

The Thermodynamic Complexity of Circuits*

Anne Condon^{†1}, David Doty^{‡2}, Niels Kornerup^{§3}, David Soloveichik^{¶4}, and David Wolpert^{||5}

¹University of British Columbia

²University of California, Davis

³Sandia National Laboratories

⁴University of Texas, Austin

⁵Santa Fe Institute, New Mexico; Complexity Science Hub, Vienna; Arizona State University, Arizona; International Center for Theoretical Physics, Trieste

Abstract

When any physical system executes a logically irreversible (non-injective) step of a computation, the excess heat released over the so-called Landauer limit is an energetic cost that cannot be recovered. In modern computers this cost vastly dwarfs the Landauer limit. Modern stochastic thermodynamics refers to this cost as the “entropy production” and has derived some very powerful lower bounds on its value, which strictly strengthen the second law of thermodynamics. Here we introduce a model of circuits using *heat functions* and investigate its entropy production, deriving new results on the thermodynamic complexity of circuits.

In our model, the circuits compute *cleanly*, erasing their inputs and all intermediate values. We first show that computing the optimal heat functions for gates in a given circuit (the gates that minimize total entropy production) when the circuit’s input is uniformly random is PP-complete. We also show that multiplicative approximation of those heat functions is NP-hard, but additive approximation admits a polynomial-time randomized algorithm. We give tight bounds for entropy production for certain specific families of circuits (on uniformly distributed inputs): an n -ary AND tree of depth $O(\log n)$ requires $\Omega(n)$ circuit entropy production, while a serial chain achieves $O(1)$, showing an energy-depth tradeoff. The Batcher odd-even mergesort network on n bits has $\Theta(n)$ entropy production, asymptotically less than its $\Theta(n \log^2 n)$ gate count. On random permutations of distinct integers its entropy production is $\Theta(n \log^2 n)$, asymptotically exceeding the Landauer cost of sorting; but preceding it with Leighton and Plaxton’s almost-sorting network reduces the total heat released to $O(n \log n)$. Finally, we show that a clean sorting circuit (not a sorting network) can sort n values with zero circuit entropy production over any exchangeable input distribution. Our framework also yields streamlined proofs of known results in stochastic thermodynamics on mismatch cost and modularity cost.

*A preliminary version of this paper appeared in chapter 6 of Niels Kornerup’s PhD thesis [Kor25]. This version substantially improves, corrects, and builds on the results in that chapter.

[†]Condon was supported by an NSERC Discovery grant.

[‡]Doty was supported by NSF awards 2329909 and 2211793 and DoE award DE-SC0024467.

[§]Kornerup was supported by the LDRD Program at Sandia National Laboratories. Sandia is managed and operated by NTESS under DOE NNSA contract DE-NA0003525.

[¶]Soloveichik was supported by a Schmidt Sciences Polymath award.

^{||}Wolpert was supported by NSF award 2626044.

Contents

1	Introduction	2
1.1	Landauer’s principle and logical reversibility	3
1.2	Beyond logical reversibility: Stochastic thermodynamics	6
1.3	Prior work on thermodynamics of circuits	9
1.4	Our results	10
2	Preliminaries	14
2.1	Our model	16
3	Deriving optimal heat functions	22
4	The complexity of optimizing heat functions in a given circuit	23
4.1	PP-complete language for optimizing heat functions	24
4.2	Approximation	25
5	Bounds on thermodynamic costs of fixed circuits	28
5.1	Warm-up: entropy-depth tradeoff for computing AND	29
5.2	Batcher’s odd-even mergesort network	30
5.2.1	Binary inputs	31
5.2.2	Arbitrary inputs	36
5.3	A more thermodynamically efficient sorting network	44
6	Clean zero entropy circuits for compaction and sorting	46
6.1	A clean compaction circuit with zero entropy production	46
6.2	A clean sorting circuit with zero entropy production	51
7	Conclusion	53
8	Acknowledgments	55
8.1	Discussion of substantive LLM use	55
A	Basic lemmas	61
B	Analyzing entropy flow in Batcher’s sorting network: overwriting model with binary inputs	62
C	Re-deriving mismatch costs	65
D	Entropy production from modularity	67

1 Introduction

A central part of the industrial revolution of the 19th century was the invention of steam engines. This invention resulted in a practical demand for powerful and efficient steam engines, which in turn ignited a mathematical revolution in physics: the development of the theory of thermodynamics to

understand and quantify how efficiently an engine could accomplish mechanical objectives such as moving a large train.

Then in the 20th century there was a similarly revolutionary invention, of computers. Accordingly, the energy required to accomplish *computational* objectives has been a subject of academic inquiry for a century. A deeper and more powerful understanding of this thermodynamics of computation would not just be a purely theoretical advance. As a practical matter, we live in a world where the energetic costs of computing are critically important. The International Energy Agency estimates that 485 terawatt-hours was consumed running data centers in the year 2025 and projects that this number will roughly double to 950 terawatt-hours in 2030 [Int26]. Despite this, our theoretical understanding of the fundamental thermodynamic costs of computation is rather limited, partially due to the lack of a strong and flexible model that captures many of the nuances of the thermodynamics of computation beyond logical reversibility. While current computational devices expend energy at rates well above those mandated by the laws of thermodynamics, as technology improves a principled understanding of thermodynamics is key to optimizing the energetic costs of computation.

However, most of the study of the thermodynamics of computation to date has been based on classical thermodynamics principles. While the field of thermodynamics itself has continued to evolve in recent decades, the study of the thermodynamics of computation has not kept pace. In particular, there has been a revolution in non-equilibrium statistical physics in the 21st century, resulting in what is now called *stochastic thermodynamics*. Whereas classical thermodynamics is, in its foundations, concerned only with systems at equilibrium, stochastic thermodynamics focuses on systems arbitrarily far from equilibrium. This makes it the ideal formalism for analyzing the thermodynamics of computation.

This paper serves two purposes: First, we introduce such a new model for investigating the thermodynamics of circuits. Our goal with this model is to strike a balance between broad accessibility to a general computer science audience and correctness with respect to stochastic thermodynamics, which represents our current best understanding of the fundamental energetic cost of machines, including computing machines. Second, we show how this circuit model can be leveraged to prove a wide range of new results ranging from the computational complexity of optimizing Boolean circuits to asymptotically tight lower and upper bounds on the thermodynamic costs of fixed circuits.

We start by giving a detailed overview of the history of the thermodynamics of computation before diving into our model and results.

1.1 Landauer’s principle and logical reversibility

Most of the computer science literature on the fundamental energetic costs of computation studies *reversible computing*,¹ This is an important and groundbreaking body of work, which we now summarize. Suppose a computing device executes a basic computational step that is *logically irreversible*, meaning that the device goes from logical state c_0 (such as the configuration of a multitape Turing machine, for instance) to logical state c_1 , where c_1 has *another* predecessor state c'_0 that also transitions to c_1 in one step. Formally, the global transition function $\delta : C \rightarrow C$ mapping the machine’s current state to its next state is not injective. Landauer [Lan61] argued that this necessarily decreases entropy within the computing device, entropy must correspondingly increase

¹Notable exceptions abound in the physics literature, e.g., [WK20, YCW26, YYW26, Wol19a, WKL⁺24], but these papers generally use physics terms and assumptions that are often difficult for computer scientists to read.

elsewhere in the universe to satisfy the Second Law of thermodynamics. Phrased in terms of *work* (i.e., accessible energy), *Landauer’s principle* states that *logically irreversible operations must expend work in order to release heat into their environments*.² Given a transition from n equally likely predecessor states to m equally likely successors, $k\mathbb{T} \ln(n/m)$ heat must be released, regardless of the physical implementation of the operation. Here $\mathbb{T}$ denotes the system’s temperature³ and k is Boltzmann’s constant.⁴

Landauer argued [Lan61] that most real computation would necessarily execute many irreversible operations, implying an unavoidable cost of $\Omega(tk\mathbb{T} \ln 2)$ to execute t steps of a computation regardless of the physical implementation of the computing device. Bennett [Ben73] challenged this assumption, showing how to convert an arbitrary Turing machine M into a logically reversible Turing machine R computing the same function. The configuration graph of R is then a 1D line $c_0 \rightarrow c_1 \rightarrow \dots \rightarrow c_t$. A completely unbiased random walk on this graph, requiring no work to bias it forwards, would then visit final configuration c_t in $O(t^2)$ time on average.⁵

Bennett’s original construction [Ben73] simulates a t -step Turing machine in $O(t)$ steps but pays with an increase in memory complexity up to $\Theta(t)$, essentially needing to write down the entire history of the computation on a separate worktape to maintain logical reversibility, bringing the memory use up to asymptotically match the time. (Thus a polynomial-time, log-space computation, for example, pays exponentially extra memory to make the computation logically reversible.) A series of papers over the next few decades [Ben89, LS90, LV96, LTV98, LMT00, BTV01] showed various improved simulation time/space tradeoffs in reversible Turing machines; see [Ben82, Vit05, Mor08] for useful surveys. Toffoli and Fredkin [Tof77, Tof80, FT82] pioneered the study of reversible computation in other computational models such as cellular automata and circuits. Meanwhile in the physics community, there has been work on understanding the thermodynamics of logically reversible Turing machines [SCSB15b, BJO21].

So what is there to say about the fundamental thermodynamic costs of computation that has not already been said in the reversible computing literature? First, we may consider a situation in which we do not have access to a logically reversible gate set, in which we wish to understand the fundamental energetic costs associated with whatever computational devices we do have on hand. While there has been progress towards actually building logically reversible computational devices (e.g. [AFK⁺98, Fra02, FBT⁺20]), these systems are not yet ready for widespread use and most computation today is performed in a highly irreversible manner. This is a deep research issue

²In this paper we use the term “release heat” mostly synonymously with “dissipate heat”, but generally reserve the term “dissipate” to mean a loss of heat/work/useful energy that is irreversible. However, we are careful to include a descriptor such as “irreversible” or “irrecoverable” when we want to identify that the loss of heat/energy cannot be undone; absent this descriptor, any release/transfer of heat/work/useful energy in principle may be recovered.

³In physics, it is common to specify a *heat bath* of temperature $\mathbb{T}$. For our purposes, we can think of $\mathbb{T}$ as the temperature of both the system and its environment.

⁴Throughout this paper, we choose the units for measuring energy so that $k\mathbb{T} \ln 2 = 1$. So we measure energy and heat in comparable units to bits.

⁵Bennett [Ben73] observed that readout from such a device would be impractical, since even after waiting t^2 time, there is only a $1/t$ probability of observing R in its final output configuration. He proposed dissipating small $\varepsilon \ll k\mathbb{T} \ln 2$ energy per step to bias the computation forward just enough to ensure a constant probability of finding the machine in an output configuration. This still implies $\varepsilon t = \Omega(t)$ energy cost for a t -step computation. See [DKL⁺24] for arguments that this total cost need not scale linearly with t , essentially by using algorithmic design techniques to increase the probability of observing the machine in an output configuration even with totally unbiased transitions, obviating the need to supply energy for even a slight forward bias. However, this trick (and the original unbiased walk outlined in [Ben73]) doesn’t account for the thermodynamic costs of resetting the device (see [Nor13, SCSB15a]). In [DKL⁺24] the authors avoid these costs by designing a device that never needs to be reset.

in its own right, and more practically, reflects the situation in almost every deployed computational system.

Second, over the last few decades the field of non-equilibrium statistical physics has undergone major developments, culminating in a new subfield called *stochastic thermodynamics* [VdBE15, Sei25]. These developments have upended our understanding of the thermodynamics of physical systems that implement computation. In particular, they reveal that the traditional understanding of logical reversibility and Landauer’s principle is at best partially correct: while the Second Law does indeed imply that decreasing entropy of a system locally requires increasing entropy outside the system, nothing in the Second Law says that this must be a physically irreversible loss of work.

To give a simple example of a logically irreversible process that can be done in a physically/thermodynamically reversible way (i.e., can be done with arbitrarily little dissipated work), suppose we have a particle in a cylinder, and identify the particle being in the top half of the cylinder as a bit 1, while the particle being in the bottom half is identified as bit 0, with equal probability. So if we compress the cylinder to its bottom half using a piston, we perform a bit erasure, since afterwards the particle is guaranteed to be in the bottom half. Moreover, in first-year physics we learn that if this compression is done sufficiently slowly, then all the work expended in that compression can be recovered afterward simply by *uncompressing* the cylinder sufficiently slowly, returning it to its original volume, and returning the particle’s distribution to equally likely to be in either half (see Szilard engines for more details, [Szi29, KMPA14]).

This is precisely the property that defines a process as being thermodynamically reversible. In contrast, if one compresses the cylinder too quickly, no matter how one subsequently uncompresses the cylinder, one won’t recover all the work that was used in the compression. This is what is meant by thermodynamic irreversibility. This example illustrates that logical and thermodynamic reversibility are independent properties of a process; it is possible to have one without the other. (See [Sag14] for a pedagogical elaboration of this point.)

Third, if the function f being computed is not injective, it is not possible to compute the mapping $x \mapsto f(x)$ in a logically reversibly way; instead reversible computing models typically compute $x \mapsto (x, f(x))$. But the mappings $x \mapsto f(x)$ and $x \mapsto (x, f(x))$ are fundamentally different. If f is non-injective or if f^{-1} is prohibitively expensive to compute (i.e., if f is a one-way function), then it is impossible to efficiently and reversibly obtain just $f(x)$ from the pair $(x, f(x))$.⁶ One might think it is okay to just leave an extra copy of x in your computer’s memory, but if you compute f repeatedly and old values of x are overwritten, the cost of erasing these old values must be paid (see [Wol19a, Sec. 11]). One might hope to store values of x generated by intermediate steps of a larger logically reversible computation and reversibly uncompute them during the uncomputation step, however, temporarily storing x increases the space complexity of the overall computation (see [Ben89]).

Finally, it remains open whether every time- t , space- s Turing machine can be simulated by a

⁶Bennett [Ben73] shows that, if f is injective, it is possible to compute the mapping $x \mapsto f(x)$ reversibly (i.e., not needing the input also to appear as part of the output). However, this construction has time complexity as high as the max of computing f and computing f^{-1} , since it involves taking a direct computation of f^{-1} by a Turing machine and making it logically reversible in the same way that f is reversibly computed. But if we want the time complexity to be no worse than that of computing f , this rules out one-way functions in which f^{-1} has time complexity fundamentally greater than that of f . Indeed, it is well-known folklore that the mapping $x \mapsto f(x)$ cannot be efficiently computed in a logically reversible manner if f is a one-way function: if such a computation, for example a Boolean circuit, were completely logically reversible, starting only with x and ending only with $f(x)$, then the mirror image of such a circuit would be an equally efficient circuit computing f^{-1} , contradicting that f is one-way.

reversible Turing machine using $O(t)$ time and $O(s)$ space; the current best results asymptotically increase at least one of them [Vit05], and there is an oracle relative to which this is necessary [FA17].

In this paper we start to investigate how stochastic thermodynamics can help untangle these issues.

