PAPER • OPEN ACCESS

Thermodynamics of deterministic finite automata operating locally and periodically

To cite this article: Thomas E Ouldridge and David H Wolpert 2023 *New J. Phys.* **25** 123013

IOP Publishing New J. Phys. 25 (2023) 123013 https://doi.org/10.1088/1367-2630/ad1070

New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics

Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

OPEN ACCESS

RECEIVED
8 August 2023

REVISED
12 October 2023

ACCEPTED FOR PUBLICATION
28 November 2023

PUBLISHED
7 December 2023

PAPER

Thermodynamics of deterministic finite automata operating locally and periodically

Thomas E Ouldridge[1,*] **and David H Wolpert**[2,3,4,5]

[1] Imperial College Centre for Synthetic Biology and Department of Bioengineering, Imperial College London, London SW7 2AZ, United Kingdom
[2] Santa Fe Institute, Santa Fe, NM 87501, United States of America
[3] Complexity Science Hub, Vienna, Austria
[4] Arizona State University, Tempe, AZ, United States of America
[5] International Center for Theoretical Physics,Trieste, Italy
[*] Author to whom any correspondence should be addressed.

E-mail: t.ouldridge@imperial.ac.uk

Keywords: Thermodynamics, computation, information, stochastic thermodynamics, mismatch cost, deterministic finite automata, modularity cost

Abstract

Real-world computers have operational constraints that cause nonzero entropy production (EP). In particular, almost all real-world computers are 'periodic', iteratively undergoing the same physical process; and 'local', in that subsystems evolve whilst physically decoupled from the rest of the computer. These constraints are so universal because decomposing a complex computation into small, iterative calculations is what makes computers so powerful. We first derive the nonzero EP caused by the locality and periodicity constraints for deterministic finite automata (DFA), a foundational system of computer science theory. We then relate this minimal EP to the computational characteristics of the DFA. We thus divide the languages recognised by DFA into two classes: those that can be recognised with zero EP, and those that necessarily have non-zero EP. We also demonstrate the thermodynamic advantages of implementing a DFA with a physical process that is agnostic about the inputs that it processes.

1. Introduction

Szilard, Landauer and Bennett emphasized that computations have thermodynamic properties [1–3]. Lately, this insight has been enriched by stochastic thermodynamics [4–6], allowing rigorous analysis of computation far from equilibrium. Recent results include: a trade-off between the minimal amounts of 'hidden' memory and the minimum number of discrete time-steps required to implement a given computation using a continuous-time Markov chain (CTMC) [7]; the excess thermodynamic costs when the distribution of inputs to a computation does not match an optimal distribution [8–11] or involves statistical coupling between physically unconnected computational variables [6, 8, 12, 13]; and results on the thermodynamics of systems implementing loop-free circuits [8], Turing machines [14–16], and Mealy machines [7, 17–19].

These analyses use minimal physical descriptions of the computations performed by the abstract constructs of computer science theory [20, 21]. Some recent work has instead probed the thermodynamics of certain types of hardware, such as CMOS-based electronic circuits [22, 23]. However, there exist practical constraints on physical computation that are not specified by the overall computation performed, but which are nonetheless relevant beyond a particular type of hardware. The thermodynamic costs of these constraints are not resolved by either of the approaches above, although their consequences can be significant [24].

Accordingly, we ask: which kinds of thermodynamic costs necessarily arise when implementing a computation using a physical system *solely due to constraints that seem to be shared by all real-world physical systems that implement digital computation*? To begin to investigate this issue, here we consider the minimal entropy production (EP) that arises due to two ubiquitous constraints on real-world digital computers. First,

the vast majority of modern physical computers are periodic: they implement the same physical process at each iteration (or clock cycle) of the computation. Second, all modern physical systems that perform digital computation are 'local', i.e. not all physical variables that are statistically coupled are also physically coupled when the system's state updates. Ultimately, the reason that these constraints are imposed in both abstract models of computation and real world computers is that they allow us to break down complex computations into simple, iterative logical steps.

In this work we explore how and when operating under these constraints imposes lower bounds on the EP of a computation modelled as a CTMC, regardless of any other details about how the computation is performed (equivalent results apply even in a quantum setting [9]). Taken together, the constraints impose necessary EP through mismatch costs [8–11] of two types: 'modularity' mismatch cost [6, 8, 12, 13], and what we call 'marginal' mismatch cost. Both types of mismatch cost have previously been identified in the literature as possibly causing EP in any given physical process; here we argue that they are in fact inescapable in complex computations. In particular, we demonstrate their effects for one of the simplest nontrivial types of computer, *deterministic finite automata* (DFA).

DFA have important applications in the design of modern compilers, as well as text searching and editing tools [25]. They are also foundational in computer science theory, at the foot of the Chomsky hierarchy [26, 27], below push-down automata [21] and Turing machines [20, 28, 29]. These properties makes DFA particularly well-suited for an initial study of the consequences of locality and periodicity in computational systems. We thus take the first step towards investigating the thermodynamic consequences of locality and periodicity in all the computational machines of computer science theory.

We next introduce our modelling approach and key definitions. We subsequently outline the general consequences of locality and periodicity for arbitrary computations, in the form of a strengthened second law. Having discussed this strengthened second law, we then derive specific expressions for constraint-driven EP in DFA, and explore how DFA could be designed to minimize the expected and worst-case costs that result. Next, we analyse how this EP relates to the underlying computation performed. Surprisingly, the most compact DFA for a given language is generally neither especially thermodynamically efficient nor inefficient. Finally, we consider regular languages, i.e. the sets of strings such that every string in the set can be recognized by some DFA. We show that such languages can be divided into a class that is thermodynamically costly for a DFA to recognise, and a class that is inherently low-cost.

2. Methods

2.1. DFA

A DFA [6, 26, 27] is a 5-tuple $(R, \Lambda, r^{\varnothing}, r^{A}, \rho)$ where: R is a finite set of (computational) *states*; Λ is a finite *alphabet* of input symbols; ρ is a deterministic *update function* specifying how the current DFA state is updated to a new one based on the next input symbol, i.e. $\rho : R \times \Lambda \to R$; $r^{\varnothing} \in R$ is a unique initial state; and $r^{A} \subset R$ is a set of *accepting states*. An example is shown in figure 1. The set of all finite input strings is indicated as Λ^{*}.

The DFA starts in state $r^{\varnothing}$ and an *input string* $\lambda \in \Lambda^{*}$ is selected. The selected input string's first symbol, λ_1, is then used to change the DFA's state to $\rho(\lambda_1, r^{\varnothing})$. The computation proceeds iteratively, with each successive component of the vector λ used as input to ρ alongside the then-current DFA state to produce the next state. We write λ_{-i} for the entire vector λ except for the ith component.

We write the DFA's computational state just before iteration i as r_{i-1}, and we use r_i for the state after the update. The update in iteration i is then the map

$$(\lambda_i, r_{i-1}) \to (\lambda_i, r_i) = (\lambda_i, \rho(\lambda_i, r_{i-1})). \tag{1}$$

We refer to this map as the *local dynamics*, and define the set of *local states* as

$$Z := R \times \Lambda, \tag{2}$$

with elements $z \in Z$. z_i^0 is the local state just before update i: $z_i^0 = (\lambda_i, r_{i-1})$, and $z_i^f = (\lambda_i, r_i)$ is the local state after update i. Note that $z_i^f \neq z_{i+1}^0$ in general, since z_{i+1}^0 involves λ_{i+1}, not λ_i. The local update function fixes the full update function of the entire state space, since λ_{-i} is unchanged during an update.

A DFA *accepts* λ if its state is contained in r^{A} after processing the final symbol. The *language* accepted by a DFA is the set of all input strings it accepts. Many DFA accept the same language L; the *minimal* DFA for L has the smallest set of computational states R for all DFA that accept L [26, 27].

Figure 1(a) shows a DFA with four computational states that processes words built from a two-symbol alphabet $\{a, b\}$. This DFA accepts all strings without three or more consecutive bs. Three iterations of this DFA when fed with an input (a, b, b) are shown in figure 1(b).

