work supported by, or in part by, the John Templeton Foundation and U. S. Army Research Laboratory and the U. S. Army Research Office under contracts W911NF-13-1-0390 and W911NF-13-1-0340.

[1] W. G. Cochran. *Sampling Techniques.* John Wiley & Sons, 2007. 1

[2] B. Jerry. *Discrete-event System Simulation.* Pearson Education India, 1984. 1

[3] D. R. Stinson. *Cryptography: Theory and Practice.* CRC press, 2005. 1

[4] R. G. Sargent. Verification and validation of simulation models. In *Proceedings of the 37th conference on Winter simulation*, pages 130–143 1

[5] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*, volume 12 Springer Science & Business Media, 2013. 1

[6] E. Alpaydin. *Introduction to Machine Learning.* MIT press, 2014. 1

[7] J. H. Conway. *On Numbers and Games*, volume 6. IMA, 1976. 1

[8] O. Dowlen. *The Political Potential of Sortition: A study of the random selection of citizens for public office*, volume 4. Andrews UK Limited, 2015. 1

[9] R. Y. Rubinstein and D. P. Kroese. *Simulation and the Monte Carlo method*, volume 707. John Wiley & Sons, 2011. 1

[10] D. E. Knuth. *The Art of Computer Programming: Semi-Numerical Algorithms*, volume 2. Addison-Wesley, Reading, Massachusetts, second edition, 1981. 1, 8

[11] C. D. Motchenbacher and F. C. Fitchen. *Low-Noise Electronic Design.* John Wiley & Sons, New York, 1973. 1

[12] B. Mende, L. C. Noll, and S. Sisodiya. SGI classic lavarand™. US Patent #5,732,138, 1996. 1

[13] M. G. Kendall and B. B. Smith. Randomness and random sampling numbers. *J. Roy. Stat. Soc.*, 101(1):147–166, 1938. 1

[14] 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. 1

[15] 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. 1

[16] L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudo-random number generator. *SIAM J. Comput.*, 15(2):364–383, 1986.

[17] 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. Physics*, 119(2):211–219

[18] 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.

[19] G. Marsaglia. Xorshift RNGs. *J. Stat. Software*, 8(14):1–6, 2003.

[20] 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. 1

[21] A. N. Kolmogorov. Three approaches to the concept of the amount of information. *Prob. Info. Trans.*, 1:1, 1965. 1

[22] G. Chaitin. On the length of programs for computing finite binary sequences. *J. ACM*, 13:145, 1966.

[23] P. Martin-Lof. The definition of random sequences. *Info. Control*, 9:602–619, 1966.

[24] 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.

[25] M. Li and P. M. B. Vitanyi. *An Introduction to Kolmogorov Complexity and its Applications.* Springer-Verlag, New York, 1993. 1

[26] A. N. Kolmogorov. Combinatorial foundations of information theory and the calculus of probabilities. *Russ. Math. Surveys*, 38:29–40, 1983. 1

[27] G. F. Knoll. *Radiation Detection and Measurement.* John Wiley & Sons, 2010. 1

[28] W. B. Davenport and W. L. Root. *Random Signals and Noise*. McGraw-Hill New York, 1958. 1

[29] 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. 1

[30] 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. 1

[31] A. Stefanov, N. Gisin, O. Guinnard, L. Guinnard, and H. Zbinden. Optical quantum random number generator. *J. Mod. Optics*, 47(4):595–598, 2000.

[32] A. Acin and L. Masanes. Certified randomness in quantum physics. *Nature*, 540:213–219, 2016. 1

[33] 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. 1

[34] M. Stipčević and K. Ç. Koç. True random number generators. In *Open Problems in Mathematics and Computational Science*, pages 275–315. Springer, 2014. 1

[35] J. E. Gentle. *Random Number Generation and Monte Carlo Methods.* Springer Science & Business Media, New York, 2013. 1, 8

[36] R. Y. Rubinstein and B. Melamed. *Modern Simulation and Modeling*, volume 7. Wiley New York, 1998. 1, 8

[37] 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. 1, 5

[38] L. Devroye. Sample-based non-uniform random variate generation. In *Proceedings of the 18th conference on Winter simulation*, pages 260–265. ACM, 1986. 2

[39] R. Landauer. Irreversibility and heat generation in the computing process. *IBM J. Res. Develop.*, 5(3):183–191, 1961. 2, 8

[40] C. H. Bennett. Thermodynamics of computation - a review. *Intl. J. Theo. Phys.*, 21:905, 1982. 2

[41] 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. 2

[42] J. M. R. Parrondo, J .M. Horowitz, and T. Sagawa. Thermodynamics of information. *Nature Physics*, 11(2):131–139, 2015. 2, 9

[43] T. Sagawa. Thermodynamics of information processing in small systems. *Prog. Theo. Physics*, 127:1–56, 2012. 2

[44] 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. 2

[45] S. Das. Black-hole thermodynamics: Entropy, information and beyond. *Pramana*, 63(4):797–815, 2004. 2

[46] 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. 2

[47] 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. Physics*, 6:988–992, 2010. 2

[48] K. Maruyama, F. Nori, and V. Vedral. Colloquium: The physics of Maxwell's demon and information. *Rev. Mod. Physics*, 81:1, 2009. 2

[49] K. Sekimoto. *Stochastic Energetics*, volume 799. Springer, New York, 2010.