1.2 Beyond logical reversibility: Stochastic thermodynamics

Bennett’s approach of requiring logical reversibility of every computational step is in one sense a very ambitious way to minimize energy costs by pushing the so-called *Landauer cost* of executing logically irreversible operations to 0, simply by removing all such operations. Li and Vitányi [LV96] investigated how it is possible to trade time and space for ‘energy’ by limiting the number of logically irreversible steps in a computation. Demaine, Lynch, Mirano, and Tyagi [DLMT16] built on this idea and proposed a generalization of logical reversibility they call *semi-reversible computing*, in which irreversible operations $f : D \rightarrow R$ are permitted and assumed to cost $\log \frac{|D|}{|R|}$ bits of energy to execute (this being the natural generalization of the Landauer cost of 1 bit when $|D| = 2$ and $|R| = 1$, assuming a uniform distribution on D). The total energy cost (by this definition) is then treated as a complexity measure to be minimized.

However, the more modern field of *stochastic thermodynamics* [VdBE15, Sei25] has generalized our understanding of the interaction of entropy and fundamental energy use in physical devices, beyond the overly simplistic assumption that each bit erasure costs $kT \ln 2$ energy. It has been argued [WKL⁺24] that stochastic thermodynamics can be fruitfully applied to computational complexity to better understand the relationship between fundamental thermodynamic costs and other traditional complexity measures such as time and space.

We now review *entropy production*, a basic concept from modern stochastic thermodynamics that lets us study the fundamental limits of energetic costs even for logically irreversible computation. As applied to the study of computation, we propose entropy production as a computational complexity measure/cost to be minimized. This is already the measure of interest for such systems in the physics community (i.e. [Wol19a, PP21, WKL⁺24, Sei25]). We then analyze these costs in a particular circuit model, noting that our approach can be extended in a natural way to other models of computation.

We provide our full definitions in Section 2 and give a semi-formal summary of the underlying concepts here. Briefly, the generalization of the Second Law of thermodynamics most central to our purpose is the equation

$$\text{EP} = \text{Q} - (\mathcal{S}(\mathcal{D}_0) - \mathcal{S}(\mathcal{D}_1)). \quad (1)$$

Equation (1) appears later in Definition 2.24 with notation specialized to the study of the thermodynamics of circuits. The terms EP and Q respectively mean *entropy production* and *entropy flow*. However, throughout this paper we focus on the case of a single (thermal) reservoir. That leads us to use the term *heat released* for Q. Furthermore, given our choice of units such that $kT \ln 2 = 1$, we can equate information in bits with equal units of heat; we discuss these terms below.

To begin, we focus on the term $\mathcal{S}(\mathcal{D}_0) - \mathcal{S}(\mathcal{D}_1)$ in Eq. (1). $\mathcal{D}_0$ represents the probability distribution on states of a physical device⁷ prior to executing a (possibly logically irreversible) step,

⁷The stochastic thermodynamic perspective can be viewed as an “intrinsically distributional” complexity measure because it defines costs that are fundamentally tied to a distribution over inputs. In other words, there is not in general a sensible notion of “worst-case entropy production” defined as $\max_{x \in \{0,1\}^n} \text{EP}(x)$, as is typical for definitions

and $\mathcal{D}_1$ represents the associated distribution after; $\mathcal{S}(\mathcal{D}) = \sum_{x \in \text{supp}(\mathcal{D})} \Pr[x] \log \frac{1}{\Pr[x]}$ is the *Shannon entropy* of $\mathcal{D}$.⁸

Concretely, suppose we perform the most widely discussed logical operation in the thermodynamics of computing: erasing a single bit $b \in \{0, 1\}$. Here “erasing b ” generally means “setting b to 0”, i.e., no matter whether b was 0 or 1 before, afterward it is 0 and so the system has “erased” any knowledge of b ’s previous value. In this case, $\mathcal{D}_1$ is the degenerate point-mass distribution with $\Pr_{\mathcal{D}_1}[b = 0] = 1$. The prior distribution $\mathcal{D}_0$ may be a number of things, but the worst case in terms of logical irreversibility is the uniform distribution $\mathcal{D}$ with $\Pr_{\mathcal{D}_0}[b = 0] = \Pr_{\mathcal{D}_0}[b = 1] = 1/2$, i.e., prior to erasing the bit, we have maximal uncertainty about its current value. The term $\mathcal{S}(\mathcal{D}_0) - \mathcal{S}(\mathcal{D}_1)$, the difference in Shannon entropy between distributions $\mathcal{D}_0$ and $\mathcal{D}_1$, is the so-called *Landauer cost*.⁹ In this example, $\mathcal{S}$ is the simple binary entropy function on $p = \Pr_{\mathcal{D}_i}[b = 0]$, so $\mathcal{S}(\mathcal{D}_0) = 1$, $\mathcal{S}(\mathcal{D}_1) = 0$, and the Landauer cost is 1 bit of entropy. Importantly, (1) means that the Landauer cost does not necessarily imply positive entropy production. By focusing instead on entropy production, we open the door to modeling and analyzing sources of heat dissipation in computation, over and above the Landauer cost, that cause EP to be positive.

The other two terms EP and Q account for how this local drop in entropy relates to the outside universe. EP, the *entropy production*, represents the total increase in entropy of the universe. The Second Law dictates that $\text{EP} \geq 0$. This is related to the *local* decrease in entropy $\mathcal{S}(\mathcal{D}_0) - \mathcal{S}(\mathcal{D}_1)$ by the *heat released* Q.¹⁰ Q corresponds to the entropy change of the environment due to heat exchange (releasing heat increases the entropy of the environment) and encompasses the underlying physical implementation of our computation. Note that to maintain the Second Law ($\text{EP} \geq 0$), we must have $\text{Q} \geq \mathcal{S}(\mathcal{D}_0) - \mathcal{S}(\mathcal{D}_1)$. If the local entropy strictly dropped through a logically irreversible step, i.e., if $\mathcal{S}(\mathcal{D}_0) > \mathcal{S}(\mathcal{D}_1)$, then in this case Q must be strictly positive to maintain $\text{EP} \geq 0$.

The ability of randomized computation to have a negative Landauer cost has implications for energy use as well, allowing a computation to use randomized operations to counteract the heat released in other steps by logically irreversible operations. As a simple example of these principles in action, and to illustrate how the laws of physics allow for computation with zero dissipated work *even if* a logically irreversible step such as erasing a bit is performed, consider a situation in which we first generate a random bit and then erase it. Imagine we have some physical sort of randomness/entropy available to generate a random bit $b \in \{0, 1\}$. Suppose we generate this bit in

of worst-case time or space complexity.

⁸For example, if the system has two bits x_1, x_2 each i.i.d. 0 or 1 with probability $\Pr_{\mathcal{D}_0}[x_1 = 1] = \Pr_{\mathcal{D}_0}[x_2 = 1] = 1/2$, and if the system computes $y = \text{AND}(x_1, x_2)$, then implied output distribution $\mathcal{D}_1$ on y is $\Pr_{\mathcal{D}_1}[y = 1] = 1/4$ and $\Pr_{\mathcal{D}_1}[y = 0] = 3/4$.

⁹Much literature on reversible computing assumes the Landauer cost of erasing one bit is, depending on units, 1 bit of entropy or $kT \ln 2$ units of heat. (In this paper we will use units of entropy, i.e., bits.) However, this assumes $\mathcal{D}_0$ is the worst-case uniform distribution; for other distributions the drop in entropy is strictly less than one bit: it is precisely the binary entropy $H_2(p)$ if one of the bits has probability p . As an extreme example, if $\Pr_{\mathcal{D}_0}[b = 0] = 1$, then the Landauer cost is 0. Indeed, obviously it costs nothing to “set” a bit to 0 when one knows for certain that it is already 0; no action at all is required for this “computation”. More generally, Frank [Fra17] points out that reversible computing need not require a function f to be fully injective, but only injective on inputs with positive probability.

¹⁰The physics literature often refers to “the environment” as a “heat bath” or “heat reservoir”. We note that the term “heat” in theoretical physics does not necessarily imply anything actually getting hotter, and even in that community there can be forms of entropy flow not interpretable as heat transfer. In principle Q does not necessarily represent heat dissipation. It could correspond to discharging a rechargeable battery or increasing degrees of freedom that do not correspond to computational states (i.e. the device might write bits to a tape outside of the computer). However, in most physical devices, *released heat* is the form entropy flow takes, and we will use this term synonymously for entropy flow, especially since it is less easily confused with entropy *production*.

a memory location where we know in advance that a 0 is already written. Applying equation (1) to the first step, we have that $\mathcal{D}_0$ is the point-mass $\Pr_{\mathcal{D}_0}[b = 0] = 1$, whereas afterward we have $\Pr_{\mathcal{D}_1}[b = 0] = \Pr_{\mathcal{D}_1}[b = 1] = 1/2$. Thus the Landauer cost $\mathcal{S}(\mathcal{D}_0) - \mathcal{S}(\mathcal{D}_1) = -1$ is actually *negative*. Physically, this corresponds to the fact that in principle, one removes $kT \ln 2$ heat from the environment to randomize this bit, releasing the same amount of heat on the subsequent erasure step that takes us from $\mathcal{D}_1$ (b is equally likely to be 0 or 1) back to $\mathcal{D}_0$ ($b = 0$). This is essentially “taking a loan and then paying it back”.

In the remainder of this paper, we focus on deterministic computation, in which every state has at most one successor. Our model may describe only part of a larger physical device: other parts may perform randomized operations with negative Landauer cost and draw heat from the environment to offset heat released by the deterministic computation. When $f : D \rightarrow R$ is a (deterministic, single-valued) function being computed, we let $\mathcal{L}_f(\mathcal{D})$ (Definition 2.19) denote $\mathcal{S}(\mathcal{D}) - \mathcal{S}(f(\mathcal{D}))$, where $f(\mathcal{D})$ denotes the distribution on R induced by distribution $\mathcal{D}$ on D . In this case, the data processing inequality (Definition 2.5) implies that $\mathcal{L}_f(\mathcal{D}) \geq 0$; i.e., a deterministic process cannot increase the entropy of a distribution.

The previous example shows that a logically irreversible operation may have zero entropy production. Next we discuss a “dual” example: an overall operation can be logically reversible, yet require positive entropy production due to “locality”, i.e., due to the fact that portions of the computation must be done in physically separate places, so that despite entropy staying constant globally, local operations must produce entropy due to lack of knowledge of rest of the state of the system. Consider three entities Alice, Bob, and Eve. Alice and Bob are at physically separate locations with no ability to communicate. Eve has a uniformly random bit b , and sends a copy of b to each of Alice and Bob. Alice must erase her bit, setting it to 0. Globally, this is logically reversible, since the mapping $(b, b) \mapsto (0, b)$ is injective, but since Alice “locally forgets” b , she must release a unit of heat into the environment to erase her bit, even though far-away Bob remembers b . Though this example may seem contrived and hand-wavey, it is in fact a good example of what we sometimes called *modularity cost* (also known as *modularity mismatch cost* in the stochastic thermodynamics literature, e.g., [Wol19b]): the fact that a self-contained computational module, such as a gate in a circuit, must release heat to compute an irreversible *local* operation, despite the fact that elsewhere in the circuit, information about the gate’s input is retained and so globally, the operation is reversible.

To summarize, (1) applied to deterministic computation of a function f obeys $\text{EP} = \text{Q} - \mathcal{L}_f(\mathcal{D})$. $\mathcal{L}_f(\mathcal{D})$ is a purely information theoretic term depending only on f and the input distribution $\mathcal{D}$: the loss of entropy $\mathcal{S}(\mathcal{D}) - \mathcal{S}(f(\mathcal{D}))$ between the input distribution $\mathcal{D}$ to f and the induced output distribution $f(\mathcal{D})$. Reversible computing literature is concerned with making $\mathcal{L}_f(\mathcal{D}) = 0$ for every computational step; in this paper we allow it to be positive, though we freely and regularly use reversible computing techniques as well. Q is the heat released into the environment during this computation; this depends on the physics of the device computing f . The Second Law of Thermodynamics states that $\text{EP} \geq 0$, so since $\text{EP} = \text{Q} - \mathcal{L}_f(\mathcal{D})$, this implies $\text{Q} \geq \mathcal{L}_f(\mathcal{D})$. From a complexity-theoretic perspective, we wish to minimize EP , since it represents a physically irreversible loss of work (useful energy). However, given a fixed f to compute and distribution $\mathcal{D}$, we have no control over $\mathcal{L}_f(\mathcal{D})$, so minimizing Q is equivalent to minimizing EP .

1.3 Prior work on thermodynamics of circuits

One perspective on the thermodynamic costs of circuits is provided by works of Michael Frank [Fra17, Fra18]. In [Fra17], Frank generalizes the conventional computer science notion of logical reversibility to include operations that, while not reversible in general, are so under a restricted subset of their inputs. He denotes such operations as being “conditionally reversible” and shows that they can be implemented to release arbitrarily little heat, so long as the gates never receive an input violating the reversibility precondition. Frank is primarily interested in conditional reversibility since it gives logically reversible gates that may be easier to implement than networks of Toffoli gates. Our work can be seen as a further generalization of this idea; in addition to allowing conditionally reversible gates to release zero expected heat on valid distributions over inputs, we consider gates that are finely tuned for the actual distribution over inputs they expect to receive. In [Fra18], Frank also points out that Q is not quite the correct measure of thermodynamic cost. He points out that the thermodynamic costs associated with Landauer’s principle is not merely the fact that Q must be positive. Rather it is that information which is dissipated as heat will rapidly become unrecoverable to the computational process, resulting in something similar to what we call a modularity cost. Frank justifies the study of logically reversible computing since it is a sufficient condition to avoid Landauer related thermodynamic costs.

Prior work by Wolpert, Kolchinsky, Caravelli, Yadav, and Yousef [WK20, YCW26, YYW26] investigated the thermodynamics of Boolean circuits using the theorems involving *mismatch cost* — a metric that quantifies the additional entropy production incurred by running a process on a different input distribution than the “ideal” one for given hardware. In this case, the physics of the device computing the gate may, in a sense, be “summarized information-theoretically” by an *optimal* input distribution $\mathcal{D}^*$, the distribution that minimizes the entropy production according to (1). They show lower bounds on the entropy production induced by running a circuit on a (potentially mismatched) input distribution $\mathcal{D}$, where the lower bound depends only on the logical behavior of the gates with respect to an optimal input distribution $\mathcal{D}^*$. In other words, it is possible to prove non-trivial lower bounds on the EP of a circuit without knowing any details of the physical implementation of its gates other than their optimal distributions over its inputs.¹¹

The perspective of these works is that we know an optimal input distribution for a gate, and proceeds from there to measure mismatch costs (which are a lower bound on EP). Mismatch costs are a powerful tool for analyzing thermodynamic costs without needing to specify the underlying physics, but they are cumbersome to work with. The thermodynamic accounting they provide requires viewing everything through the lens of distributions over state and costs are neither linear (with respect to the probabilities of inputs) nor local to the gates performing computation. Moreover, the optimal distribution for a gate is a counter-intuitive way to specify its behavior and such a distribution is not unique even when the dynamics of the gate are deterministic.¹²

Nevertheless, these methods are particularly nice as with mild assumptions on the behavior of gates in a circuit (i.e. modularity or time-invariant behavior), they can prove entropy lower bounds that are truly independent of the underlying hardware.