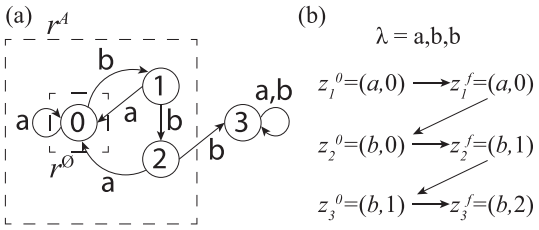

Figure 1. Example DFA with states $R = \{0,1,2,3\}$, alphabet $\Lambda = \{a,b\}$, initial state $r^\varnothing = 0$ and accepting set $r^A = \{0,1,2\}$. The update function ρ is illustrated in (a); the current computational state and the current input symbol specify the next computational state. This DFA accepts input strings that do not contain three or more consecutive *b*s. (b) shows the evolution of the local state through three iterations; the input string is read from left to right.

DFA can be divided into those with an invertible local map ρ, and those with a non-invertible ρ. The map ρ defines islands in the local state space: an *island* of ρ is a set of all inputs to ρ that map to the same output (i.e. it the pre-image of an output of ρ). If the local dynamics defined by ρ is invertible, all local states are islands of size 1; otherwise Z is partitioned by ρ^{-1} into non-intersecting islands, some of which contain multiple elements. We write c_i for the island that contains z_i^0. The DFA in figure 1 is non-invertible, since $z_i^f = (a,0)$ could have arisen from either $z_i^0 = (a,0)$, $(a,1)$ or $(a,2)$, which comprise an island.

2.2. Relevant stochastic thermodynamics

2.2.1. Mismatch cost

Consider an arbitrary system X with a finite set of states $\mathcal{X} = \{x_1, x_2 \ldots\}$. There is a distribution $p(x)$ over $\mathcal{X}$ at some initial time, and that distribution evolves according to a (potentially time-dependent) Markov process $\mu(t)$. We assume that the system is attached to a single heat bath during this process, choosing units so that the bath's temperature equals $1/k_B$. We also assume that $\mu(t)$ obeys local detailed balance with respect to that bath and the system's (potentially time-evolving) Hamiltonian [4]. Although we will not need to specify whether the Markov process is discrete-time or continuous-time, to fix the reader's intuition (and accord with real-world digital computers) we can assume that it is continuous-time.

Suppose that the process runs for some pre-fixed time. The distribution over $\mathcal{X}$ at the end of that time is a linear function of the initial distribution, which we write as $p'(x') = \sum_x P(x'|x)p(x)$, or just $p' = Pp$ for short, where P is implicitly fixed by the stochastic process $\mu(t)$. A given P will partition $\mathcal{X}$ into *islands*. Two states x and x' are within the same island if and only if $P(x''|x) \neq 0$ and $P(x''|x') \neq 0$ for any state x''.

Let $q_\mu^c(x)$ be the initial probability distribution that minimizes the EP under $\mu(t)$ for distributions with support restricted to the island c. This optimal distribution will be unique within each island. No matter what the actual initial distribution p is, and regardless of the specific details of the process $\mu(t)$ that implements P, so long as each q_μ^c has full support within island c, the EP when the process is run with the initial distribution p will be [8, 9, 11]

$$\sigma_\mu(p) = D(p||q_\mu) - D(Pp||Pq_\mu) + \sum_c p(c)\,\sigma_\mu\left(q_\mu^c\right). \tag{3}$$

Here, the index c runs over the islands of the process, $p(c) = \sum_{x\in c} p(x)$ and $q_\mu(x) = \sum_c q(c) q_\mu^c(x)$ is called the *prior* distribution [6, 15], and is specific to the applied protocol $\mu(t)$. $D(p||q_\mu)$ is the Kullback–Leibler (KL) divergence between p and q_μ.

The first two terms in equation (3) are the *mismatch cost* [6, 8, 9] of the process. The mismatch cost is the drop in KL divergence between p and q_μ due to the matrix P. The mismatch cost is zero if $p = q_\mu$, and non-negative by the data processing inequality. Intuitively, the mismatch cost is the contribution to the EP of the misalignment between the actual input distribution $p(x)$ and an optimal distribution $q_\mu(x)$ specified by the physical process $\mu(t)$. If the input distribution is well-matched to the protocol applied, $p(x) = q_\mu(x)$, EP is minimised.

The final term in equation (3) is the *residual entropy production*. Unlike the statistical mismatch cost, the residual EP depends on the physical details of the process implementing $\mu(t)$. Each term in the sum is non-negative, but can be reduced to zero using a quasi-static process [6, 8, 9].

Note that the distribution over islands, $q_\mu(c)$, is arbitrary. Any distribution $q_\mu(x)$ that is a sum over the set of optimal distributions $\{q_\mu^c(x)\}$ could be used with the same results. In practice, the existence of many possible q_μ does not affect our analysis; we shall simply use a convenient q_μ with $q_\mu(c) \neq 0$ for all c.

2.2.2. Marginal and modularity mismatch costs

Let X_a and X_b be two co-evolving systems that are physically separated from one another during a time period $[0,1]$, though they may have been coupled in the past. Due to this separation, we may consider separate protocols $\mu_a(t)$ and $\mu_b(t)$. Moreover, the prior for the overall process must be a product distribution, $q_\mu(x) = q_{\mu_a}(x_a)q_{\mu_b}(x_b)$ Taking $p(x_a)$ and $p(x_b)$ as the marginal distributions of the initial joint distribution $p(x_a,x_b)$, the drop in KL divergence during $[0,1]$ is

$$D\left(p\left(x_a,x_b\right)||q_{\mu_a}\left(x_a\right)q_{\mu_b}\left(x_b\right)\right) - D\left(Pp\left(x_a,x_b\right)||P_a q_{\mu_a}\left(x_a\right)P_b q_{\mu_b}\left(x_b\right)\right), \tag{4}$$

where P_a, P_b are the two matrices specifying the overall evolution of the a and b subsystems during $[0,1]$. This drop equals

$$-\Delta H\left(p\left(x_a,x_b\right)\right) + \Delta H\left(p_a\left(x_a\right)||q_{\mu_a}\left(x_a\right)\right) + \Delta H\left(p_b\left(x_b\right)||q_{\mu_b}\left(x_b\right)\right), \tag{5}$$

where H is the entropy, $H(.\,||\,.)$ is cross-entropy, and Δ means change from beginning to end of the evolution under P. Adding and subtracting marginal entropies, this form can be re-expressed as

$$-\Delta H\left(p\left(x_a,x_b\right)\right) + \Delta H\left(p_a\left(x_a\right)\right) + \Delta H\left(p_b\left(x_b\right)\right) + \Delta D\left(p_a||q_{\mu_a}\right) + \Delta D\left(p_b||q_{\mu_b}\right). \tag{6}$$

By the definition of the change of mutual information between X_a and X_b, ΔI, we obtain

$$D\left(p\left(x_a,x_b\right)||q_{\mu_a}\left(x_a\right)q_{\mu_b}\left(x_b\right)\right) = \Delta D\left(p_a||q_{\mu_a}\right) + \Delta D\left(p_b||q_{\mu_b}\right) - \Delta I. \tag{7}$$

We may thus write

$$\sigma_\mu\left(p\right) = \sigma_{\mu_a}\left(p_a\right) + \sigma_{\mu_b}\left(p_b\right) - \Delta I, \tag{8}$$

for the EP during $[0,1]$, which simplifies to

$$\sigma = \sigma_{\mu_a}\left(p_a\right) - \Delta I. \tag{9}$$

if $X_b = X_{-a}$ and X_{-a} is static; we say that X_a evolves under a *solitary* process in this case. Here, $p_a(x_a)$ is the initial marginal distribution for subsystem a, and ΔI is the change in the mutual information between X_a and X_{-a} over the period in question.

The first term in equation (9) is the non-negative mismatch cost generated by X_a running in isolation, having marginalised over the other degrees of freedom. We call this the *marginal* mismatch cost, σ_{mar}. Like any other mismatch cost, it is non-negative. The second term is the reduction in mutual information between X_a and X_{-a} [7, 13], which we call the *modularity* mismatch cost, σ_{mod}, after [12]. By the data processing inequality [30], $\sigma_{\text{mod}} \geqslant 0$. Intuitively, this term reflects the fact that information about the statistical coupling between X_a and X_{-a} is a store of non-equilibrium free energy, and that information is reduced in a solitary process.