[50] U. Seifert. Stochastic thermodynamics, fluctuation theorems and molecular machines. *Reports Prog. Physics*, 75(12):126001, 2012. 2

[51] P. Diaconis, S. Holmes, and R. Montgomery. Dynamical bias in the coin toss. *SIAM Review*, 49(2):211–235, 2007. 2

[52] P. Elias. The efficient construction of an unbiased random sequence. *Ann. Math. Stat.*, pages 865–870, 1972. 2

[53] Y. Peres. Iterating von Neumann's procedure for extracting random bits. *Ann. Statistics*, 20(1):590–597, 1992.

[54] D. Romik. Sharp entropy bounds for discrete statistical simulation. *Stat. Prob. Lett.*, 42(3):219–227, 1999. 2, 6

[55] 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. 3

[56] A. C. Barato and U. Seifert. An autonomous and reversible Maxwell's demon. *Europhys. Lett.*, 101:60001, 2013.

[57] A. B. Boyd, D. Mandal, and J. P. Crutchfield. Identifying functional thermodynamics in autonomous Maxwellian ratchets. *New J. Physics*, 18:023049, 2016. 3, 5

[58] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley-Interscience, New York, second edition, 2006. 3, 4, 6

[59] H. R. Lewis and C. H. Papadimitriou. *Elements of the Theory of Computation.* Prentice-Hall, Englewood Cliffs, N.J., second edition, 1998. 5

[60] 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. 5

[61] 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. 5, 6

[62] J. R. Roche. Efficient generation of random variables from biased coins. In *Information Theory, Proc. 1991 IEEE Intl. Symp.*, pages 169–169, 1991. 6

[63] M. Hoshi. Interval algorithm for random number generation. *IEEE Trans. Info. Th.*, 43(2):599–611, 1997. 6

[64] O. N. Temkin, A. V. Zeigarnik, and D. G. Bonchev. *Chemical reaction networks: A graph-theoretical approach.* CRC Press, 1996. 7

[65] M. Cook, D. Soloveichik, E. Winfree, and J. Bruck. Programmability of chemical reaction networks. In *Algorithmic Bioprocesses*, pages 543–584. Springer, 2009. 7

[66] 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. 7

[67] M. O. Magnasco. Chemical kinetics is Turing universal. *Phys. Rev. Lett.*, 78(6):1190, 1997. 7

[68] 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. 7

[69] L. Cardelli. Strand algebras for DNA computing. *Natural Computing*, 10(1):407–428, 2011. 7

[70] D. Soloveichik, G. Seelig, and E. Winfree. DNA as a universal substrate for chemical kinetics. *Proc. Natl. Acad. Sci.*, 107(12):5393–5398, 2010. 7

[71] D. Soloveichik, M. Cook, E. Winfree, and J. Bruck. Computation with finite stochastic chemical reaction networks. *Natural Computing*, 7(4):615–633 7

[72] H. Chen, D. Doty, and D. Soloveichik. Deterministic function computation with chemical reaction networks. *Natural computing*, 13(4):517–534, 2014. 7

[73] D. Doty and M. Hajiaghayi. Leaderless deterministic chemical reaction networks. *Natural Computing*, 14(2):213–223, 2015. 7

[74] Unix Berkeley Software Distribution. Random(3). BSD Library Functions Manual, 2016. 8

[75] J. R. Mahoney, C. Aghamohammadi, and J. P. Crutchfield. Occam's quantum strop: Synchronizing and compressing classical cryptic processes via a quantum channel. *Scientific Reports*, 6:20495, 2016. 8, 10

[76] C. Aghamohammdi, J. R. Mahoney, and J. P. Crutchfield. Extreme quantum advantage when simulating strongly coupled classical. *Sci. Reports*, in press, 2017. arxiv.org:1609.03650.

[77] M. Gu, K. Wiesner, E. Rieper, and V. Vedral. Quantum mechanics can reduce the complexity of classical models. *Nature Comm.*, 3:762, 2012.

[78] R. Tan, D. R. Terno, J. Thompson, V. Vedral, and M. Gu. Towards quantifying complexity with quantum mechanics. *Euro. Phys. J. Plus*, 129(9):1–12

[79] C. Aghamohammadi, J. R. Mahoney, and J. P. Crutchfield. The ambiguity of simplicity. *Physics Lett. A*, in press, 2016. arXiv preprint arXiv:1602.08646.

[80] 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. 8

[81] 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. *Nature Photonics*, 2(12):728–732, 2008. 8

[82] I. Kanter, Y. Aviad, I. Reidler, E. Cohen, and M. Rosenbluh. An optical ultrafast random bit generator. *Nature Photonics*, 4(1):58–61, 2010. 8

[83] 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. 8

[84] 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. 8

[85] 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. 8

[86] L. Szilard. On the decrease of entropy in a thermodynamic system by the intervention of intelligent beings. *Z. Phys.*, 53:840–856, 1929. 10

[87] 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. 10

[88] W. H. Press. Flicker noises in astronomy and elsewhere. *Comments on Astrophysics*, 7(4):103–119, 1978. 10

[89] 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. 10

[90] 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. 10

[91] C. Easttom. *Modern Cryptography: Applied Mathematics for Encryption and Information Security.* McGraw-Hill Education, New York, 2015. 10

[92] M. Foltyn and M. Zgirski. Gambling with superconducting fluctuations. *Phys. Rev. App.*, 4(2):024002, 2015. 10

[93] V. Zhirnov, R. Cavin, and L. Gammaitoni. *Minimum energy of computing, fundamental considerations*, chapter 7. InTech, 2014. 10

[94] M. Hilbert and P. López. The world's technological capacity to store, communicate, and compute information. *Science*, 332(6025):60–65 10