¹¹The physical implementation does influence what is the optimal distribution. The significance of mismatch cost is that it can be measured knowing *only* what is the optimal distribution, i.e., the physical implementation influences the mismatch cost *only* through how it influences what is the optimal distribution.

¹²There is a well defined way to handle this with mismatch costs, but it requires analyzing the pre-image of each possible output independently, but this can be messy to work with.

1.4 Our results

Here we provide a high level overview of our results and the techniques we use to obtain them.

The thermodynamic circuit model We introduce a model of circuits where each gate releases heat determined by its input value in order to perform logically irreversible computation. For a circuit to compute a function f , we require it to map each input state $(x, \vec{0})$ to output state $(f(x), \vec{0})$; the $\vec{0}$'s (which have different lengths if $|x| \neq |f(x)|$) denote “ancilla” wires that may be used as extra storage during the computation. In other words, the circuit must erase any “dirty” bits introduced during the computation including the original value of x . The amount of heat a gate releases can depend on that gate’s input; imagine that the gate must undergo some physical process to map each of its possible input states to the correct output. Thus, we can assign a “heat function” to each gate $Q : \{0, 1\}^n \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ that maps each input to an associated amount of released heat. Technically, the amount of heat released by this process may not be deterministic, so one should think of these values as being expected costs, though sometimes for brevity we may say a gate releases $Q(x)$ heat on input x , understood to mean *expected* heat. Thanks to the linearity of expectation, this does not make a difference in the details of our model. We say a heat function for a gate is *valid* if it obeys the Second Law of thermodynamics (i.e. the EP in (1) is always non-negative) on every marginal input distribution. From this perspective, we connect our model to mismatch costs in Appendix C of the appendix, and show that some of the results in [WK20, YCW26, YYW26] have more intuitive proofs using heat functions. We also show how our model can be used to prove new results about the thermodynamic costs of circuits.

Recall from Section 1.2 that our goal is to minimize EP, and that once $\mathcal{L}_f(\mathcal{D})$ is fixed, minimizing EP is equivalent to minimizing Q. Despite this equivalence once $\mathcal{L}_f(\mathcal{D})$ is fixed, when we prove a one-way bound, we generally prove lower bounds on EP (i.e., negative results stating that some computation *must* dissipate a certain minimum amount of work that cannot be recovered) and upper bounds on Q (i.e., positive results stating that it is *possible* to achieve a certain computation while only releasing a small amount of heat). For deterministic computation, $\text{EP} \leq Q$ so a lower bound on EP is also a lower bound on Q and an upper bound on Q is also an upper bound on EP. As is typical in theoretical computer science, we will use O -notation to suppress constant factors in our bounds. Note that constant factors are important to track in the case where $Q = \Omega(h(n))$ and $\mathcal{L}_f(\mathcal{D}) = O(h(n))$. Depending on the hidden constants, the EP for such a computation could be anywhere between 0 and $\Omega(h(n))$.

There are two important caveats with our approach in this paper. First, when discussing EP, we are implicitly assuming that the input distribution to our circuit is sampled independently from any larger context. If a copy of the input to a circuit is retained somewhere outside of the circuit, then the drop in entropy in (1) should actually be zero and we get that $\text{EP} = Q$. When proving lower bounds on EP this is a non-issue, as the additional context can only increase the EP. When proving upper bounds on EP, we specify that we are interested in the *circuit entropy production*, which we define as EP under this assumption on the input distribution. Second, when giving upper bounds on either the Q or the EP of a circuit, we are operating under the “axiom” that every gate could in principle be built with any valid heat function. While the validity constraint protects our gates from violating the Second Law of thermodynamics, other physical or engineering constraints may theoretically prevent such a gate from existing. For example, speed limit theorems [SFS18] imply that, with bounds on transition rates, EP must be strictly positive in order for gates to

compute their outputs in finite time¹³ and current CMOS technology is far from the thermodynamic limit imposed by validity. Thus our upper bounds should be interpreted under the “spherical cow” assumption that validity is the only constraint on realizable heat functions.

Deriving optimal heat functions In Section 3, Lemma 3.1 shows how to derive the optimal valid heat function for a known distribution over inputs. In particular if p_x is the probability of input x and q_y is the probability of output y , then any optimal heat function for computing f on this distribution over inputs must satisfy $Q(x) = \log(q_{f(x)}/p_x)$ on all x where $p_x > 0$. This in turn is a key ingredient in proving our results in Sections 4, 5, and 6.

Hardness results and approximation algorithms for computing optimal heat functions

Section 4 is concerned with computing the optimal heat functions for gates in a given Boolean circuit (so as to minimize EP), assuming the i.i.d. uniform distribution for the circuit’s inputs. We first show that the problem of computing the optimal heat function for each gate (so as to minimize total entropy production of the entire circuit) is unfortunately PP-complete (Lemmas 4.3 and 4.4). Since Lemma 3.1 shows how to compute the optimal heat function for a *single* gate and a given input distribution to that gate, this hardness follows essentially from the fact that computing the induced input distribution for a given gate in a circuit, knowing that the circuit’s inputs are i.i.d. with probability $1/2$, is itself PP-complete. The (straightforward) reduction is from the PP-complete language $\mathcal{L}_{\text{SAT}/2}$ of Boolean circuits $\mathcal{C}$ where at least half the assignments satisfy $\mathcal{C}$: by adding a single erasure gate g on the output, with an optimal heat function Q_g , $Q_g(1) \leq 1$ if and only if $\Pr[\text{output of } \mathcal{C} \text{ is } 1] \geq 1/2$, i.e., if and only if $\mathcal{C} \in \mathcal{L}_{\text{SAT}/2}$. Containment in PP directly follows from an algorithm that accepts or rejects depending on the relative probabilities a gate receives a specified input x or another input $x' \neq x$ where $f(x') = f(x)$.

Since PP-hardness rules out polynomial-time exact algorithms, this motivates the question of whether polynomial-time approximation algorithms exist. We show that this depends crucially on the form of error. With *multiplicative* error, i.e., for some constant $c > 1$, to compute heat functions with expected value at most c times optimal, then this is NP-hard, i.e., untractable unless $\text{NP} \subseteq \text{BPP}$. There is an easy argument for this: suppose we have a Boolean formula ϕ (a special type of circuit) and want to know whether ϕ is satisfiable. If we then add an erasure gate g to ϕ ’s output, then for an optimal heat function Q_g^* , if ϕ is unsatisfiable, $Q_g^*(0) = 0$, i.e., it costs 0 heat to erase the output since we know it is 0. If ϕ is satisfiable, then $Q_g^*(0) > 0$. Thus any approximation algorithm outputting heat value q will tell us whether ϕ is satisfiable depending on whether $q > 0$. However, this is a somewhat degenerate case, in the sense that *any* optimization problem permitting an optimal value of 0 has the property that no multiplicative approximation is good unless it is exact. The more interesting case is when a wire might have both bits 0 or 1 ($\text{supp}(\mathcal{D}) = \{0, 1\}$), so that we have a positive optimal expected heat $\mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)] > 0$. We can in fact prove NP-hardness of multiplicative approximation continues to hold even in this case (Lemma 4.7).

No NP-hard problem has a polynomial-time *randomized* algorithm unless $\text{NP} \subseteq \text{BPP}$;¹⁴ Lemma 4.7

¹³As is common in physics, we sidestep this issue by assuming that all gates are computed *quasistatically* (i.e. infinitely slowly). In the limit as time to implement a gate goes to infinity, this lower bound on the EP goes to zero. We could still implement all gates in a circuit in finite time while making this cost a small constant independent of the number of gates, but we choose to do things quasistatically to simplify our analysis.

¹⁴The famous and widely-believed conjecture that $\text{P} \neq \text{NP}$ is the statement that NP-complete problems such as SAT have no polynomial-time deterministic algorithms that are guaranteed to get the answer correct. The statement $\text{NP} \not\subseteq \text{BPP}$ generalizes this to the also widely-believed conjecture that such problems have no polynomial-time

implies this applies to the problem of approximating optimal heat functions with multiplicative error. However, considering *additive* error in approximation, we show there is a polynomial-time randomized algorithm that outputs a heat function at most ε *plus* optimal with probability of failure at most δ ; the running time is polynomial in the input, $1/\varepsilon$, and $1/\delta$ (Lemma 4.8). The idea is simply to sample uniformly random inputs to the whole circuit $\mathcal{C}$ to empirically estimate a distribution on inputs to a gate g in $\mathcal{C}$, from which a heat function is computed via Lemma 3.1.

Bounds on thermodynamic costs of fixed circuits Having shown the difficulty in computing optimal heat functions for arbitrary given circuits, Section 5 then analyzes particular circuits and derives both lower bounds on EP and upper bounds on Q. Section 5.1 begins with a warm-up exercise analyzing the computation of the n -input AND function $x_1 \wedge x_2 \wedge \dots \wedge x_n$ on uniformly distributed inputs in $\{0, 1\}^n$, using only input-preserving binary-AND gates of the form $\bigwedge(x_1, x_2, z) = (x_1, x_2, x_1 \wedge x_2)$ and erasure gates $E(x) = 0$. We show the (possibly) counterintuitive result that the most natural implementation, a tree of AND gates of depth $O(\log n)$, along with appropriate erasure gates to enable the circuit to cleanly compute the 1-bit output $x_1 \wedge x_2 \wedge \dots \wedge x_n$, has circuit entropy production $\Omega(n)$, whereas the “serial” variant with depth $\Theta(n)$, can be done with circuit entropy production that is only $O(1)$.

Section 5.2 then analyzes the $O(n \log^2 n)$ -gate, $O(\log^2 n)$ -depth Batcher even-odd mergesort sorting network on Boolean inputs. When applied to Boolean values, sorting networks are circuits where every gate is a 2-input, 2-output *comparator* gate: computing the identity on inputs 00, 01, 11, but on input 10, swapping the bits to output 01. The main results of this section, Theorems 5.2 and 5.4, show an $O(n)$ upper bound on Q using only $O(\log n)$ distinct heat functions and a matching $\Omega(n)$ lower bound on EP over a uniform distribution of inputs in $\{0, 1\}^n$ when using the optimal valid heat function on every gate. This bound is notable in that it is asymptotically smaller than the number $\Theta(n \log^2 n)$ of gates in the circuit, despite none of the gates performing logically reversible operations. The $O(n)$ upper bound (Theorem 5.2) is proven by slicing the network into regions of gates that all either (1) contain a small number of gates or (2) have one input that is overwhelmingly likely. In contrast, the $\Omega(n)$ lower bound (Theorem 5.4) follows directly from observing that the first 3 layers of the Batcher’s odd even mergesort network consist of $n/4$ independent copies of sorting networks on 4 bits each and that each of these networks has $\Omega(1)$ entropy production. In Appendix B we prove a similar upper bound on the Q for sorting bits in a more restricted overwriting model of circuits considered in other works (i.e., [YCW26, YYW26]).

Section 5.2.2 analyzes the more interesting case of the Batcher network applied to n distinct values drawn uniformly at random from an ordered set D . In this case, we are able to show that the entropy production must be $\Omega(n \log^2 n)$ which is tightly matched by a trivial $O(n \log^2 n)$ upper bound on Q given using any valid heat function where the expected released heat is $O(1)$ on all inputs. This result is rather technical to prove, but the key ingredient is Lemma 5.11, which shows that for a constant fraction of the gates in the network, if that gate receives input (x_1, x_2) with probability p_{x_1, x_2} then whenever $p_{a, b} > 0$ it must be the case that $1/\alpha \leq p_{b, a}/p_{a, b} \leq \alpha$ for some universal constant α . Since such gates must release $\Omega(1)$ expected heat each, we get an $\Omega(n \log^2 n)$ lower bound on Q. Since the drop in entropy associated with sorting is only $O(n \log n)$, we can conclude that EP is $\Omega(n \log^2 n)$ in Corollary 5.14.

Section 5.3 presents a straightforward method for constructing a sorting network with better thermodynamic costs than Batcher on n distinct inputs drawn uniformly at random from an ordered

randomized algorithms that are allowed to get the answer wrong with small probability.

set D . This circuit has Q that is $O(n \log n)$ which is asymptotically optimal.¹⁵ Our circuit works by first using an $O(n \log n)$ gate network developed by Leighton and Plaxton [LP98] that sorts all but a small fraction of the inputs. After running that circuit, we apply Batcher to its output. The result, as we show in Theorem 5.18, is a sorting network with $O(n \log^2 n)$ gates that only releases $O(n \log n)$ expected heat. By first using Leighton and Plaxton’s network, the input to Batcher’s will already be sorted most of the time. Thus we can use a heavily biased heat function for the gates in Batcher’s that releases less heat whenever the input is already sorted.

Compaction and sorting with zero entropy production Section 6.1 shows that we can do better than Batcher by designing a bit-sorting (a.k.a., *bit-compaction*) circuit specifically to minimize EP, and in fact can achieve zero circuit entropy production (Theorem 6.1) on any exchangeable distribution over inputs. Note that this construction is not a sorting network, as it uses ancilla wires and more general gates than only comparators.

We do so by using standard Bennett-style “compute/copy output/uncompute” (applied to circuits) to convert any circuit computing the mapping $x \mapsto f(x)$ to one reversibly computing $(x, \vec{0}) \mapsto (x, f(x))$,¹⁶ with only a constant increase in gate and depth complexity.¹⁷ Once we have $(x, f(x))$ in memory, we give a way to pick heat functions for erasing each bit of x such that the expected heat released is exactly the drop in entropy from mapping x to $f(x)$. The choice of heat function for erasing each x_i is determined based on the value of $f(x)$ and the suffix of x after index i , which can be computed and uncomputed in a logically reversible fashion even after x_i has been erased. We design such a circuit based on any sorting network C with $t(n)$ gates and depth $d(n)$ that compacts n inputs with m bit payloads using only constant arity gates with only $O(t(n)m + n^2)$ gates, $O(d(n) \log m + n \log n)$ depth, and $O(t(n) + nm)$ ancilla wires. In Section 6.2, we show how this approach can be extended to give zero circuit entropy production circuits for sorting on any exchangeable distribution over inputs.

New proofs of known results on mismatch cost and circuit mutual information using heat functions Following some basic lemmas in Appendix A of the appendix and the aforementioned analysis of Batcher’s sorting algorithm in Appendix B, we prove new results connecting heat functions to mismatch costs and reprove existing results on mismatch costs using the language of heat functions in Appendices C and D. This is intended to share these insights from stochastic thermodynamics with the computer science community and better connect our heat function model with prior work on the thermodynamics of circuits. Appendix C concerns *mismatch cost*, which is informally defined as the *extra* entropy production incurred by using a gate with a heat function optimal for distribution $\mathcal{D}^*$, when in fact $\mathcal{D} \neq \mathcal{D}^*$ is the true distribution. The mismatch cost of course depends on the physical implementation of the gate. Wolpert and Kolchinsky [WK20] showed that the way this physics influences the extra entropy production is entirely contained in its optimal distribution $\mathcal{D}^*$, in the sense that the mismatch cost is precisely the information-theoretic terms $D_{\text{KL}}(\mathcal{D}||\mathcal{D}^*) - D_{\text{KL}}(f(\mathcal{D})||f(\mathcal{D}^*))$, where D_{KL} refers to the Kullback-Leibler divergence, see Section 2. We show how this result admits a clean proof using heat functions.