Equation (9) implies that the mismatch cost can be exactly decoupled into modular and marginal components when evolution is solitary. This result may, at first glance, seem inconsistent with the general discussion in [8], which used a more general Bayes net formalism. In fact there is no inconsistency. In the language of [8], the variables in $\mathbf{z}_0^i$ are the 'parents' of $\mathbf{r}_i$, resulting in the same marginal and modularity mismatch costs as derived here.

2.3. Physical model of DFA

2.3.1. State space

In order to apply stochastic thermodynamics of section 2.2 to the computational model of DFA in section 2.1, it is necessary to make assumptions about how the logic is instantiated in a physical system. We assume that all the possible logical states of the system, defined by the set $R \times \Lambda^* \times \mathbb{Z}^+$ (combining the possible computational states, input words and iteration steps) correspond to well-defined discrete physical states [4, 31]. For example, the DFA could be a molecular assembly processing a copolymer tape [14]. Metastable configurations of the assembly would represent the computational state, the sequence of the copolymer the state of the input word, and the position of the polymer the iteration. We also assume that if it is necessary to implement ρ, the DFA has access to ancillary hidden states—which with probability 1 are unoccupied at the start and end of any update [32].

Computation will, in general, involve an externally applied *control protocol* that varies the physical conditions of the system over time; in the case of the molecular computer, we would use time-varying concentrations of molecular fuel [14]. This protocol defines the dynamics $\mu(t)$. Although the dynamics will

be stochastic, strictly speaking, we assume that $\mu(t)$ biases trajectories sufficiently to obtain effectively deterministic computation by the end of each update More formally, we are interested in the limits of stochastic protocols under which they approximate deterministic dynamics to arbitrary accuracy [33]. We abuse notation, using $\mu(t)$ to refer to both the external protocol and the dynamics it induces over the system's states.

We take the input word $\boldsymbol{\lambda}$ to be a random variable sampled from a distribution $p(\lambda)$. As a result the computational state of the DFA will also be a random variable, despite its deterministic operation on a given word. We use $\mathbf{r}_i$, $\mathbf{z}_i^0$, $\mathbf{z}_i^f$ and $\mathbf{c_i}$ to represent the random variables corresponding to the computational state of the DFA after update i, local state before and after update i; and the island occupied during iteration i, respectively.

We will consider a distribution $p(\lambda)$ in which all words are the same finite length N. Within this setup, a distribution of input words with lengths less than or equal to N could be simulated by adding an extra null symbol that induces no computational transitions to the alphabet. Processing these null input symbols would have no thermodynamic cost under the assumptions considered here. For simplicity, we do not include these null symbols in our examples.

2.3.2. Thermodynamic costs of DFA

In this paper we focus on EP as the fundamental thermodynamic cost of running DFA. EP represents the lost ability to extract work from a system, and is a metric for the thermodynamic irreversibility. In certain contexts, the work required to perform a process, or the heat transferred to the environment therein, are also used to quantify the thermodynamic cost of a process.

The operation of a DFA does not increase the entropy of the computational degrees of freedom of the system, since the map from $(r_0 = r^{\varnothing}, \lambda)$ to (r_N, λ) is one-to-one if the full input word is taken into account. If the computational states all have the same energy and intrinsic entropy [4, 31] as is typically assumed, the energy and entropy change of the system will thus be zero. Any EP is equal to the heat transferred to the environment, which must be exactly compensated by the work done on the system. All three measures of thermodynamic cost are therefore identical.

We do not consider further the residual EP at each iteration, nor the costs of incrementing i (both can, in principle, be made arbitrarily small). We also neglect costs associated with actually generating $\mu(t)$ itself, as discussed in [14]. Given these assumptions, whenever we use the term '(minimal) EP', we refer to the (minimal) EP due to the mismatch cost (and its decomposition into marginal and modularity mismatch cost).

2.3.3. Decomposition of EP generated at each iteration

In general, when applying the mismatch cost formula to a computation there are multiple choices for the times of the beginning and end of the underlying process. This choice matters, because the mismatch cost contribution to EP is not additive over time. for example, the drop in KL divergence for a two-timestep computation will generally differ from the sum of the drops in KL divergence for each of those timesteps.

One could consider a single mismatch cost evaluated over the entire computation. Under this choice, none of the details of *how* the conditional distribution P of the overall computation arises by iterating the conditional distributions of each step are resolved by the mismatch cost. All that matters is the drop in KL divergence between the initial distribution, when the computer is initialized, and the ending distribution, when the output of the computation is determined. This approach has been used to analyse Turing Machines [7, 15] as well as DFA [34].

An alternative choice is to focus on the EP generated at each iteration of the DFA, with the total EP of the entire computation being a sum of those iteration-specific EPs. Doing so allows us to manifest restrictions on the applied protocol inherent to the iterative process in the mismatch cost, rather than burying them in the residual EP of the computation as a whole. Given that we focus on costs arising from the iterative nature of the computation, it is natural to focus on the EP at each iteration of the DFA.

2.4. Implementation of physical constraints

2.4.1. Locality

In principle, one could build a DFA that physically couples the entire input word, λ, to the local subsystem z_i during update i. However, this coupling is not required by the computational logic, which is local to z_i. Moreover, it would be extremely challenging to implement in practice; modern computers do not physically couple bits that do not need to be coupled by the logical operation in question. Accordingly, we assume that the evolution of the local state z_i is solitary. As a result, the global mismatch cost splits into two non-negative components, as outlined in section 2.2: a marginal mismatch cost associated with the evolution of the local

state in isolation; and a modularity mismatch cost associated with non-conserved information between the local state and the rest of the system.

2.4.2. Periodicity

The marginal mismatch cost for iteration i will depend on the similarity of $p(z_i^0)$, the initial distribution over local states, and $q_{\mu_i}(z_0^i)$, the prior distribution for the protocol $\mu_i(t)$ implemented at iteration i. Typically, $p(z_i^0)$ will vary with i. In theory, one could design $\mu_i(t)$ to match these variations, ensuring $q_{\mu_i}(z_0^i) = p(z_i^0)$ at each update, eliminating $\sigma_{\rm mar}$. However, designing such a protocol would require knowledge of $p(z_i^0)$ — which in turn would require running a computation emulating the DFA before running the DFA, gaining nothing. Moreover, one of the major strengths of computing paradigms such as DFA, Turing machines and real world digital computers is that their logical updates are not iteration-dependent. It is therefore natural to impose a second constraint: that the protocol $\mu_i(t)$, like the logical update ρ, is identical at each update i ($\mu_i(t) = \mu(t)$).

The applied protocol is thus periodic, because it repeats at each iteration, and we use the term periodic to describe a DFA operated in this way. We note that a periodic protocol does not necessarily imply periodic dynamics, which also depends on the input word. Formally, we define a *local, periodic* DFA (LPDFA) as any process that implements a DFA via a repetitive, solitary process on the local state z_i.

2.5. Calculation of numerical results

The numerical results reported in this manuscript were obtained via an explicit summation over all possible input words and the states that result at each iteration. Code for calculating the data can be found at [35].

3. Results and discussion

3.1. General consequences of local and periodic constraints

We briefly consider the consequences of locality and periodicity in general, before re-focussing on DFA specifically. The mismatch and modularity costs previously introduced are well established. However, systems that perform non-trivial computations by iterating logical steps on subsystems are exposed to these costs in a way that simpler operations, like erasing a bit, are not. The need to operate iteratively on an input that is evolving from iteration to iteration makes the mismatch cost unavoidable. Additionally, modularity-cost-inducing statistical correlations result from the need to carry information between iterations, which will not be required in simpler systems, such as erasing a series of bits.

Consider a physical realisation of an arbitrary computation that is local and periodic in a way that reflects the locality and periodicity of the computational logic. Then the marginal and modularity mismatch costs set a lower bound on EP, regardless of any further details about how the computation is implemented. Specifically, let X be the computational system and X_i the local subsystem that is updated at iteration i. Then over the course of N iterations, the system will experience a total marginal mismatch cost

$$\sigma_{\rm mar} = \sum_{i=1}^{N} D\left(p\left(x_i\right) || q_\mu\left(x_i\right)\right) - D\left(Pp\left(x_i\right) || Pq_\mu\left(x_i\right)\right), \tag{10}$$

where P is the update matrix, $p(x_i)$ is the initial distribution of the local state and $q_\mu(x_i)$ is the prior built in to the actual protocol $\mu(t)$.