¹⁵Technically [AKS83] automatically gives us this result since the network only has $O(n \log n)$ gates, but this network is known to have prohibitively large constant factors that render it impractical.

¹⁶We actually perform a slightly different logically reversible mapping for this problem to efficiently handle payloads, see the proof of Theorem 6.1 for more details.

¹⁷This is useful because, as mentioned, reversible computation is automatically zero EP assuming a heat function of $Q(x) = 0$ for all gates.

In Appendix D, we give a simple proof of a result of Wolpert and Kolchinsky [WK20], who showed that the entropy production of applying a gate in a circuit is at least the drop in mutual information between the wires in this gate and the state of the remaining wires of the circuit. We reprove this result (Proposition D.1) in the language of heat functions showing that it is an immediate consequence of the locality of valid heat functions, i.e., the fact that heat functions reference a single gate independently of the correlation of its inputs to other gates in the circuit.

2 Preliminaries

We use $\log$ to denote $\log_2$, and fix $2^{-\infty} = 0$, $-\log 0 = \infty$, as well as $z/0 = \infty$ when $z > 0$. Let $\mathcal{D}$ be a distribution over domain D and $x \in D$. We use $\Pr_{\mathcal{D}}[x]$ to denote the probability that a random variable $X \sim \mathcal{D}$ has value x (i.e. $\Pr_{X \sim \mathcal{D}}[X = x]$). Let $S \subseteq D$. We will also use $\mathcal{D}_{|S}$ to denote the distribution $\mathcal{D}$ conditioned on the sample being in S . More formally, $\Pr_{\mathcal{D}_{|S}}[x] = \Pr_{X \sim \mathcal{D}}[X = x | X \in S]$. When $S \cap \text{supp}(\mathcal{D}) = \emptyset$, we say that $\mathcal{D}_{|S}$ is undefined. We use $\mathbb{E}_{x \sim \mathcal{D}}[f(x)]$ to denote $\sum_{x \in \text{supp}(\mathcal{D})} \Pr_{\mathcal{D}}[x] f(x)$.

Definition 2.1. *The Shannon entropy of a distribution $\mathcal{D}$ is given by*

$$\mathcal{S}(\mathcal{D}) = -\mathbb{E}_{x \sim \mathcal{D}} \left[\log \Pr_{\mathcal{D}}[x] \right] = - \sum_{x \in \text{supp}(\mathcal{D})} \Pr_{\mathcal{D}}[x] \log \Pr_{\mathcal{D}}[x].$$

We use e_i to refer to a vector with the value 1 in the i 'th index and zero everywhere else. When thinking about function computation, we will often use D to refer to the (finite sized) domain and R to refer to the (finite sized) range. When needed, we will often specify $D = \Sigma^n$ and $R = \Sigma^m$ for some finite alphabet Σ containing an element 0. We use x_i to denote the i 'th coordinate of x and f_i to denote the function i 'th coordinate function of f . For $I \subseteq [n]$ we use x_I to denote the length- $|I|$ vector consisting of x restricted to its coordinates on indices in I , and Σ^I to denote the set of all such vectors.

In our model, gates apply a function f to some subset I of the W total wires in the circuit. It is useful to extend this function f to an $f_{\uparrow}$ that acts on all W wires of the circuit, but only modifies the wires in I according to f , leaving the other wires unchanged. Thus we define the following extension of a function f .

Definition 2.2 (lifting function $f_{\uparrow}$). *If $f : \Sigma^I \rightarrow \Sigma^I$ for some finite set $I = i_1, \dots, i_k \subseteq [W]$, then we can define the function $f_{\uparrow} : \Sigma^W \rightarrow \Sigma^W$ as:*

$$(f_{\uparrow})_i(x) = \begin{cases} f_i(x_I) & i \in I \\ x_i & i \notin I. \end{cases}$$

The following definition gives us clean notation for the distribution over states after applying a gate computing some function f .

Definition 2.3 (induced output distribution). *If $f : D \rightarrow R$ and $\mathcal{D}$ is a distribution over D , then we use $f(\mathcal{D})$ to denote the distribution over R produced by first sampling $x \sim \mathcal{D}$ and then outputting $f(x)$.*

Note that we have defined two distinct ways to apply functions to distributions, reflecting whether the co-domain is a scalar or an entire distribution: $\mathcal{L}_f(\mathcal{D}) \in \mathbb{R}$ is an example of the former, and Definition 2.3, a new distribution resulting from applying the function f to a sample, is an example of the latter. Our intended meaning should be clear based off the context. However, in general we will use capital letters such as $\mathcal{S}$ and $\mathcal{L}_f$ are functions that compute some property of the input distribution whereas lower case letters are used for mappings that define new distributions.

Definition 2.4. *The mutual information between two random variables X, Y is given by:*

$$\mathcal{I}(X; Y) = \mathcal{S}(X) + \mathcal{S}(Y) - \mathcal{S}(X, Y).$$

Definition 2.5 (Data processing inequality, [CT06]). *Let f be any function with domain D and $\mathcal{D}$ be any distribution over D . Then $\mathcal{S}(\mathcal{D}) \geq \mathcal{S}(f(\mathcal{D}))$.*

Definition 2.6. *The Kullback-Leibler divergence between two distributions $\mathcal{D}$ and $\mathcal{D}'$ is*

$$D_{\text{KL}}(\mathcal{D}||\mathcal{D}') = \mathbb{E}_{x \sim \mathcal{D}} \left[\log \frac{\Pr_{\mathcal{D}}[x]}{\Pr_{\mathcal{D}'}[x]} \right].$$

Proposition 2.7 (Data processing inequality for KL divergence, [CT06]). *Let $f : D \rightarrow R$ be an arbitrary function and $\mathcal{D}, \mathcal{D}'$ be two distributions over D . If $\text{supp}(\mathcal{D}) \subseteq \text{supp}(\mathcal{D}')$, then:*

$$D_{\text{KL}}(\mathcal{D}||\mathcal{D}') \geq D_{\text{KL}}(f(\mathcal{D})||f(\mathcal{D}')).$$

Definition 2.8. *If X and Y are random variables over D and R respectively, then the conditional entropy of X given Y is*

$$\mathcal{S}(X|Y) = \sum_{x \in D, y \in R} \Pr[X = x, Y = y] \log \left(\frac{\Pr[Y = y]}{\Pr[X = x, Y = y]} \right)$$

.

Proposition 2.9. *If X is a random variable and f is a deterministic mapping, then*

$$\mathcal{S}(X|f(X)) = \mathcal{S}(X) - \mathcal{S}(f(X))$$

.

Proposition 2.10 (Gibbs' Inequality). *Let $\mathcal{D}, \mathcal{D}'$ be two distributions over D . Then:*

$$D_{\text{KL}}(\mathcal{D}||\mathcal{D}') \geq 0$$

with equality only when $\mathcal{D} = \mathcal{D}'$.

Definition 2.11. *Let $f : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ be a function. We define the Gibbs distribution $\mathcal{G}_f$ over D with respect to f , where for any $x \in D$, (recall $2^{-\infty} = 0$)*

$$\Pr_{\mathcal{G}_f}[x] = \frac{2^{-f(x)}}{\sum_{x' \in D} 2^{-f(x')}}.$$

We prove the following lemma in Appendix A.

Lemma 2.12. *Let $f : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ be a function. For any set $S \subseteq D$ where $(\mathcal{G}_f)_{|S}$ is defined, it is the unique choice that minimizes $\mathbb{E}_{x \sim \mathcal{D}_{|S}}[f(x)] - \mathcal{S}(\mathcal{D}_{|S})$ over all distributions supported on S .*

We use $H_2(p) = p \log(1/p) + (1-p) \log(1/(1-p))$ to denote the *binary entropy* of a Bernoulli random variable with probability p . We use the following standard inequality on H_2 .

Proposition 2.13. $H_2(p) \leq p \log(e/p)$.

Proof.

$$\begin{aligned} H_2(p) &= p \log(1/p) + (1-p) \log(1/(1-p)) \\ &= p \log(1/p) + (1-p) \frac{\ln(1/(1-p))}{\ln 2} \\ &\leq p \log(1/p) + \frac{p}{\ln 2} \quad \text{since } \ln(x) \leq x - 1 \\ &= p \log(1/p) + p \log e \\ &= p \log(e/p). \end{aligned}$$

□

We also require the following standard inequality.

Proposition 2.14 (Hoeffding’s inequality [Hoe63]). *Let $\mathcal{U}$ be the uniform distribution on n bits and $t \leq n/2$. Then*

$$\Pr_{x_1, \dots, x_n \sim \mathcal{U}} \left[\sum_{i \in [n]} x_i \leq n/2 - t \right] = \Pr_{x_1, \dots, x_n \sim \mathcal{U}} \left[\sum_{i \in [n]} x_i \geq n/2 + t \right] \leq e^{-2t^2/n}.$$

2.1 Our model

While we will focus on circuits in this document, the ideas used in this section are sufficiently general to apply to any computational model that performs deterministic local transformations at fixed time steps.

Heat and entropy flow To simplify our math, we will assume that all computation is performed at a fixed temperature $\mathbb{T}$ and units are chosen such that $k_B \mathbb{T} \ln 2 = 1$. Throughout this paper we analyze how much entropy a computational device must release into its environment to obey the second law of thermodynamics. We will be assuming that each logical state of the machine has the same energy level and number of microstates.¹⁸ In principle this entropy can take many forms, but for our analysis it is safe to assume that it is always released as heat. This motivates what we will call *heat functions* as a way to abstract away messy details for how computations release excess entropy.

¹⁸We use the term “microstate” in this paper to mean the various different physical states a machine can be in while still representing the same logical state. As a concrete example, a Boolean gate implemented with semiconductors has a logical state identified by the bits on the input and output wires, which generally will correspond to many different physical microstates, depending on the precise positions and motion of electrons in the wires.

Thermodynamic circuits We start by giving two working definitions of circuits that we use in this paper. We start with our model of circuits, which is heavily inspired by ‘serial reinitialized’ model introduced in [WK20].

This model of circuits should be familiar to readers acquainted with the quantum circuit model. For those unfamiliar with such models, recall that a standard model defines a Boolean circuit as a directed acyclic graph, where each edge represents a wire connecting exactly two gates (nodes). Instead, in our model, all information is stored in a fixed set of n wires, and gates operate on a subset (typically of $O(1)$ -size) of these wires to update their values. However, unlike quantum circuits, we do not require gates to be unitary. Instead, they can perform any arbitrary mapping on their associated wires. While we use the language of circuits with wires and gates, this formulation works equally well to describe straight line programs where wires correspond to registers and gates correspond to operations on these registers. While we define such circuits in abstract, as is common in complexity theory, it will often be interesting to restrict them to specific gate sets (thus each gate would operate on $O(1)$ wires independently of the total number of wires). This rules out the possibility of implementing arbitrary functions with a single gate acting simultaneously on all wires of the circuit and lets us analyze the thermodynamic costs of computations with restricted gate sets.¹⁹

Definition 2.15. *A deterministic circuit $\mathcal{C}$ on n inputs with T gates and $W \geq n$ wires is described as an ordered collection of T gates $g_t = (I_{g_t}, f^{g_t})$ for $t \in \{1, \dots, T\}$. Each gate specifies a set $I_{g_t} \subseteq [W]$ and a function $f^{g_t} : \Sigma^{I_{g_t}} \rightarrow \Sigma^{I_{g_t}}$. On input $x_1, \dots, x_n \in \Sigma^n$ the circuit starts in the state $\mathcal{C}^0 \in \Sigma^W$ where $\mathcal{C}_i^0 = x_i$ for $i \in [n]$ and $\mathcal{C}_i^0 = 0$ for $i \in [W] \setminus [n]$. The state of the circuit at step t is given by $\mathcal{C}^t = f_{\uparrow}^{g_t}(\mathcal{C}^{t-1})$. We say that wire $i \in [W]$ of $\mathcal{C}$ evaluates to $x \in \Sigma$ when $\mathcal{C}_i^T = x$.*

Note that Definition 2.15 allows to have more wires (W) than input bits (n), to allow extra “workspace” for computation. It requires all non-input wires to be initially 0. The next definition captures our notion of computing output, which similarly allows the number of output bits m to be strictly less than the number of wires W , and similarly requires that at the end of the computation, all non-output wires are 0.²⁰

Definition 2.16. *We say that a circuit $\mathcal{C}$ cleanly computes a function $f : \Sigma^n \rightarrow \Sigma^m$ if on every input $x \in \Sigma^n$, wire i of $\mathcal{C}$ evaluates to $f_i(x)$ if $i \leq m$ and evaluates to 0 if $i > m$.²¹*

An example of a circuit that cleanly computes the and of its inputs is shown in Fig. 1.

Next we define how heat functions can augment a circuit to model its thermodynamic costs.

¹⁹ As with the standard DAG-based model of Boolean circuits, if one allows larger gates than $O(1)$, even up to a single gate taking all inputs and computing an arbitrary Boolean function in one step, then this will not give informative complexity measures. In this paper, restricting gates to operate on $O(1)$ wires will enable stronger lower bounds on entropy production, in an analogous sense that restricting standard Boolean circuits to have $O(1)$ -size gates enables nontrivial lower bounds on gate complexity.

²⁰ This requirement of zeroing out non-output wires is particularly important in the context of thermodynamics of computing, because it forces on the circuit the obligation to erase any non-output bits left over from the computation. Absent this requirement, a circuit could appear naïvely to produce less entropy than one would expect, hiding the cost of erasing those leftover “garbage” bits. However, if one wished to reuse the circuit for a subsequent computation, since those bits would at some point need to be set to 0 to prepare the circuit for the next input. The definition of clean computation forces the circuit to prepare itself for reuse. (Note that we do *not* account for the thermodynamic cost of writing the input, recording the output, or replacing the old output with the new input.)

²¹ The choice to make the *first* m wires correspond to output is arbitrary. If a circuit places its output on other wires, logically reversible swap gates can move the output to the first m wires with no extra entropy production.