Equation (10) depends on the details of $\mu(t)$ beyond the locality and periodicity constraints. However, some choice of q_μ (and hence $\mu(t)$) will minimize $\sigma_{\rm mar}$, setting a lower bound on EP that is independent of these details

$$\sigma_{\rm mar} \geqslant \min_{q_\mu} \sum_{i=1}^{N} D\left(p\left(x_i\right) || q_\mu\left(x_i\right)\right) - D\left(Pp\left(x_i\right) || Pq_\mu\left(x_i\right)\right) \geqslant 0. \tag{11}$$

Unless $p(x_i)$ is identical for all i, or P is a simple permutation, it is not generally possible to choose a single q_μ that will eliminate $\sigma_{\rm mar}$ at every iteration i. In this case, equation (11) provides a strictly positive periodicity-induced lower bound on the EP that depends purely on the logic of local update.

Similarly, the accumulated modularity cost follows directly as

$$\sigma_{\rm mod} = -\sum_{i=1}^{N} \Delta I(X_i; X_{-i}) \geqslant 0, \tag{12}$$

where $\Delta I(X_i; X_{-i})$ is the change in mutual information between X_i and X_{-i} due to update i. As with equation (11), this contribution to EP is entirely determined by the computational paradigm used and the

distribution of inputs; it is independent of the details of the implementation given the assumption of locality and periodicity. Taken together, the sum of σ_{mar} and σ_{mod} from equations (11) and (12) constitute a strengthened second law for periodic, local computations that depends only on the logic of the computation, not the details of its implementation.

These implementation-independent lower bounds, alongside the qualitative observation that computing systems are particularly vulnerable to modularity and mismatch costs, is the first main result of this work. These results apply to any computational system implemented using a periodic, local process. For the rest of the paper, we will focus on DFA. Doing so allows us to illustrate the consequences of local and periodic restrictions in a concrete computational model.

3.2. EP for LPDFA

Under our assumptions, the EP when applying a solitary dynamics $\mu(t)$ to an initial distribution $p(z_i^0, \lambda_{-i})$ at the update stage of iteration i of a DFA is

$$\sigma_\mu^i \left(p\left(z_i^0, \lambda_{-i}\right)\right) = \sigma_{\text{mar}}^i + \sigma_{\text{mod}}^i, \tag{13}$$

where

$$\sigma_{\text{mar}}^i := D\left(p\left(z_i^0\right) || q_\mu\left(z_i^0\right)\right) - D\left(p\left(z_i^f\right) || q_\mu\left(z_i^f\right)\right) \tag{14}$$

is the marginal mismatch cost of update i, and

$$\sigma_{\text{mod}}^i := I\left(\mathbf{z}_i^0; \boldsymbol{\lambda}_{-i}\right) - I\left(\mathbf{z}_i^f; \boldsymbol{\lambda}_{-i}\right) \tag{15}$$

is the modularity mismatch cost of update i. A variant of the modularity cost in equation (15) was considered in isolation in [36], for the special case of DFA operating in steady state.

Henceforth, for simplicity, we suppress the dependence of σ^i on μ since μ is constant over all iterations. The KL divergences in equation (14), giving σ_{mar}, can be simplified for LPDFA. Since each update in an LPDFA deterministically collapses all probability within an island to one state, $p(z_i^f|c_i) = q(z_i^f|c_i)$. As shown in section 1 of the supplementary information, this simplification implies that

$$\sigma_{\text{mar}}^i = \sum_{c_i} p\left(c_i\right) D\left(p\left(z_i^0|c_i\right) || q_\mu\left(z_i^0|c_i\right)\right), \tag{16}$$

which is the second main result of this work. σ_{mar}^i is therefore the divergence between initial and prior distributions, conditioned on the island of the initial state.

In figure 2, we explore the properties of σ_{mar}^i for the DFA shown in figure 1. The four sub-figures show σ_{mar}^i for four distinct distributions $p(\lambda)$, and a fixed (uniform) prior q_μ. We immediately see that σ_{mar}^i is strongly dependent on both the distribution of input words and the iteration, with σ_{mar}^i non-monotonic in i in all four cases.

σ_{mar}^i is determined by a combination of how well tuned the prior is to the input distribution within a given island, and the probability of that island at each iteration. At the start of iteration 1, particularly for subfigure (b), there is a high probability of the system being in the island $\{(a,0);(a,1);(a,2)\}$ and the uniform prior is poorly aligned with the actual initial condition within this island (all in $(a,0)$). At larger i, this cost drops both because the probability of being in that island drops, and the conditional distribution within the island gets more uniform.

For iterations $i \geqslant 3$, the system has a non-zero probability of being in the other non-trivial island $\{(b,2);(b,3)\}$. The uniform prior is initially poorly matched to the conditional distribution within this island (at the start of iteration $i = 3$, the system cannot be in $(b,3)$). Additionally, the probability of the system being in this island is quite low for subfigure (a) and (b), but much higher for (c) and (d) – explaining the jumps in those traces.

The third main result of this work is a simple expression for the modularity mismatch cost for DFA. As we show in section 2 of the supplementary information,

$$\sigma_{\text{mod}}^i = H\left(\mathbf{z}_i^0|\mathbf{c}_i\right). \tag{17}$$

Surprisingly, σ_{mod}^i, a *global* quantity, is given by the entropy of the *local* state at the beginning of the update, conditioned on the island occupied at the start of iteration i. This result holds regardless of the distribution of input strings or the DFA's complexity.

To understand equation (17) intuitively, we note that $\mathbf{z}_i^0$ in general contains information about $\boldsymbol{\lambda}_{-i}$. After the update, any information provided by $\boldsymbol{\lambda}_i$ alone is retained, since the input symbol is not updated by

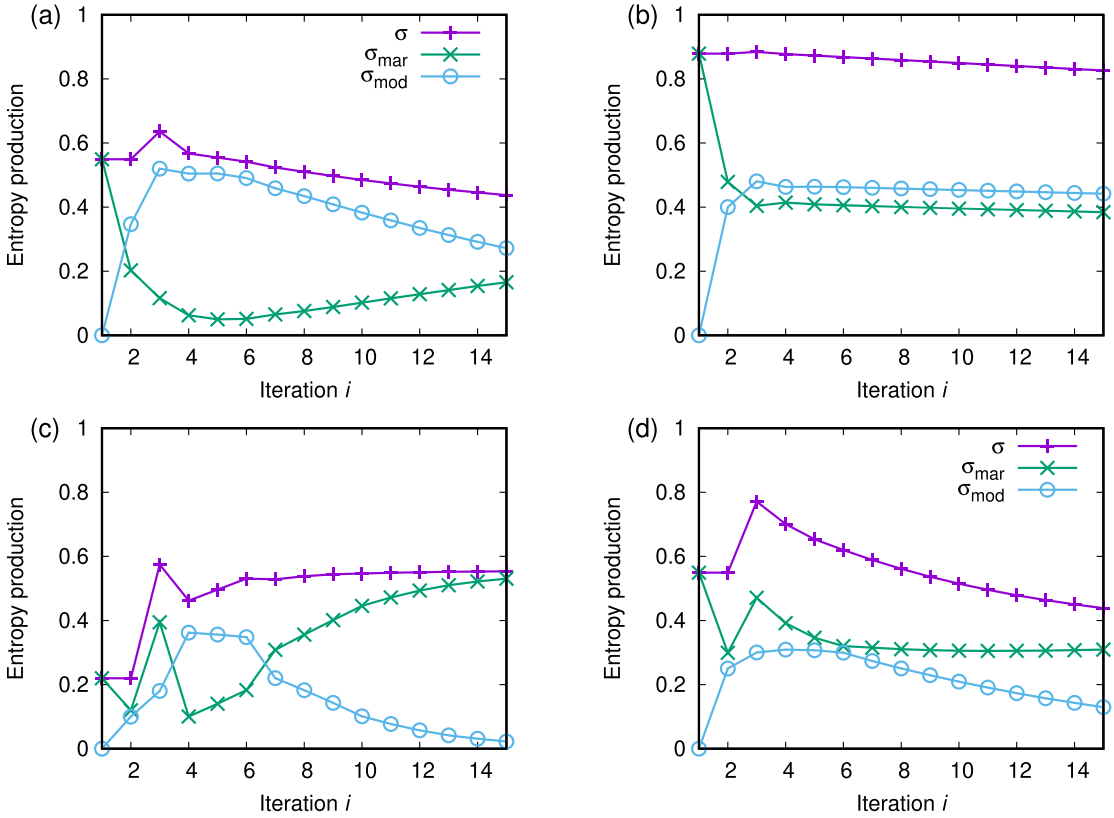

Figure 2. EP in a simple system shows non-trivial dependence on iteration and input word distribution. We plot total EP σ^i, and its decomposition into σ^i_{mar} and σ^i_{mod}, for the DFA in figure 1(a), which accepts all words that do not contain three or more consecutive *b*s. In all cases we use a uniform prior $q_\mu(z_i^0|c_i)$ within each island, and consider a distribution of input words with fixed length $N = 15$, but vary the distribution of input words $p(\lambda)$. (a) Input words have independent and identically distributed (IID) symbols with $p(a) = p(b) = 0.5$. (b) Input words have IID symbols with $p(a) = 0.8$ and $p(b) = 0.2$. (c) Input words have IID symbols with $p(a) = 0.2$ and $p(b) = 0.8$. (d) Input words are Markov chains. The first symbol is *a* or *b* with equal probability, and subsequently $P(\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i) = 0.8$.