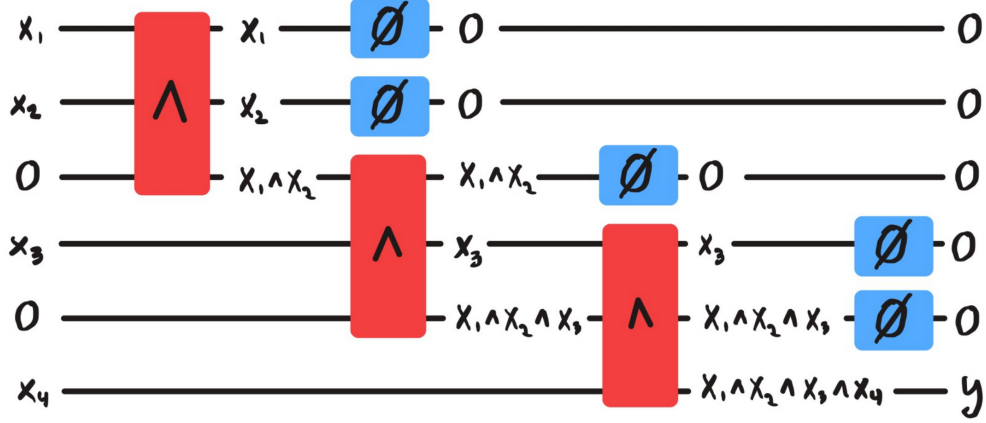


Figure 1: An example of a circuit that cleanly computes $y = x_1 \wedge x_2 \wedge x_3 \wedge x_4$ up to permutation of the wire indices. Here the red $\wedge$ gates perform the mapping $(a, b, c) \rightarrow (a, b, a \wedge b)$ and the blue $\emptyset$ erasure gates perform the mapping $a \rightarrow 0$. Each gate overwrites its inputs (values on wires to the left of the gate) with its outputs (values on wires to the right of the gate).

Definition 2.17. A thermodynamic circuit is a circuit $\mathcal{C}$ where each gate g computing a function with domain D is assigned a heat function $Q_g : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$. On input $x \in D$, the gate releases $Q_g(x)$ expected heat into the environment.

Remark 2.18. As we are fixing the temperature, it is reasonable to view Q as expected heat release. A more generally accurate description of $Q(x)$ would be the expected entropy flow out of the system on input x .

As we will see in Appendix D, it is important to note that Q is only allowed to depend on the inputs to a gate. This represents the fact that a gate is modular and oblivious to the larger state of the circuit. Otherwise, these heat functions can in principle be an arbitrary function of D . A gate that releases ∞ heat on some input x can be thought of being designed so it is not physically possible to specify x . This conveys the notion of restricted reversibility in [Fra17], where Frank points out that some logically irreversible operations can become reversible with such restrictions on their domain. However, not all choices of heat functions are valid. The Second Law of Thermodynamics (or Landauer’s costs) places constraints on the possible heat functions that can be assigned to each gate.

Definition 2.19 (Landauer cost, [Lan61]). For any function $f : D \rightarrow R$ and distribution $\mathcal{D}$ over D , we define the Landauer cost function of f on $\mathcal{D}$:

$$\mathcal{L}_f(\mathcal{D}) = \mathcal{S}(\mathcal{D}) - \mathcal{S}(f(\mathcal{D})).$$

The following is a corollary of the Second Law of Thermodynamics as applied to computation of functions. As a physical law, it cannot be proven in full generality, though it can be proven within specific models. We state it formally to motivate the subsequent Definition 2.21.

Proposition 2.20 (Second law, [Lan61, Sei05]). Let $f : D \rightarrow R$ be a function. Any physical process that maps every state $x \in D$ to the state $f(x)$ must release at least $\mathcal{L}_f(\mathcal{D})$ expected heat when x is sampled from distribution $\mathcal{D}$ over D .

In the context of physical systems evolving according to f , this is simply the lower bound on the expected heat release (or more generally entropy flow) out of a system where all logical configurations contain the same number of microstates as given by the second law of thermodynamics. The following definition essentially formalizes our notion of a heat function being compatible with the second law.

Definition 2.21. *Let g be a gate in circuit computing a function f with domain D . A heat function $Q : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ is valid for g if and only if, for every input distribution $\mathcal{D}$ over D , we have that:*

$$\mathcal{L}_f(\mathcal{D}) \leq \mathbb{E}_{x \sim \mathcal{D}}[Q(x)].$$

Definition 2.21 “directly” captures what we mean by a valid heat function—no matter the distribution it obeys the Second Law—and as stated intuitively appears to require validation over all input distributions over D . The following equivalent formulation of Definition 2.21 shows that validity in fact is solely a property of the value $Q(x)$ on *individual* inputs x . This is the deterministic specialization of Turgut’s Theorem 1 (Eq. (1)) in the simplified setting where each logical state has the same internal entropy [Tur09]. Zhu [Zhu24] subsequently rederived the same stochastic realizability criterion from a differential fluctuation theorem; with deterministic dynamics, Zhu’s condition likewise reduces to Lemma 2.22. We give a self-contained proof of Lemma 2.22 in Appendix A.

Lemma 2.22. *Let g be a gate in a circuit computing a function $f : D \rightarrow R$. A heat function $Q : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ is valid for g if and only if, for every $y \in R$,*

$$\sum_{x \in f^{-1}(y)} 2^{-Q(x)} \leq 1.$$

In practice, there is a large gap between valid heat functions (those obeying the Second Law) and physically realizable heat functions (which may release far *more* heat than the minimum dictated by the Second Law, e.g., current CMOS technology). Additional heat is released by computation to drive the physical processes that implement the logical mapping of the gate in the desired direction in finite time [SFS18]. However, valid heat functions let us prove lower bounds on the thermodynamic costs of a computation and provide a limit for the physical implementation of the computation. In this paper, we state upper bounds on thermodynamic costs that make the idealized assumption that any combination of logical gate g and valid heat function Q for g could actually be constructed. The known laws of physics neither explicitly construct such gates nor rule out their existence (in the quasistatic limit). Thus our upper bounds all implicitly require the following “spherical cow” like assumption.

Assumption 2.23 (Idealized realizability assumption). *Let $f : D \rightarrow R$ be any function and $Q : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ be any valid heat function for computing f . Then we assume that it is possible to fabricate a gate that computes f while releasing $Q(x)$ expected heat on each input where $Q(x) \neq \infty$ in the quasistatic (i.e. infinitely slowly) limit. We assume that whenever $Q(x) = \infty$ the gate is implemented in a way where it physically cannot receive such an input. In other words, the gate actually computes a restriction of f on inputs where $Q(x)$ is finite.²²*

²²The fact that $Q(x) = \infty$ already makes this domain restriction implicit in our model. Any distribution with support on such an input would immediately make Q infinite. Thus heat functions where $Q(x) = \infty$ on any input occurring with positive probability trivially satisfy lower bounds and would automatically make any resulting upper bound trivial.

This is well motivated. Theorem 2 of Turgut [Tur09] shows how, in an idealized model, it is possible to implement any fixed mapping from D to R where the expected released heat on every input exactly matches any valid Q . If one wishes to reject this assumption, then they should interpret all of our upper bounds on Q and EP as theoretical limits on the best possible lower bounds for these quantities that could be proven in our model. All of our lower bounds on EP and Q hold independently of this axiom.

Excess heat released by a physical process is called the entropy production [Sei12]. More formally (recall Definition 2.2 of $f_{\uparrow}$),

Definition 2.24 (gate entropy production). *Let $\mathcal{C}$ be a W -wire circuit and g be a gate in $\mathcal{C}$ computing a function f with domain $D = \Sigma^I$ for $I \subseteq [W]$ that is assigned heat function Q . The entropy production EP_g of g on distribution²³ $\mathcal{D}$ over Σ^W is given by:*

$$EP_g(\mathcal{D}) = \mathcal{S}(f_{\uparrow}(\mathcal{D})) - \mathcal{S}(\mathcal{D}) + \mathbb{E}_{x \sim \mathcal{D}}[Q(x_I)]$$

Note $\mathcal{S}(f_{\uparrow}(\mathcal{D}))$ is used in Definition 2.24 instead of $\mathcal{S}(f(\mathcal{D}))$ since $\mathcal{D}$ is a distribution on all W wires. Note that $\mathcal{S}(f_{\uparrow}(\mathcal{D})) - \mathcal{S}(\mathcal{D})$ is the *negation* of the Landauer cost of f on $\mathcal{D}$ (Definition 2.19), thus is nonpositive by Definition 2.5. $\mathbb{E}_{x \sim \mathcal{D}}[Q(x_I)]$ is the *entropy flow/heat released* (see Section 1 for discussion of this term in the broader field of stochastic thermodynamics) in the context of circuits that exchange heat with the environment through their gates.

It is necessary to define EP_g in the context of the larger circuit, since $\mathcal{S}(f_{\uparrow}(\mathcal{D})) - \mathcal{S}(\mathcal{D})$ is a global property of the circuit, while $\mathbb{E}_{x \sim \mathcal{D}}[Q(x_I)]$ is a property of the marginal distribution of $\mathcal{D}$ on the coordinates in I . We can also define a local equivalent, termed EP_g^{Loc} , to EP_g , which gives a lower bound on the actual entropy production EP_g . In Appendix D we show how these two quantities are related (Proposition D.1).

Definition 2.25 (local gate entropy production). *Let g be a gate computing a function f with domain D that is assigned a heat function Q . The local entropy production EP_g^{Loc} of g on input distribution $\mathcal{D}$ over D is given by:*

$$EP_g^{\text{Loc}}(\mathcal{D}) = \mathcal{S}(f(\mathcal{D})) - \mathcal{S}(\mathcal{D}) + \mathbb{E}_{x \sim \mathcal{D}}[Q(x)].$$

We are ultimately interested in characterizing the entropy production of a circuit. When a circuit receives some distribution $\mathcal{D}$ over its inputs, this could in principal be a marginal distribution where the circuit's inputs are only a subset of the variables in a larger system, with which the inputs may have mutual information that would call for different heat functions for “globally optimal” heat release for each gate. We give the following definition for the entropy production of a circuit under the assumption that its input distribution $\mathcal{D}$ is independent from the outside world.

Definition 2.26 (circuit entropy production). *Let $\mathcal{C}$ be a thermodynamic circuit with gates $g \in G$ that cleanly computes a function f with domain D . Let $\mathcal{D}$ be a distribution over D and $\mathcal{D}_g$ be the induced marginal distribution over inputs for gate g which has heat function Q_g . The circuit entropy production of $\mathcal{C}$ on distribution $\mathcal{D}$ is given by*

$$EP_{\mathcal{C}}(\mathcal{D}) = \mathcal{S}(f(\mathcal{D})) - \mathcal{S}(\mathcal{D}) + \sum_{g \in G} \mathbb{E}_{x \sim \mathcal{D}_g}[Q_g(x)]$$

²³Note $\mathcal{D}$ is not the input distribution to the whole circuit, but the induced distribution on all W wires *just prior* to applying gate g .

$\text{EP}_{\mathcal{C}}(\mathcal{D})$ as defined above is a lower bound on the actual entropy production involved with running the circuit, as we show in Appendix D (Proposition D.1). This bound is tight when $\mathcal{D}$ is an independent source of randomness uncorrelated with the rest of the universe. We call this circuit entropy production because it is exactly the entropy production of the circuit as if it were isolated from its environment. We note that this is the same notion of entropy production for circuits used in prior works (i.e., [WK20, YCW26, YYW26]).

Remark 2.27. *We have defined 3 separate types of entropy production in this section. Informally, EP_g^{Loc} is the entropy production due to using a sub-optimal heat function for gate g given its induced marginal distribution over inputs. This captures what the entropy production of g would be if its input distribution $\mathcal{D}$ were sampled independently of the rest of the universe. EP_g , which is always at least EP_g^{Loc} , captures EP_g^{Loc} plus additional costs due to $\mathcal{D}$ having correlations with other wires in the circuit. Finally $\text{EP}_{\mathcal{C}}$ is the total entropy production across all gates in the circuit under the assumption that $\mathcal{D}$ is sampled independently from the rest of the universe. We will be proving lower and upper bounds on $\text{EP}_{\mathcal{C}}$ throughout the body this paper and we use both EP_g and EP_g^{Loc} primarily in Appendices C and D where we discuss mismatch costs.*

Overwriting circuits. It is worth noting that in [YCW26, YYW26] the authors use a different notion of a thermodynamic circuit than we consider here. In their *overwriting* model, a circuit is defined (as is common in most non-quantum circuit complexity classes) as a DAG where the nodes are labeled with gates and the in-edges define the inputs for each gate. The authors are interested in repeated runs of their circuit where the old logical configuration of each gate is overwritten one gate at a time during the next run, so the thermodynamic costs are incurred incrementally as each gate is updated and replaces its old output with the new one. The overwriting model is an accurate abstraction of the CMOS architecture; however, it places some restrictions on what is feasible with circuits from a theoretical perspective. For example a logically reversible gate in the our circuit model admits the valid heat function $Q(x) = 0$ on all inputs. In the overwriting model, this gate must release heat when run on independently distributed inputs unless it is computing a constant function.²⁴ In fact, we can see that any circuit in the overwriting model can be simulated in our circuit model, simulating a k -input, 1-output gate by using k ‘input’ wires and one ‘output’ wire in each of our gates, setting the other $k - 1$ wires to 0. Each gate in the simulated overwriting circuit is now a gate in our circuit model that treats the output wires of its parents (or the input) as read-only inputs and only modifies its own output wire. At the end of the computation, the values stored in all output wires for intermediate gates are erased.²⁵

Cycles of computation and changing inputs. An important subtlety with the thermodynamics of computation is how to think about the state of the device before and after a computation has been completed. Logical reversibility gives a black box way to, for any efficiently computable function f , efficiently perform the mapping $x \rightarrow (x, f(x))$ without releasing any heat. However, even if f is injective (hence “globally” logically reversible), this does not imply a black box way to compute the mapping $x \mapsto f(x)$ unless f^{-1} is also efficiently computable.²⁶ More generally any circuit that

²⁴In the overwriting model, the validity of the heat function for a deterministic gate is entirely determined by the drop in entropy caused by erasing the old value.

²⁵As long as an overwriting circuit is run on independently distributed inputs, it is safe to assume without loss of generality that it erases the values stored in every gate before computing the new output value.

²⁶For instance, it is well-known folklore that one-way functions cannot be efficiently computed in the forward direction (outputting only $f(x)$ cleanly as in Definition 2.16, without retaining a copy of x anywhere) using logically

produces garbage in addition to its output must erase that garbage before it can be run on a new input. Failing to do so causes real issues; if one extends Definition 2.21 to probabilistic gates, then there are valid heat functions where $Q(x)$ may be negative. By using such gates, you can define circuits that appear to violate the second law of thermodynamics until you account for the increase in entropy in their mapping of inputs to outputs.

Much of the complexity associated with reuse/cycling can be simplified by thinking of the circuit as a cog in a larger machine that goes through cycles. Let's consider a circuit computing a function $f : \Sigma^n \rightarrow \Sigma^m$ with W total wires. At the start of a cycle, all W wires are initialized to zero. Then some upstream process writes a new input x into the first n wires of our circuit. The circuit will perform its computation in accordance with Definition 2.15, ending in a state where the first m wires contain $f(x)$ and all other wires are reset to zero. The value $f(x)$ can then be swapped with m fresh zeroes from the downstream process that takes $f(x)$ as its input at no cost. Resetting these values is a task for the downstream process. Note that f can be defined to keep a copy of its input. Thus the standard logically reversible mapping $f(x) = (x, h(x))$ is allowed in this model, but then x is explicitly part of the output of f and must be handled as such by downstream processes.