the DFA. Moreover, for islands of size 1, the combined values of $\boldsymbol{\lambda}_i$ and $\mathbf{r}_i$ are just as informative about $\boldsymbol{\lambda}_{-i}$ as $\boldsymbol{\lambda}_i$ and $\mathbf{r}_{i-1}$ were. However, for non-trivial islands, the extra information provided by $\mathbf{r}_{i-1}$ on top of $\boldsymbol{\lambda}_i$ is lost, yielding equation (17).

We see from our example system in figure 2 that modularity costs behave very differently from marginal mismatch costs. In general, σ^i_{mod} tends to zero as the probability of being absorbed into state 3 increases: in this case, there is no entropy of $\mathbf{z}_i^0$. Modularity costs stay high for system (b), in which *bbb* substrings are infrequent.

Modularity costs are relatively low in figure 2(d), in which symbols of the input word are correlated. Naïvely, one might have assumed that a larger $I(\mathbf{z}_i^0, \boldsymbol{\lambda}_{-i})$ generated by a correlated input word would be more susceptible to large modularity costs. We explore this question in more detail in figure 3 for both the DFA illustrated in figure 1(a) and a second DFA that accepts words that are concatenations of *bb* and *baa* substrings (figure 3(a)).

In figure 3(b) we plot the total modularity cost, $\sum_{i=1}^N \sigma^i_{\text{mod}}$, for both DFA processing a Markovian input, as a function of the degree of correlation, $P(\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i)$. We see that in both cases, the uncorrelated input words with $P(\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i) = 0.5$ have relatively high (though not maximal) modularity cost, and fully correlated strings have $\sigma_{\text{mod}} = 0$.

To understand why, consider figure 3(c), in which we plot the information between the local state and the rest of the input word before ($I_0 = I(\mathbf{z}_i^0; \boldsymbol{\lambda}_{-i})$) and after ($I_f = I(\mathbf{z}_i^f; \boldsymbol{\lambda}_{-i})$) the update of iteration *i*, for the original DFA in figure 1(a). We consider uncorrelated input words ($P(\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i) = 0.5$) and moderately correlated input words ($P(\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i) = 0.8$). At early iterations, I_0 is larger for the correlated input, as would be expected (at later times, the DFA with correlated input is more likely to be absorbed into state 3, reducing I_0). More importantly, the system with correlated inputs retains more of its information in the final state. Because $\boldsymbol{\lambda}_{-i}$ is correlated with the current symbol $\boldsymbol{\lambda}_i$, it is a better predictor of the final state of the update. In the limit of $P(\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i) = 1$ or 0, there is no modularity cost as $\mathbf{z}_i^f$ is perfectly predictable from $\boldsymbol{\lambda}_{-i}$.

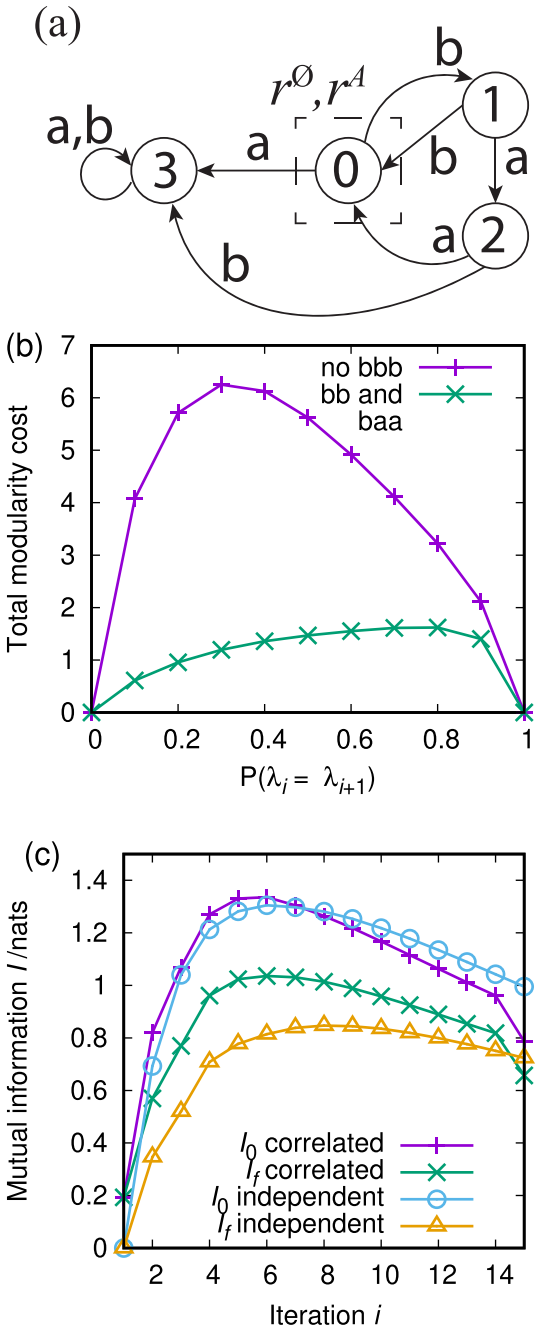

Figure 3. Correlated input words do not generate high modularity costs. (a) A 4-state DFA that processes words formed from a two-symbol alphabet, accepting those formed by concatenating *bb* and *baa* substrings. (b) Total modularity cost $\sum_{i=1}^{N} \sigma^{i}_{\text{mod}}$ for the DFA in (a) and the DFA in figure 1(a), when processing words of length $N = 15$ that are generated using a Markov chain. Modularity cost is plotted as a function of the probability that subsequent symbols in the word have the same value. (c) Mutual information between the local state and the rest of the input word before (I_0) and after (I_f) the update of iteration i, for the DFA in figure 1(a). Data is plotted for $P(\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i) = 0.8$ (correlated) and $P(\boldsymbol{\lambda}_{i+1} = \boldsymbol{\lambda}_i) = 0.5$ (independent).

Combining equations (16) and (17) gives

$$\sigma^{i}\left(p\left(z_i^0, \lambda_{-i}\right)\right) = \sum_{c_i} p\left(c_i\right) H\left(p\left(z_i^0|c_i\right) \,||\, q_\mu\left(z_i^0|c_i\right)\right), \tag{18}$$

where

$$H\left(p\left(z_i^0|c_i\right) \,||\, q_\mu\left(z_i^0|c_i\right)\right) := -\sum_{z_i^0 \in c_i} p\left(z_i^0|c_i\right) \ln q_\mu\left(z_i^0|c_i\right) \tag{19}$$

is the *cross entropy* between $q_\mu(z_i^0|c_i)$ and $p(z_i^0|c_i)$. This total EP is also shown for the example DFA of figure 1(a) in figure 2.

It is interesting to note that, in figure 2, changes in σ^i_{mod} and σ^i_{mar} from iteration to iteration appear to be anticorrelated, with the total σ^i showing only relatively small changes. For the specific case of the simple uniform prior used here, the marginal mismatch cost in equation (16) reduces to $\sigma^i_{\text{mar}} = \langle \ln L_{c_i} \rangle - H(\mathbf{z}^0_i|\mathbf{c}_i) = \langle \ln L_{c_i} \rangle - \sigma^i_{\text{mod}}$, with L_c being the size of island c. With a uniform prior, changes in σ^i_{mar} resulting from changes to $p(z^0_i|c_i)$ are exactly compensated by changes in σ^i_{mar}, and the change in overall EP is only due to changes in $p(c_i)$.