3 Deriving optimal heat functions

Now we show how to derive the optimal heat function for a known distribution over inputs. Recall that for a logically reversible operation, i.e., computing an injective function $f : D \rightarrow R$, an optimal heat function is 0 on all inputs. In this case, no matter the input distribution $\mathcal{D}$, since each input x maps to its own unique output $f(x)$, $\Pr_{f(\mathcal{D})}[f(x)] = \Pr_{\mathcal{D}}[x]$, thus $\log(\Pr_{f(\mathcal{D})}[f(x)]/\Pr_{\mathcal{D}}[x]) = 0$. If f is not injective, it remains the case that any input x mapping to its own unique output $f(x)$ has zero expected heat released on input x . “Colliding” inputs will release more heat the more inputs map to a single output (more precisely, the greater the total probability $\Pr_{f(\mathcal{D})}[f(x)]$ of inputs x sharing output $f(x)$): still $\log(\Pr_{f(\mathcal{D})}[f(x)]/\Pr_{\mathcal{D}}[x])$ (now strictly positive) is the expected heat released.

Lemma 3.1. *Let f be an arbitrary function with domain D and $\mathcal{D}$ be a distribution over D . Then any valid heat function for a gate computing f that minimizes the expected heat release on input distribution $\mathcal{D}$ satisfies for all $x \in D$:*

$$Q(x) = \begin{cases} \log(\Pr_{f(\mathcal{D})}[f(x)]/\Pr_{\mathcal{D}}[x]) & \Pr_{\mathcal{D}}[x] > 0 \\ \infty & \Pr_{\mathcal{D}}[x] = 0, \Pr_{f(\mathcal{D})}[f(x)] > 0 \end{cases}$$

on all x where $f(x) \in \text{supp}(f(\mathcal{D}))$.²⁷ On x where $f(x) \notin \text{supp}(f(\mathcal{D}))$, Q can behave arbitrarily so long as Q still satisfies Lemma 2.22 on these inputs.

Note that in the finite case, $Q(x)$ is the log of the one divided by the probability that the input is x , conditional on the output being $f(x)$. In other words, the optimal heat function is unique up to the choice of values for inputs with zero probability under the input distribution $\mathcal{D}$ and whose outputs also have zero probability under the output distribution $f(\mathcal{D})$. While an earlier version of

reversible computation; Bennett's construction [Ben73, Table 2] for efficiently and reversibly computing the mapping $x \mapsto f(x)$ for bijective f requires both f and f^{-1} to be efficiently computable, ruling out one-way functions.

²⁷Note that equivalently $\log(\Pr_{f(\mathcal{D})}[f(x)]/\Pr_{\mathcal{D}}[x]) = -\log \Pr_{\mathcal{D}}[x | \text{output is } f(x)]$.

this proof appeared in [Kor25], it contained some bugs. Here we give a corrected proof of optimality and uniqueness up to the behavior of Q on such inputs.

Proof. Picking Q as in the statement of Lemma 3.1 exactly satisfies the best possible bound of:

$$\mathcal{L}_f(\mathcal{D}) = \mathbb{E}_{x \sim \mathcal{D}}[Q(x)]. \quad (2)$$

To see that this choice is valid, we show that Q satisfies Lemma 2.22. For every $y \in \text{supp}(f(\mathcal{D}))$ we have:

$$\sum_{x \in f^{-1}(y)} 2^{-Q(x)} = \sum_{x \in f^{-1}(y)} \frac{\Pr_{\mathcal{D}}[x]}{\Pr_{f(\mathcal{D})}[y]} = 1.$$

And for every $y \notin \text{supp}(f(\mathcal{D}))$ the only constraint we place on Q is that this quantity is at most 1.

Next we show that any heat function that minimizes the heat release on $\mathcal{D}$ must satisfy Lemma 3.1. Let Q^* be some other valid heat function that minimizes the expected heat release on input distribution $\mathcal{D}$. Then per Lemma 2.22 for every y , $\sum_{x \in f^{-1}(y)} 2^{-Q^*(x)} \leq 1$. Moreover if $\sum_{x \in f^{-1}(y)} 2^{-Q^*(x)} < 1$ for some $y \in \text{supp}(f(\mathcal{D}))$ then there must be some $x \in \text{supp}(\mathcal{D}_{|f^{-1}(y)})$ where $Q^*(x)$ could have been smaller while remaining valid, and therefore Q^* could not have been optimal. Thus we can assume that for every $y \in \text{supp}(f(\mathcal{D}))$:

$$\sum_{x \in f^{-1}(y)} 2^{-Q^*(x)} = 1.$$

This means that we can define the conditional Gibbs distribution $\Pr_{(\mathcal{G}_{Q^*})_{|f^{-1}(y)}}[x] = 2^{-Q^*(x)}$ for every $y \in \text{supp}(f(\mathcal{D}))$. Since Q^* is valid we know that for every $y \in \text{supp}(f(\mathcal{D}))$:

$$\mathbb{E}_{x \sim (\mathcal{G}_{Q^*})_{|f^{-1}(y)}}[Q^*(x)] - \mathcal{S}((\mathcal{G}_{Q^*})_{|f^{-1}(y)}) \geq 0.$$

And since Q^* is a valid heat function that minimizes the heat release on the distribution $\mathcal{D}$ we know that for every $y \in \text{supp}(f(\mathcal{D}))$,

$$\mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}}[Q^*(x)] - \mathcal{S}(\mathcal{D}_{|f^{-1}(y)}) = 0,$$

as Definition 2.21 requires that this expression be non-negative for every $y \in \text{supp}(f(\mathcal{D}))$ and we need the average of these terms to be 0 to match Eq. (2). But by Lemma 2.12, we know that this implies that for every y where $\mathcal{D}_{|f^{-1}(y)}$ is defined, $\mathcal{D}_{|f^{-1}(y)} = (\mathcal{G}_{Q^*})_{|f^{-1}(y)}$ and hence for every x where $f(x) \in \text{supp}(f(\mathcal{D}))$,

$$2^{-Q^*(x)} = \frac{\Pr_{\mathcal{D}}[x]}{\Pr_{f(\mathcal{D})}[f(x)]} = 2^{-Q(x)}.$$

For x where $\Pr_{f(\mathcal{D})}[f(x)] = 0$ then both $Q(x)$ and $Q^*(x)$ can take on arbitrary non-negative values without affecting the optimality of the heat function, so long as it remains valid. $\square$

4 The complexity of optimizing heat functions in a given circuit

Here we show that computing optimal heat functions for gates in a fixed circuit is PP-hard.²⁸ In light of Lemma 3.1, this hardness result is hardly surprising: computing an optimal heat function

²⁸See [AB09, Chapter 9] for a discussion of the complexity class PP. Briefly, it is the class of decision problems/languages $\mathcal{L}$ for which yes-instances x , that like NP problems, have a polynomial-length $k = \text{poly}(|x|)$ witness $w \in \{0, 1\}^k$, furthermore have at least *half* of all $w \in \{0, 1\}^k$ as witnesses; see for example the language $\mathcal{L}_{\text{SAT}/2}$ below. A probabilistic characterization (PP actually stands for “probabilistic polynomial-time”) is that there is a randomized polynomial-time algorithm for $\mathcal{L}$ that answers ‘yes’ on input x with probability at least $1/2$ if and only if $x \in \mathcal{L}$.

requires computing probabilities induced by the circuit. Nonetheless, we view this as an overall strength of the heat function framework.

4.1 PP-complete language for optimizing heat functions

We start with the language $\mathcal{L}_{Q\text{-min}}$ that we will prove PP-complete.

Definition 4.1. $\mathcal{L}_{Q\text{-min}}$ is a set of tuples $(\mathcal{C}, g, x, a, b)$ where $\mathcal{C}$ is a Boolean circuit, g is a gate in that circuit, x is a potential input to that gate, and $q = a/b$ is a positive rational number. We say that $(\mathcal{C}, g, x, a, b) \in \mathcal{L}_{Q\text{-min}}$ if, when $\mathcal{C}$ receives the uniform distribution over inputs, there exists some valid heat function Q for gate g that minimizes the expected heat release on its induced marginal input distribution such that $Q(x) \geq \log q$.

We note that our choice to express threshold $q = a/b$ as a rational number whose log corresponds to the optimal expected released heat is for simplicity but is not actually needed. Per Lemma 3.1 the optimal value of q is always the ratio of two integer valued fractions whose numerators are at most 2^n and whose denominators are both exactly 2^n . Thus we can safely round up any q obtain a rational q' that is the ratio of two $\leq n + 1$ bit integers a' and b' without changing membership in $\mathcal{L}_{Q\text{-min}}$.

To show that $\mathcal{L}_{Q\text{-min}}$ is PP-hard, we will reduce from the following PP-hard problem [AW22].²⁹

Proposition 4.2 ([AW22]). Let $\mathcal{L}_{\text{SAT}/2}$ be the subset of SAT instances that evaluate to 1 on at least half of all inputs. Deciding membership in $\mathcal{L}_{\text{SAT}/2}$ is PP-hard.

Lemma 4.3. $\mathcal{L}_{Q\text{-min}}$ is PP-hard.

Proof. We prove this by Karp reduction from $\mathcal{L}_{\text{SAT}/2}$. Given a Boolean circuit $\mathcal{C}$, assume its output is the first wire. We modify $\mathcal{C}$ by appending a single-wire gate g that erases its input to the first wire at the end of the circuit and call our new circuit $\mathcal{C}'$. Observe that, due to Lemma 3.1, the optimal heat function for the added gate g releases at least 1 expected heat on input 0 precisely when the probability of that input is at most $1/2$. Thus, $(\mathcal{C}', g, 0, 2, 1) \in \mathcal{L}_{Q\text{-min}}$ precisely when $\mathcal{C} \in \mathcal{L}_{\text{SAT}/2}$. $\square$

Next we show that this language is in PP.

Lemma 4.4. $\mathcal{L}_{Q\text{-min}} \in \text{PP}$.

Proof. Consider input $(\mathcal{C}, g, x, a, b)$. Let $\mathcal{D}$ be the marginal distribution over inputs to gate g when $\mathcal{C}$ receives the uniform distribution over inputs, f be the function computed by gate g , $y = f(x)$, and $q = a/b$. The value $q \leq 1$ asks if the optimal heat function releases at least 0 expected heat on input x . Since such a condition is trivially satisfied by the definition of heat functions, we can automatically accept such inputs. Now we assume that $a \geq b$. By Lemma 3.1, we know that deciding if $(\mathcal{C}, g, x, a, b) \in \mathcal{L}_{Q\text{-min}}$ is equivalent to deciding if $q = a/b \leq \Pr_{f(\mathcal{D})}[y] / \Pr_{\mathcal{D}}[x]$. We will now construct a non deterministic program that accepts with probability at least $1/2$ exactly when

²⁹A version of this problem where ‘at least’ is replaced with ‘more than’ was proven to be PP-hard in [Gil74]. That paper also uses the convention that PP algorithms must accept with probability strictly larger than $1/2$ on yes instances. In [AW22] the authors show how to convert between these definitions of PP.

$q \leq \Pr_{f(\mathcal{D})}[y] / \Pr_{\mathcal{D}}[x]$. We start by observing that this condition is equivalent to (multiplying both sides by $b \Pr_{\mathcal{D}}[x]$, then subtracting $b \Pr_{\mathcal{D}}[x]$ from both sides)

$$(a - b) \Pr_{\mathcal{D}}[x] \leq b \left(\Pr_{f(\mathcal{D})}[y] - \Pr_{\mathcal{D}}[x] \right).$$

Thus by designing a non deterministic program whose success probability is at least $1/2$ when the above inequality is satisfied, we have a PP algorithm for $\mathcal{L}_{Q\text{-min}}$. Importantly, note that $\Pr_{f(\mathcal{D})}[y] - \Pr_{\mathcal{D}}[x]$ is exactly the probability that $\mathcal{D}$ selects some input $x' \neq x$ where $f(x') = y$. We start by picking an input for $\mathcal{C}$ uniformly at random and evaluating it on $\mathcal{C}$. Let x' be the observed input to gate g and $y' = f(x')$ be the observed output. We then sample a Bernoulli random variable v that has outcome 1 with probability b/a .³⁰ If we have $v = 1, x' \neq x$, and $y' = y$ then we accept. If we have $v = 0$ and $x' = x$ then we reject. On all other patterns we accept or reject with probability $1/2$. We note that this protocol accepts with probability exactly $1/2$ precisely when $q = \Pr_{f(\mathcal{D})}[y] / \Pr_{\mathcal{D}}[x]$ or $\Pr_{f(\mathcal{D})}[y] = 0$. In both cases we have that our input is in $\mathcal{L}_{Q\text{-min}}$ and thus should accept; when $q = \Pr_{f(\mathcal{D})}[y] / \Pr_{\mathcal{D}}[x]$ we have that q is precisely the optimal value and when $\Pr_{f(\mathcal{D})}[y] = 0$ every choice of q can be optimal. Given our definition of PP, this is the intended behavior. $\square$

4.2 Approximation

We first make a simple argument that if circuits are permitted to have wires where only a 0 bit can ever appear (i.e., such as the output of an unsatisfiable formula), then it is NP-hard even to approximate an optimal heat function. Fix a circuit $\mathcal{C}$ and let g be an erasure gate acting on the first wire at the end of that circuit. Let $\mathcal{D}$ be the marginal induced input distribution for g when $\mathcal{C}$ receives a uniform distribution over its inputs and Q^* be a valid heat function that minimizes the expected heat release on distribution $\mathcal{D}$. If $\Pr_{\mathcal{D}}[0] = 1$ then we can safely set $Q(0) = 0, Q(1) = \infty$ and observe that $\mathbb{E}_{x \sim \mathcal{D}}[Q(x)] = 0$. However, if there is any input to $\mathcal{C}$ that causes the first wire to evaluate to 1 before applying g , then this choice of Q will cause $\mathbb{E}_{x \sim \mathcal{D}}[Q(x)] = \infty$. Thus, for any $c \geq 1$, this choice of Q is the unique choice that satisfies $\mathbb{E}_{x \sim \mathcal{D}}[Q(x)] \leq c \mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)]$ when $\Pr_{\mathcal{D}}[0] = 1$ and Q can never satisfy this condition when $\Pr_{\mathcal{D}}[1] > 0$. Therefore by fixing $\mathcal{C}$ to be the $\mathcal{C}'$ in Lemma 4.3, finding any multiplicative approximation to the optimal heat function for the added erasure gate is at least as hard as solving SAT.

However, this is a somewhat degenerate case, in the sense that *any* optimization problem permitting an optimal value of 0 has the property that no multiplicative approximation is good unless it is exact. The more interesting case is when a wire might have both bits 0 or 1 ($\text{supp}(\mathcal{D}) = \{0, 1\}$), so that we have a positive optimal expected heat $\mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)] > 0$. Using a bit of machinery, we can prove that NP-hardness of multiplicative approximation holds even in this case.

Remark 4.5 (Pointwise versus expected approximation). *The motivation for studying the thermodynamics of a circuit is to reduce the amount of work (i.e. useful energy) that must be irreversibly spent each time circuit is run. Ultimately we care about finding heat functions that minimize the*

³⁰While this is not possible in general under standard definitions of PP, we can get the desired behavior by flipping $\lceil \log(a + b) \rceil + 1$ coins. Treat all but the last of these coins as a number z in binary. If $z < b$ we set $v = 1$. If $b \leq z < a$ we set $v = 0$. If $a \leq z$ then we use the final coin to immediately accept or reject without changing the correctness of our algorithm. Post-conditioned on not immediately halting, this protocol simulates the desired Bernoulli random variable.

circuit entropy production, or equivalently (since the mapping of inputs to outputs is fixed) the net expected released heat by each gate. Thus, for our approximation results, the natural computational objective is to produce any heat function where the expected released heat is within an additive (or multiplicative) factor of optimal, rather than to approximate the individual entries of the optimal heat function.

We first require a technical lemma about converting a formula ϕ to another ψ such that, depending on whether ϕ is satisfiable, ψ has either 1 or a much larger number of satisfying assignments. This follows by adding t dummy variables to a construction by Blass and Gurevich [BG82] that converts any SAT formula ϕ on n variables into another formula ψ on $n + 1$ variables which contains exactly one additional satisfying assignment.

Lemma 4.6. *Let $\phi(x_1, \dots, x_n)$ be a SAT instance on n variables with $Z(\phi)$ satisfying assignments. For every $t \geq 0$, there exists another SAT instance $\psi(x_1, \dots, x_n, y_1, \dots, y_t, z)$ on $n + t + 1$ variables that has $Z(\psi) = 1 + 2^t Z(\phi)$ satisfying assignments.*

Proof. We fix $\psi(x_1, \dots, x_n, y_1, \dots, y_t, z) = \left(\left(\bigwedge_{i \in [n]} x_i \right) \wedge \left(\bigwedge_{i \in [t]} y_i \right) \wedge z \right) \vee (\phi(x_1, \dots, x_n) \wedge \neg z)$. When $z = 1$, ψ is satisfied exactly when all x_i and y_i are assigned 1. When $z = 0$, each assignment of $x_1, \dots, x_n$ that satisfy ϕ now results in 2^t assignments of $y_1, \dots, y_t$ that all satisfy ψ . Thus the total number of satisfying assignments is now $1 + 2^t Z(\phi)$. $\square$

Next, we prove NP-hardness of multiplicative approximation even for gates whose inputs have full support.

Lemma 4.7. *Fix any $c > 1$. Let $\mathcal{C}$ be a circuit receiving the uniform distribution over inputs and g be a gate in $\mathcal{C}$. Let $\mathcal{D}$ be the induced marginal distribution over inputs to g . Let Q^* be a valid heat function for g that minimizes $\mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)]$. Finding a valid heat function Q for g such that $\mathbb{E}_{x \sim \mathcal{D}}[Q(x)] \leq c \mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)]$ that is specified by a pairs a_x, b_x of $n + 1$ bit integers such that $Q(x) = \log(a_x/b_x)$ is impossible in randomized polynomial time even with the promise that $\mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)] > 0$ unless $\text{NP} \subseteq \text{BPP}$.*

Proof. We prove this via an oracle reduction from SAT. Let ϕ be a SAT instance on n variables and ψ be the resulting instance on $n + t + 1$ variables after applying Lemma 4.6 for $t = \alpha cn$ with constant $\alpha > 1$ that we will choose later. By construction, if ϕ had $Z(\phi)$ satisfying assignments, then ψ must have $Z(\psi) = 1 + 2^t Z(\phi)$ satisfying assignments. This implies that $1 \leq Z(\psi) \leq 1 + 2^{n+t} < 2^{n+t+1}$. We convert ψ into a Boolean circuit $\mathcal{C}$ whose output is the first wire and then append an erasure gate to that wire to obtain circuit $\mathcal{C}'$. Let $Q^*(x)$ be some heat function that minimizes the expected heat release on the induced input distribution $\mathcal{D}$ to the added erasure gate. Since ψ evaluates to both true and false on some input, we know that $\mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)] > 0$. If ϕ was unsatisfiable, then ψ has exactly 1 satisfying assignment and $\Pr_{\mathcal{D}}[1] = 1/2^{n+t+1}$. If ϕ had some satisfying assignment then $\Pr_{\mathcal{D}}[1] \geq (1 + 2^t)/2^{n+t+1}$. Let Q be a valid heat function such that $\mathbb{E}_{x \sim \mathcal{D}}[Q(x)] \leq c \mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)]$. By Lemma 3.1 we know that $Q^*(x) = -\log(\Pr_{\mathcal{D}}[x])$ and thus if ϕ was unsatisfiable we have:

$$(1 - 1/2^{n+t+1})Q(0) \leq \mathbb{E}_{x \sim \mathcal{D}}[Q(x)] \leq c H_2(1/2^{n+t+1}) \leq c(n + t + 1 + \log e)/2^{n+t+1},$$

which implies $Q(0) \leq \frac{c(n+t+1+\log e)}{2^{n+t+1}-1} \leq \frac{c(n+t+1+\log e)}{2^{n+t}}$. Since Q must be valid Lemma 2.22 gives us:

$$Q(1) \geq -\log(1 - 2^{-Q(0)})$$

$$\begin{aligned}
&\geq -\log(Q(0) \ln 2) \\
&\geq \log \left(\frac{2^{n+t}}{c(n+t+1+\log e) \ln 2} \right) \\
&= n+t - \log(c(n+t+1+\log e) \ln 2) \\
&= \alpha cn + n - \log(c((\alpha c + 1)n + 1 + \log e) \ln 2) \\
&> \alpha cn - 2 \log(5\alpha cn) \quad \text{since } \alpha, c, n \geq 1
\end{aligned} \tag{3}$$

But if ϕ were satisfiable then we must have that:

$$\Pr_{\mathcal{D}}[1]Q(1) \leq \mathbb{E}_{x \sim \mathcal{D}}[Q(x)] \leq cH_2(\Pr_{\mathcal{D}}[1]) \leq c\Pr_{\mathcal{D}}[1] \log(e/\Pr_{\mathcal{D}}[1])$$

Which implies:

$$\begin{aligned}
Q(1) &\leq c \log(e/\Pr_{\mathcal{D}}[1]) \\
&= c(\log(e) - \log(\Pr_{\mathcal{D}}[1])) \\
&\leq c \left(\log(e) - \log \left(\frac{1+2^t}{2^{n+t+1}} \right) \right) \\
&= c(n+t+1+\log e - \log(1+2^t)) \\
&\leq c(n+1+\log e) \\
&< 4cn.
\end{aligned}$$

As long as $\alpha cn \geq 18$, Eq. (3) is at least $\alpha cn/4$. For simplicity, we will fix $\alpha = 18$ to get that $Q(1) > 9cn/2$ when ϕ is unsatisfiable. But, if ϕ has a single satisfying assignment then $Q(1) < 4cn$. Thus finding a valid heat function Q for this erasure gate such that $\mathbb{E}_{x \sim \mathcal{D}}[Q(x)] \leq c\mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)]$ tells us whether ϕ has a satisfying assignment. $\square$

The NP-hardness of multiplicative approximation is unfortunate, as being able to estimate the optimal heat functions for each of T gates in a circuit to multiplicative error c immediately gives us a thermodynamic circuit that is within a multiplicative factor c of optimal. Like all NP-hard problems, it also rules out a polynomial-time *randomized* algorithm unless $\text{NP} \subseteq \text{BPP}$.

There *is* a polynomial-time randomized algorithm for the problem of finding optimal heat functions with *additive* error on the expected released heat. Finding heat functions for each of the T gates with additive error at most ε will give us a thermodynamic circuit where the additive error (i.e. expected excess heat dissipation) across all gates is at most εT . By setting $\varepsilon = \varepsilon'/T$ we can obtain an additive error ε' for the full circuit that doesn't scale with T . The basic idea is simple: randomly sample inputs to the circuit and use them to empirically estimate the true input distribution to the gate with add-one (or Laplace) smoothing.³¹

Lemma 4.8. *Fix any $\varepsilon > 0$. Given a circuit $\mathcal{C}$ where all gates act on a constant number of bits and one such gate g in $\mathcal{C}$, let $\mathcal{D}$ be the induced marginal distribution over the inputs to gate g when $\mathcal{C}$ receives the uniform distribution over its inputs. Let Q^* be a valid heat function for g that minimizes $\mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)]$. There is a $\text{poly}(n, 1/\varepsilon, 1/\delta)$ time randomized algorithm that produces a valid heat function $\hat{Q}$ such that $\mathbb{E}_{x \sim \mathcal{D}}[\hat{Q}(x)] \leq \mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)] + \varepsilon$ with probability at least $1 - \delta$.*

³¹The smoothing factor gives us a straightforward way to ensure that our produced heat function $\hat{Q}$ never sets $\hat{Q}(x) = \infty$ on any input.

Proof. Let f be the function computed by gate g , D be the domain of f , and $d = |D|$. We pick t inputs to $\mathcal{C}$ uniformly at random, evaluate $\mathcal{C}$ on these inputs, and count the number of times we see each input to g . Let $C : D \rightarrow \mathbb{Z}_{\geq 0}$ be the obtained counts for each input $x \in D$ to g . From these counts, we define some empirical distribution $\widehat{\mathcal{D}}$ and let $\widehat{Q}(x) = \log(\Pr_{f(\widehat{\mathcal{D}})}[f(x)] / \Pr_{\widehat{\mathcal{D}}}[x])$ be the optimal heat function (per Lemma 3.1) if $\widehat{\mathcal{D}}$ were the actual input distribution to g . We use $\widehat{Q}_C, \widehat{\mathcal{D}}_C$ to denote the choices of $\widehat{\mathcal{D}}$ and $\widehat{Q}$ respectively resulting from observing counts C . We use $\mathcal{C}$ to denote the distribution over the observed counts. This gives us that:

$$\begin{aligned}
& \mathbb{E}_{C \sim \mathcal{C}} [\mathbb{E}_{x \sim \mathcal{D}} [\widehat{Q}_C(x) - Q^*(x)]] \\
&= \mathbb{E}_{C \sim \mathcal{C}} \left[\mathbb{E}_{x \sim \mathcal{D}} \left[\log \left(\frac{\Pr_{f(\widehat{\mathcal{D}}_C)}[f(x)]}{\Pr_{\widehat{\mathcal{D}}_C}[x]} \right) - \log \left(\frac{\Pr_{f(\mathcal{D})}[f(x)]}{\Pr_{\mathcal{D}}[x]} \right) \right] \right] \quad \text{definition of } \widehat{Q}_C; \text{ Lemma 3.1 for } Q^* \\
&= \mathbb{E}_{C \sim \mathcal{C}} \left[\mathbb{E}_{x \sim \mathcal{D}} \left[\log \left(\frac{\Pr_{\mathcal{D}}[x]}{\Pr_{\widehat{\mathcal{D}}_C}[x]} \right) - \log \left(\frac{\Pr_{f(\mathcal{D})}[f(x)]}{\Pr_{f(\widehat{\mathcal{D}}_C)}[f(x)]} \right) \right] \right] \quad \text{rearrange terms} \\
&= \mathbb{E}_{C \sim \mathcal{C}} \left[\mathbb{E}_{x \sim \mathcal{D}} \left[\log \left(\frac{\Pr_{\mathcal{D}}[x]}{\Pr_{\widehat{\mathcal{D}}_C}[x]} \right) \right] - \mathbb{E}_{y \sim f(\mathcal{D})} \left[\log \left(\frac{\Pr_{f(\mathcal{D})}[y]}{\Pr_{f(\widehat{\mathcal{D}}_C)}[y]} \right) \right] \right] \quad \text{since } x \sim \mathcal{D} \implies f(x) \sim f(\mathcal{D}) \\
&= \mathbb{E}_{C \sim \mathcal{C}} \left[D_{\text{KL}}(\mathcal{D} \parallel \widehat{\mathcal{D}}_C) - D_{\text{KL}}(f(\mathcal{D}) \parallel f(\widehat{\mathcal{D}}_C)) \right] \quad \text{definition of } D_{\text{KL}}
\end{aligned}$$

This connection between excess heat dissipation and the drop in KL-divergence associated with computing f on a device optimized for the wrong distribution over inputs is well known in the stochastic thermodynamics of computation literature.³² Since KL-divergence is non-negative, $E_{C \sim \mathcal{C}} [D_{\text{KL}}(\mathcal{D} \parallel \widehat{\mathcal{D}}_C) - D_{\text{KL}}(f(\mathcal{D}) \parallel f(\widehat{\mathcal{D}}_C))] \leq E_{C \sim \mathcal{C}} [D_{\text{KL}}(\mathcal{D} \parallel \widehat{\mathcal{D}}_C)]$ and the expected excess heat dissipation from using $\widehat{\mathcal{D}}$ to pick our choice of $\widehat{Q}$ is at most the expected KL-divergence between $\mathcal{D}$ and $\widehat{\mathcal{D}}$. It is known that the add-one smoothing estimator is well suited to this task. For count vector $C \sim \mathcal{C}$, this estimator sets $\Pr_{\widehat{\mathcal{D}}_C}[x] = (C(x) + 1)/(t + d)$. Braess, Forster, Sauer and Simon [BFSS02] showed that for any true distribution $\mathcal{D}$, this estimator after t samples satisfies

$$\mathbb{E}_{C \sim \mathcal{C}} [D_{\text{KL}}(\mathcal{D} \parallel \widehat{\mathcal{D}}_C)] \leq \log \left(1 + \frac{d-1}{t+1} \right),$$

which is at most $(d-1) \log(e)/(t+1)$.

Thus by using this choice of $\widehat{\mathcal{D}}$ and $\widehat{Q}$ we can set the number of samples $t = \lceil \log(e)d/(\varepsilon\delta) \rceil$ to get that $\mathbb{E}_{C \sim \mathcal{C}} [\mathbb{E}_{x \sim \mathcal{D}} [\widehat{Q}_C(x) - Q^*(x)]] \leq \varepsilon\delta$. By Markov's inequality, this means that the probability over $C \sim \mathcal{C}$ that $\mathbb{E}_{x \sim \mathcal{D}} [\widehat{Q}_C(x) - Q^*(x)] \geq \varepsilon$ is at most δ . Since g operates on a constant number of bits, d is also a constant and this can be done in a polynomial number of steps. $\square$

5 Bounds on thermodynamic costs of fixed circuits

Section 4 was about efficient algorithms for finding optimal heat functions in a given circuit in order to minimize heat released on the uniform input distribution. Lemma 4.8 aside, the hardness results

³²A universal lower bound on EP of this form was first identified by Kolchinsky and Wolpert [KW17] and was later called a *mismatch cost*. The same authors expanded on this concept in [WK20] and we include and re-prove their generalized version as Proposition C.3.

imply efficient algorithms for determining optimal thermodynamic parameters are unlikely to exist in general. Nevertheless we can analyze fixed families of circuits to derive bounds in these fixed cases. We first warm up by analyzing two simple implementations of an n -ary AND function: a balanced tree (depth $\log n$), and a linear chain of (depth n), of 2-input AND gates (Section 5.1). We then study the Batcher even-odd sorting networks, analyzing first the Boolean case (Section 5.2) then the case where each wire carries a whole integer (Section 5.2.2).