3.3. Reducing the marginal mismatch cost through choice of priors

3.3.1. Applying a bias to the prior

Although changes in σ^i_{mar} and σ^i_{mod} cancel each other when $p_(z^0_i|c_i)$ is varied at fixed $p(c_i)$ and uniform $q_\mu(z^0_i|c_i)$, it is still natural to ask how $q_\mu(z^0_i|c_i)$ might be chosen to minimize EP for a given $p(\lambda)$ and a given DFA. We note that σ^i_{mod} is determined exclusively by the computational map performed by the DFA and the distribution of input words (equation (17)); it is independent of $q_\mu(z^0_i|c_i)$. Optimizing the prior therefore corresponds to minimising σ^i_{mar} for a fixed design of DFA and distribution of input words.

One might hope that $q_\mu(z^0_i|c_i)$ could be tuned to $p(\lambda)$ alone, without any reference to the operation of the DFA. Unfortunately, however, such an approach will fail. The states within each island all have the same value of λ, because the update map ρ does not update the input symbol. Applying a prior that is a function of λ alone results in a uniform $q_\mu(z^0_i|c_i)$.

Reducing the mismatch cost through choice of prior thus requires some understanding of the computational state, not just the inputs. For example, for the DFA in figure 1(a), the computation starts in the state $r^\varnothing = 0$. Biasing $q_\mu(z^0_i|c_i)$ towards states with $r = 0$, as we show in figure 4(a), can reduce the marginal mismatch cost of the first step. If the bias is too strong, then increased costs at later iterations overwhelm the initial reduction. It is possible, however, to reduce the total EP with a moderate bias of $q_\mu(z^0_i|c_i)$ towards states with $r = 0$.

Alternatively, one could bias $q_\mu(z^0_i|c_i)$ towards states with $r = 3$, since most trajectories will eventually be absorbed. As shown in figure 4(b), doing so incurs an extra cost at short times, particularly at iteration $i = 3$. At the start of the third iteration, the DFA is moderately likely to be in computational state $r = 2$, but cannot be in computational state $r = 3$, so the biased prior is a poor match for $p(z^0_i|c_i)$. At later iterations, however, the biased prior performs better. Again, a moderate bias performs best overall.

3.3.2. Advantages of a uniform prior

We have seen that it is, in principle, possible to reduce EP by applying biased priors. However, we also saw that very biased priors could lead to very high EP. As noted in [33], in which a similar result to equation (16) was derived in the absence of distinct islands, σ^i_{mar} penalizes an over-confident prior $q_\mu(z^0_i|c_i)$. If $q_\mu(z^0_i|c_i) = 0$ for a given state but $p(z^0_i|c_i) \neq 0$, equation (18) implies $\sigma^i_{\text{mar}} \to \infty$. The authors of [33] hypothesised, therefore, that a uniform $q_\mu(z^0_i|c_i)$ may be optimal.

As a fourth main result of this work, we present three important properties of a $q_\mu(z^0_i|c_i)$ that is uniform for each c_i, i.e. a prior $q_\mu(z^0_i|c_i) = 1/L_{c_i}$. First, for such a prior, equation (18) becomes

$$\sigma^i\left(p\left(z^0_i, \lambda_{-i}\right)\right) = \langle \ln L_{c_i} \rangle \leqslant \ln L_{c_{\max}}. \tag{20}$$

Here, $L_{c_{\max}}$ is the size of the largest island of ρ. Equation (20) gives a finite upper bound to EP for LPDFA employing a uniform prior distribution $q_\mu(z^0_i|c_i) = 1/L_{c_i}$, constrained by the size of the largest island.

Second, for any protocol, the worst case EP is at least $\ln L_{c_{\max}}$. A uniform prior distribution $q_\mu(z^0_i|c_i) = 1/L_{c_i}$ therefore minimizes the worst case EP. To verify this claim, consider the input distribution $p(z^i_0) = \delta_{z^i_0, z_{\min}}$, where $z_{\min}$ is a state that minimizes $q_\mu(z^0_i|c_i)$ within the largest island. For such a distribution, equation (20) reduces to

$$\sigma^i\left(p\left(z^0_i, \lambda_{-i}\right)\right) = -\ln q_\mu\left(z_{\min}|c_{\max}\right) \geqslant \ln L_{c_{\max}}, \tag{21}$$

where the final inequality follows from $q_\mu(z_{\min}|c_{\max}) \leqslant 1/L_{c_{\max}}$.

Finally, the uniform prior distribution $q_\mu(z^0_i|c_i) = 1/L_{c_i}$ minimizes predicted average EP if a designer is maximally uncertain about $p(z^0_i, \lambda_i)$. A designer may not know that $p_i(z^0_i|c_i)$ is the input distribution at iteration i—either because $p(\lambda)$, or the DFA's dynamics on $p(\lambda)$, are unknown. Thus the choice of protocol $\mu(t)$, and hence $q_\mu(z^0_i|c_i)$, is performed under uncertainty over not just the input state, but also the distribution from which that state is drawn.

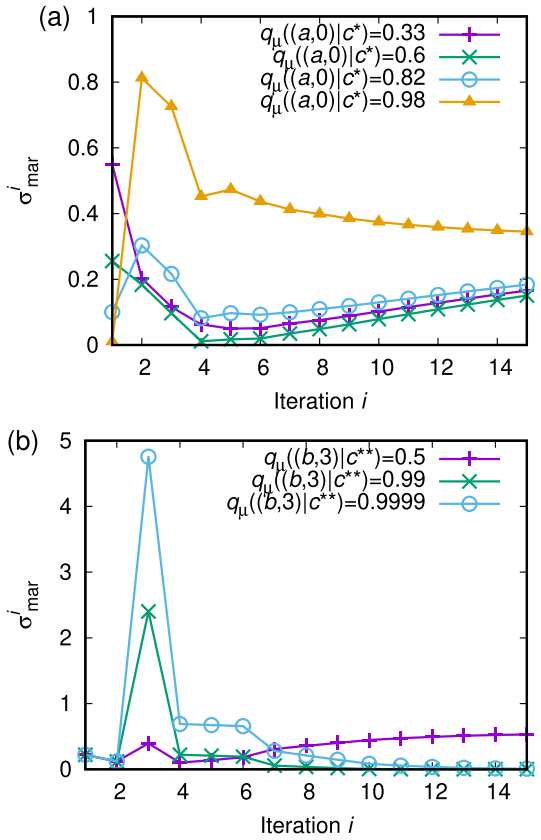

Figure 4. Applying a biased prior $q_\mu(z_i^0|c_i)$ can reduce the local mismatch cost. (a) $\sigma^i_{\rm mar}$ for the DFA in figure 1(a), given input words of length $N = 15$ with IID symbols ($p(a) = p(b) = 0.5$). Results are plotted for different values of $q_\mu((a,0)|c^\star)$, where $c^\star = \{(a,0),(a,1),(a,2)\}$ is the island containing $(a,0)$. $q_\mu(z_i^0|c_i)$ is otherwise unbiased, and $q_\mu((a,0)|c^\star) = 1/3$ corresponds to a totally unbiased prior. (c) Equivalent to (a), but for input $p(a) = 0.2$, $p(b) = 0.8$, and applying a bias to $q_\mu((b,3)|c^{\star\star})$, where $c^{\star\star} = \{(b,2),(b,3)\}$ is the other non-trivial island for this DFA. $q_\mu(z_i^0|c_i)$ is otherwise unbiased, and $q_\mu((b,3)|c^{\star\star}) = 1/2$ corresponds to a totally unbiased prior.

Let the designer's belief about the distributions $p(c_i)$ and $p(z_i^0|c_i)$ be represented by a distribution $\pi(v,w)$ over an (arbitrary) discrete set of possible distributions indexed by v, w: $p_v(c_i)$, $p_w(z_i^0|c_i)$. The designer's best estimate of the expected EP at iteration i is then (see section 3 of the supplementary information)

$$\hat{\sigma}^i = H\left(\mathbf{z}_i^0|\mathbf{c}_i,\mathbf{v},\mathbf{w}\right) + I\left(\mathbf{z}_i^0;\mathbf{w}|\mathbf{c}_i,\mathbf{v}\right) + \sum_v \pi\left(v\right) \sum_{c_i} p_v\left(c_i\right) D\left(\hat{p}\left(z_i^0|c_i,v\right) \,||\, q_\mu\left(z_i^0|c_i\right)\right). \tag{22}$$

Here, $H(\mathbf{z}_i^0|\mathbf{c}_i,\mathbf{v},\mathbf{w})$ and $I(\mathbf{z}_i^0;\mathbf{w}|\mathbf{c}_i,\mathbf{v})$ are defined with respect to the estimated joint distribution, $\hat{p}(v,w,z_i^0,c_i) = \pi(v)\pi(w|v)p_v(c_i)p_w(z_i^0|c_i)$, and $\hat{p}(z_i^0|c_i,v) = \sum_w \pi(w|v)p_w(z_i^0|c_i)$ is the designer's estimate for the probability distribution within an island, having averaged over the uncertainty quantified by $\pi(w|v)$.

All three terms in equation (22) are non-negative. The first is $\sigma^i_{\rm mod}$ averaged over $\mathbf{v}$ and $\mathbf{w}$. The third is the marginal mismatch cost between $\hat{p}(z_i^0|c_i,v)$ and $q_\mu(z_i^0|c_i)$. However, even if $q_\mu(z_i^0|c_i)$ matches the average estimated distribution within an island, $\hat{p}(z_i^0|c_i,v) = q_\mu(z_i^0|c_i)$, the best estimate of $\hat{\sigma}^i_{\rm mar}$ is non-zero. The second term, $I(\mathbf{z}_i^0;\mathbf{w}|\mathbf{c}_i,\mathbf{v})$, quantifies how much uncertainty in $\mathbf{w}$ is actually manifest in an uncertainty in the input distribution; variability about $\hat{p}(z_i^0|c_i,v)$ gives positive expected EP. An equivalent term was previously identified in [37, 38] for arbitrary processes with a single island.

$H(\mathbf{z}_i^0|\mathbf{c}_i,\mathbf{v},\mathbf{w})$ and $I(\mathbf{z}_i^0;\mathbf{w}|\mathbf{c}_i,\mathbf{v})$ are protocol-independent and cannot be changed for a given computation. $D(\hat{p}(z_i^0|c_i,v)\,||\,q_\mu(z_i^0|c_i))$, however, can be minimized by choosing $q_\mu(z_i^0|c_i) = \hat{p}(z_i^0|c_i,v)$. Given maximal uncertainty, the designer's best estimate will be uniform: $\hat{p}(z_i^0|c_i,v) = 1/L_{c_i}$. In this case, a uniform $q_\mu(z_i^0|c_i) = 1/L_{c_i}$ minimizes estimated average EP.

The results hitherto apply to LPDFA, but do not reflect the actual computation performed. The results for $\sigma^i_{\rm mar}$—including the optimality of a uniform protocol—apply to any deterministic process; the LPDFA's restrictions simply justify why $q_{\mu_i}(z_i^0|c_i)$ cannot be tuned to $p(z_i^0|c_i)$ at each i. The results for $\sigma^i_{\rm mod}$ are more specific, relying on a solitary process using a single symbol λ_i from an unchanging 'input string', and a device whose state after the update is unambiguously specified by $\lambda_{i\leqslant j}$. Nonetheless, $\sigma^i_{\rm mod}$ in equation (17) is not

directly related to the computational task. We now explore how EP is related to ρ, and the language accepted by the DFA.

3.4. Relating EP to computational tasks

3.4.1. Nonzero EP is common in non-invertible LPDFA

The EP in equation (18) is positive if $q_\mu(z_i^0|c_i) \neq 1$ for any z_i^0, c_i for which $p(z_i^0|c_i) \neq 0$ and $p(c_i) \neq 0$. This condition is met whenever an island c_i with $p(c_i) > 0$ has at least two elements z_0 with $p(z_i^0|c_i) > 0$. There are two ways to avoid this EP. One is if all islands have a single element, i.e. the local update function ρ is invertible (this observation was made for σ_{mod} alone in [36]). The second is if the distribution of input strings $p(\lambda)$ is such that for every island c_i with at least two elements, all but one of those elements always have $p(z_i^0|c_i) = 0$. However, in that case, $q_\mu(z_i^0|c_i)$ must be finely-tuned to match this condition when the physical system implementing the computation is constructed. As discussed in the context of applying non-uniform priors, this strategy risks high costs for overconfidence.

We now focus on the former way of achieving zero EP, asking what determines whether ρ is invertible. Since ρ preserves the input symbol λ_i, it can only be non-invertible if it maps two distinct computational states to the same output for the same symbol λ_i. If we illustrate ρ by a series of directed graphs, one for each value of λ_i, then a non-invertible DFA will have at least one state with at least two incoming transitions for at least one value of λ_i. We label states with more than one incoming transition for a given λ_i as *conflict states*; conflict states for the DFA in figure 1(a) are shown in figure 5. We note that conflict states cause both $\sigma^i_{\text{mar}} > 0$ and $\sigma^i_{\text{mod}} > 0$; we see no obvious trade-off in DFA structure between the two sources of EP.

3.4.2. The minimal DFA for a given language does not generally minimize or maximize EP

The minimal DFA for a language L has the smallest set of computational states R for all DFA that accept L. This minimal DFA has just enough memory to sort parsed substrings into classes of equivalent strings, so that information can be passed forward to complete the computation [26, 27, 39]. More formally, define input strings λ and μ to be *equivalent* with respect to language L if $\lambda\nu \in L \iff \mu\nu \in L$ for any string of input symbols ν, where $\lambda\nu$ is a concatenation of ν after λ. The Myhill–Nerode theorem states that the number of states of the minimal DFA for L is the number of equivalence classes of this equivalence relation [26, 27, 39].

Perhaps surprisingly, minimal LPDFA do not in general either maximise or minimise EP. This claim is our fifth main result; to illustrate it, first consider the two DFA in figure 6, which both have $\Lambda = \{a, b\}$ and accept input strings with an even number of bs. Figure 6(a) is the minimal DFA for this language. It is invertible, and so has zero EP. The larger DFA in figure 6(b) is non-invertible, and so $\sigma^i(p(z_i^0, \lambda_{-i})) > 0$ in general. For example, EP is positive if the sequences $(\Lambda_{i-2}, \Lambda_{i-1}, \Lambda_i) = (a \text{ or } b, b, a)$ and $(\Lambda_{i-2}, \Lambda_{i-1}, \Lambda_i) = (b, a, a)$ both have non-zero probability. The minimal LPDFA never has higher EP than larger DFA, and often has lower EP.

Now consider the two DFA in figure 7. Both accept any input string constructed from $\Lambda = \{a, b\}$ with no b symbols, and figure 7(a) is the minimal DFA for this language. Neither DFA is invertible, so EP is generally non-zero for both. However, the non-minimal LPDFA in figure 7(b) delays EP by a single iteration relative to figure 7(a). As outlined in section 4 of the supplementary information, this delay ensures that the overall EP for the larger LPDFA is always less than or equal to the EP for the minimal LPDFA.

3.5. Languages are divided into costly and low-cost classes by the structure of their minimal DFA

The DFA in figure 7(b) can be extended, delaying non-zero EP. However, a finite number of additional states cannot prevent EP for arbitrary length inputs, and DFA are *necessarily* finite. Indeed, the sixth main result of our work, proven in detail in section 5 of the supplementary information, is that if a minimal DFA is non-invertible, any DFA that accepts the same language must also be non-invertible. One cannot eliminate conflict states without disrupting the sorting of strings into equivalence classes. Thus if the minimal DFA for a regular language L is non-invertible, recognising that language is inherently costly. Conversely, if the minimal DFA that accepts L is invertible, recognising that language is low-cost.

As an example, consider a DFA that takes inputs of integers in base n, and accepts the integer y if y is divisible by m. As we show in section 6 of the supplementary information, the minimal DFA for such a computation is invertible if n and m have no common factors. Therefore, it is inherently costly to decide whether a number is divisible by 9 if the number is expressed in base 3, but not if the number is expressed in base 2, showing that even conceptually similar computations can have very different thermodynamic consequences.

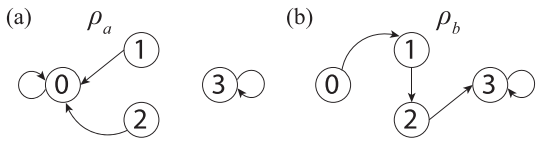

Figure 5. Decomposition of the DFA in figure 1(a) into networks of transitions for each input symbol, ρ_λ. (a) Network for $\lambda_i = a$, where the state $r = 0$ is a conflict state. (b) Network for $\lambda_i = b$, where the state $r = 3$ is a conflict state.

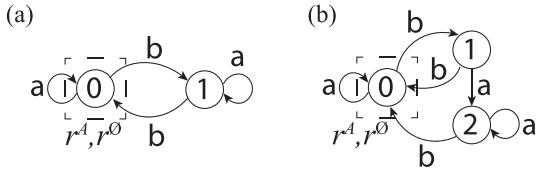

Figure 6. Two DFA that accept input strings with an even number of bs built from $\Lambda = \{a, b\}$. (a) The minimal DFA for this language; it is invertible. (b) A larger DFA that accepts the same language but is non-invertible; state 0 is a conflict state for ρ_b and state 2 is a conflict state for ρ_a.

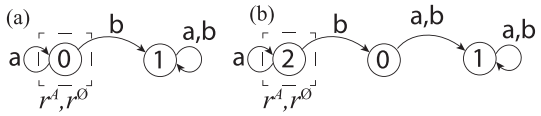

Figure 7. Two DFA that accept input strings built from an alphabet $\Lambda = \{a, b\}$ that contain no bs. (a) The minimal DFA that accepts this language. (b) An alternative, larger DFA.

4. Conclusion

Breaking down complex computations into simple periodic updates, involving small parts of the computational system, is at the heart of both theoretical computer science and real-world computing devices. It is natural that physical systems designed to implement computations involve physical processes that are also local and periodic; indeed, that is how synchronous, clocked digital computers are designed.

However, physical systems that implement periodic, local computations are particularly vulnerable to stronger lower bounds on EP than the zero bound of the second law. Any physical operation—including computations—can, in principle, be performed in a thermodynamically reversible way, with a sufficiently well-designed protocol [32]. The nature of non-trivial computations, however, means that such a protocol would need to reflect not just the distribution of possible inputs to the computer, but also how those inputs are processed, and the subtle statistical coupling that is generated as the computation proceeds.

We have illustrated how these challenges manifest as marginal and modularity mismatch costs in DFA with non-invertible local update maps. Interestingly, the overall computation performed by a DFA—mapping the input word and starting computational state to the same input word and a final computational state—is always invertible. The logical properties of the overall computation are therefore not helpful in understanding the necessary EP of a local, periodic device.

We have an incomplete understanding of why the curves in figures (2)–(4) have the forms they do. Additionally, although similar results will hold for quantum mechanical or finite heat-bath treatments of DFA's thermodynamics, additional subtleties will arise. More generally, DFA are just the simplest machine in the Chomsky hierarchy and it is unknown how marginal and modularity mismatch costs behave for other paradigms. The constraints of locality and periodicity will also apply to (physical systems implementing) other machines in the hierarchy, such as push-down automata, RAM machines, or Turing Machines. We would expect that variants of the results concerning σ_{mod} and σ_{mar} presented here also apply to those systems. However, there will also be important differences. For example, the overwriting of input and/or memory that occurs in machines more powerful than DFA will affect σ_{mod} in ways not considered in this paper. Moreover, Turing machines and push-down automata have access to an infinite memory. DFA, by definition, do not—indeed, it is this restriction that divides regular languages into low- and high-cost.

Finally, it is interesting to consider how the consequences of locality and periodicity relate to other resource costs. Recent work on transducers—a computational machine that generates an output corresponding to a hidden Markov model—has shown that a quantum advantage exists over a classical implementation if and only if the machine is not locally invertible [40]; it is unclear whether a similar result holds for DFA. The role of the input distribution in determining the thermodynamic costs in our work is also reminiscent of the way computational complexity depends on the distribution over inputs.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://doi.org/10.5281/zenodo.7950129.

Acknowledgment

T E O was supported by a Royal Society University Research Fellowship, and D H W was supported by US NSF Grant CHE-1648973. D H W also thanks the Santa Fe Institute for support.

Author contributions

T E O and D H W conceived the project. T E O performed the analysis. T E O and D H W interpreted the results and wrote the manuscript.

ORCID iDs

Thomas E Ouldridge https://orcid.org/0000-0001-8114-8602
David H Wolpert https://orcid.org/0000-0003-3105-2869

References

[1] Szilard L 1964 *Behav. Sci.* **9** 301
[2] Landauer R 1961 *IBM J. Res. Dev.* **5** 183
[3] Bennett C H 1982 *Int. J. Theor. Phys.* **21** 905–40
[4] Seifert U 2012 *Rep. Prog. Phys.* **75** 126001
[5] Parrondo J M R, Horowitz J M and Sagawa T 2015 *Nat. Phys.* **11** 131
[6] Wolpert D H 2019 *J. Phys.* A **52** 193001
[7] Wolpert D H, Kolchinsky A and Owen J A 2019 *Nat. Commun.* **10** 1727
[8] Wolpert D H and Kolchinsky A 2020 *New J. Phys.* **22** 063047
[9] Kolchinsky A and Wolpert D H 2021a *Phys. Rev.* E **104** 054107
[10] Kolchinsky A and Wolpert D H 2017 *J. Stat. Mech.* 083202
[11] Riechers P M and Gu M 2021a *Phys. Rev.* E **103** 042145
[12] Boyd A B, Mandal D D and Crutchfield J P 2018 *Phys. Rev.* X **8** 031036
[13] Wolpert D H 2020 *Phys. Rev. Lett.* **125** 200602
[14] Brittain R A Jones N S and Ouldridge T E 2021 arXiv:2102.03388
[15] Kolchinsky A and Wolpert D H 2020 *Phys. Rev. Res.* **2** 033312
[16] Strasberg P, Cerrillo J, Schaller G and Brandes T 2015 *Phys. Rev. E* **92** 042104
[17] Mandal D and Jarzynski C 2012 Proc *Proc. Natl Acad. Sci. USA* **109** 11641
[18] Boyd A B, Mandal D and Crutchfield J P 2016 *New J. Phys.* **18** 023049
[19] Brittain R A, Jones N S and Ouldridge T E 2019 *New J. Phys.* **21** 063022
[20] Arora S and Barak B 2009 *Computational Complexity: A Modern Approach* (Cambridge University Press)
[21] Sipser M 1996 *ACM SIGACT News* **27** 27
[22] Gao C Y and Limmer D T 2021 *Phys. Rev. Res.* **3** 033169
[23] Freitas N, Delvenne J-C and Esposito M 2021 *Phys. Rev.* X **11** 031064
[24] Kolchinsky A and Wolpert D H 2021b *Phys. Rev.* X **11** 041024
[25] Thain D 2019 *Introduction to Compilers and Language Design* (Lulu. com)
[26] Hopcroft J E and Ullman J D 1979 *Introduction to Automata Theory, Languages and Computation* (Addison-Wesley)
[27] Moore C 2019 arXiv:1907.12713
[28] Li M and Paul V 2008 *An Introduction to Kolmogorov Complexity and Its Applications* (Springer)
[29] Grunwald P and Vitányi P 2004 arXiv:cs/0410002
[30] Cover T M and Thomas J A 2006 *Elements of Information Theory* (Wiley-Interscience)
[31] Ouldridge T E 2018 *Nat. Comput.* **17** 3
[32] Owen J A, Kolchinsky A and Wolpert D H 2019 *New J. Phys.* **21** 013022
[33] Riechers P M and Gu M 2021b *Phys. Rev.* A **104** 012214
[34] Kardeș G and Wolpert D H 2022 arXiv:2206.01165

[35] Ouldridge T E and Wolpert D H https://zenodo.org/records/7950129 *zenodo* 2023
[36] Ganesh N and Anderson N G 2013 *Phys. Lett.* A **377** 3266
[37] Korbel J and Wolpert D H 2021 arXiv:2103.08997
[38] Wolpert D H 2015 arXiv:1508.05319
[39] Nerode A 1958 *Am. Math. Soc.* **9** 541
[40] Elliott T J 2021 *Phys. Rev.* A **103** 052615