For all of these results, the upper and lower bounds we show on entropy production are for EP_C (i.e. they assume that the input distribution is sampled independently from the outside world). This assumption is perhaps stronger than it looks. Since EP measures the total increase in entropy of the universe, this is essentially saying the inputs, which presumably came from somewhere in the universe, are distributed independently of the state of the rest of the universe. Nevertheless, upper bounds on EP_C which are smaller than Q are still worth studying. When $\text{EP}_C \ll Q$ we know that most of the released heat is offsetting the drop in entropy of the marginal input distribution. When $\text{EP}_C = 0$ we know that no circuit could compute the same mapping of inputs to outputs while releasing less expected heat.

5.1 Warm-up: entropy-depth tradeoff for computing AND

We note that the circuits we consider in this example are compatible with the overwriting model. Thus the bounds we show here apply equally well to that model of computation under the assumption that inputs to the circuit across runs are independently distributed.

Assume that the input is drawn from the uniform distribution U_n over $\{0, 1\}^n$ and say that you want to cleanly compute the function $\text{AND}(x_1, \dots, x_n) = (x_1 \wedge \dots \wedge x_n)$ where n is a power of two using only $\wedge$ gates of the form $\wedge(x_1, x_2, z) = (x_1, x_2, x_1 \wedge x_2)$ and erasure gates $E(x) = 0$. Per Lemma 3.1 we know that the optimal heat functions for each of these gates depend on their distributions over inputs. Since $\wedge(x_1, x_2, z)$ leaves inputs x_1, x_2 untouched and it overwrites z with a deterministic function only of x_1, x_2 , when inputs x_1, x_2 are independently distributed from z , the optimal heat function for this gate only depends on the marginal probability distribution over z . We can see this as follows for such a product input distribution $\mathcal{D} = \mathcal{D}_{x_1, x_2} \otimes \mathcal{D}_z$:

$$Q_{\wedge}(x_1, x_2, z) = \log \left(\frac{\Pr_{\wedge(\mathcal{D})}[(x_1, x_2, x_1 \wedge x_2)]}{\Pr_{\mathcal{D}}[(x_1, x_2, z)]} \right) = \log \left(\frac{\Pr_{\mathcal{D}_{x_1, x_2}}[(x_1, x_2)]}{\Pr_{\mathcal{D}_{x_1, x_2}}[(x_1, x_2)] \Pr_{\mathcal{D}_z}[z]} \right) = \log \left(\frac{1}{\Pr_{\mathcal{D}_z}[z]} \right)$$

Similarly, per Lemma 3.1, the optimal heat function for E on input distribution $\mathcal{D}$ is given by:

$$Q_E(x) = \log \left(\frac{1}{\Pr_{\mathcal{D}}[x]} \right)$$

Importantly we also have that the expected heat release of these gates is given by $\mathbb{E}_{x \sim \mathcal{D}}[Q_E(x)] = H_2(\Pr_{\mathcal{D}}[0]) = H_2(\Pr_{\mathcal{D}}[1])$.

With these optimal heat functions in mind, we will now analyze the local entropy production of two different fixed Boolean circuits computing $\text{AND}(x_1, \dots, x_n) = (x_1 \wedge \dots \wedge x_n)$ using these gates.

The first we consider is the binary tree composed of $\wedge(x_1, x_2, z)$ gates on pairs of inputs x_1, x_2 and an initially zeroed wire z . After executing each of these gates, we apply $E(x)$ gates on the wires containing the values of x_1 and x_2 . Since each $\wedge(x_1, x_2, z)$ gate in this circuit is run with $z = 0$, they can all be implemented with the heat function $Q_{\wedge}(x_1, x_2, 0) = 0$, $Q_{\wedge}(x_1, x_2, 1) = \infty$ and have an expected heat release of zero. Thus it is only necessary to keep track of the expected heat release

of the erasure gates. The erasure gates deployed after applying the t 'th layer of $\wedge(x_1, x_2, z)$ gates are each erasing bits that have the value 1 with probability 2^{-2^t} . Thus the expected heat release of each of these gates is $H_2(2^{-2^t})$. Since there are $n/2^t$ such erasure gates in layer t , the expected heat released in this layer is $(n/2^t)H_2(2^{-2^t})$. Thus the total expected heat release is

$$\sum_{t=0}^{\log(n)-1} \frac{n}{2^t} H_2(2^{-2^t}).$$

Meanwhile the circuit maps a state with n bits of entropy to only $H_2(2^{-n})$ bits of entropy. By Definition 2.25, the circuit entropy production is therefore:

$$\text{EP}_{\mathcal{C}} = H_2(2^{-n}) - n + \sum_{t=0}^{\log(n)-1} \frac{n}{2^t} H_2(2^{-2^t}) = \sum_{t=1}^{\log(n)} \frac{n}{2^t} H_2(2^{-2^t}) \geq \frac{n}{2} H_2(1/4).$$

Which is $\Omega(n)$.

Next we consider the serial circuit where the i 'th $\wedge(x_1, x_2, z)$ gate has the AND of the first i inputs as its first argument, x_{i+1} as its second argument, and 0 as its third argument. The circuit then applies erasure gates on the wires containing inputs x_1 and x_2 . This circuit is illustrated for $n = 4$ in Fig. 1. Again these $\wedge(x_1, x_2, z)$ gates can be implemented with 0 expected heat release under the promise that $z = 0$, so the only source of heat release are the erasure gates. There are a total of n erasure gates in this circuit that erase the individual inputs and $n - 2$ erasure gates that erase the partial products $x_1 \wedge x_2$ through $x_1 \wedge \dots \wedge x_{n-1}$. Since the inputs are unbiased bits, the expected heat release of these gates is $H_2(1/2) = 1$ per gate and n across all gates. For the partial product erasures, the i 'th such gate erases the value of $x_1 \wedge \dots \wedge x_{i+1}$. Again we know that the x_i are independently and uniformly distributed bits and therefore the i 'th partial erasure gate is erasing a bit that has the value 1 with probability 2^{-i-1} . The expected heat release for this gate is therefore $H_2(2^{-i-1})$. Thus our total expected heat release for the circuit is $n + \sum_{i=1}^{n-2} H_2(2^{-i-1})$. Since we have the same overall mapping of inputs to outputs as the binary tree case, our change in entropy is $H_2(2^{-n}) - n$. This gives us a circuit entropy production of

$$\text{EP}_{\mathcal{C}} = (H_2(2^{-n}) - n) + n + \sum_{i=1}^{n-2} H_2(2^{-i-1}) = \sum_{i=2}^n H_2(2^{-i}) \leq \sum_{i=2}^n 2^{-i}(i + \log(e)) \leq (3 + \log(e))/2$$

which is $O(1)$.

Another way one could compute the circuit entropy production of these circuits would be to use Proposition D.1 and sum up the drops of mutual information associated with executing each gate. Since the gates we are considering here are each locally optimal, this would give us the same values of $\text{EP}_{\mathcal{C}}$ that we computed here. Doing so is the primary approach used in prior works (i.e. [YYW26]) to produce lower bounds on the local entropy production of circuits. By using heat functions, we can exactly compute the circuit entropy production by only analyzing the local behavior of the individual gates. This lets us ignore more global properties of the circuit like the drops in mutual information, making our approach better suited to analyzing complex circuits where these mutual information terms are hard to quantify.

5.2 Batcher's odd-even mergesort network

We now give bounds on the circuit entropy production of Batcher's odd-even mergesort network [Bat68]. We note that the Landauer cost of a sorting network acting on bits chosen uniformly

at random is $\Omega(n)$, since for an input with exactly $k \approx n/2$ 0's, it collapses $\binom{n}{k} \geq (n/k)^k$ different inputs to the single output $0^k 1^{n-k}$. As a result, by Proposition 2.20 applied to the whole circuit, any sorting network necessarily releases heat at least $\Omega(n)$. In this section we show that the Batcher network gates can be assigned heat functions meeting this bound on heat released: $O(n)$ (Theorem 5.2). It remains an open question whether entropy production is necessarily $\Omega(n)$ in any sorting network but we are able to show that the Batcher's odd-even mergesort network on binary inputs does have cost $\Omega(n)$ (Theorem 5.4). We then turn to the integer case, where the Landauer cost (i.e. drop in entropy) of a sorting network acting on a uniformly random permutation of $\{0, 1, \dots, n-1\}$ is $\Theta(n \log n)$. In this case, however, we show in Theorem 5.13 that, when the input is the distribution resulting from picking n elements of an ordered set without replacement uniformly at random, then Batcher's network releases heat $\Theta(n \log^2 n)$ with optimal heat functions, which is directly proportional to the size of the network. In Theorem 5.18 we leverage a sorting network of Leighton and Plaxton [LP98], to obtain a sorting network with size $\Theta(n \log^2 n)$ but only $\Theta(n \log n)$ heat dissipation. In contrast, by generalizing to a sorting circuit with more than n wires, it is possible to achieve zero circuit entropy production; see Section 6.1.

The Batcher network is illustrated in Figure 2 with $n = 16$ inputs. Each vertical line represents a comparator, which sorts two bits, putting the smaller of the two on the "upper" line, i.e., the line with the smaller index, and the larger on the "lower" line. If $a \leq b$ then we say that the comparator gate $\text{Comp}(a, b) = \text{Comp}(b, a) = (a, b)$. For simplicity in this section, we will assume that the number of sorting inputs n is a power of two.

More formally, we will index wires w_i for $i \in \{0, \dots, n-1\}$ for n being a power of two. We use $w(\ell)$ to denote the value stored in wire w after layer ℓ . By the definition of Batcher's, we know that $w_0(0) \leq \dots \leq w_{n/2-1}(0)$ and $w_{n/2}(0) \leq \dots \leq w_{n-1}(0)$ as these two halves of the input were sorted during the previous recursive step. We denote a comparator gate acting on indices i and j with $i < j$ as (i, j) . Let $k_\ell = \log(n) - \ell$. For layer 1 our set of gates is

$$G_1 = \{(0, n/2), \dots, (n/2 - 1, n - 1)\}.$$

The set of gates at layer $\ell > 1$ are

$$G_\ell = \{(i, i + 2^{k_\ell}) \mid 2^{k_\ell} \leq i < n - 2^{k_\ell} \text{ and } \lfloor i/2^{k_\ell} \rfloor \equiv 1 \pmod{2}\}. \quad (4)$$

To account for the total number of comparators needed, consider first the number $M(n)$ in the non-recursive part (layers 1 to $\log n$). We have that $M(2) = 1$ and

$$M(n) = 2M(n/2) + n/2 - 1,$$

where the two $M(n/2)$ terms account for the separate sorting of odd-numbered bits and even-numbered bits, and the $n/2 - 1$ accounts for the last layer. Therefore, $M(n) = \Theta(n \log n)$. Now let $C(n)$ be the total number of comparator gates needed to sort $n = 2^k$ bits. We have that $C(2) = 1$ and for $n = 2^k \geq 4$,

$$C(n) = 2C(n/2) + M(n) = 2C(n/2) + \Theta(n \log n).$$

Therefore $C(n) = \Theta(n \log^2 n)$.

5.2.1 Binary inputs

We first consider the case where the input to be sorted is a length n bitstring selected uniformly at random. It is known that the correctness of a sorting network can be determined by only checking its behavior on such inputs.

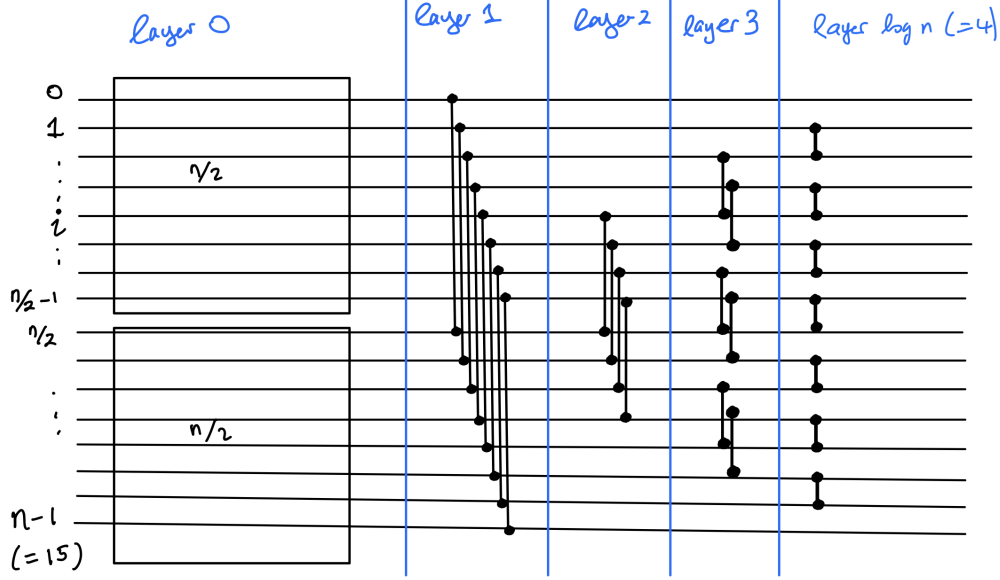


Figure 2: Recursive case of Batcher's odd-even mergesort network for 16 inputs. The base case is $n = 4$ is shown in Figure 3.

Proposition 5.1 ([Knu98]). *A sorting network on n inputs from any ordered set is correct iff it correctly sorts all inputs in $\{0, 1\}^n$.*

The binary input restriction allows for comparator gates to be implemented in a more thermodynamically efficient way. This is because the inputs $(0, 0)$ and $(1, 1)$ are the unique pre-images of outputs $(0, 0)$ and $(1, 1)$ respectively, meaning that we can set $Q(0, 0) = Q(1, 1) = 0$ and we only need to optimize the values of $Q(0, 1)$ versus $Q(1, 0)$ which both map to output $(0, 1)$.

Theorem 5.2. *When given the uniform distribution over input x in $\{0, 1\}^n$, there is an assignment of heat function to every gate that results in the circuit releasing only $O(n)$ expected heat across all gates. Moreover, the construction need only use $2\log(n) + 1$ distinct heat functions.*

Proof. We start by bounding the expected heat released in each layer of the non-recursive merge step. We will use three different types of heat functions in this construction. Q^{unif} is defined such that $Q^{\text{unif}}(x_i, x_j) = 0$ when $x_i = x_j$ and $Q^{\text{unif}}(x_i, x_j) = 1$ when $x_i \neq x_j$. This is an unbiased heat function that we will use either when inputs $(0, 0)$ and $(1, 1)$ are overwhelmingly common or when the number of gates in a group is sufficiently small. This heat function has the advantage that neither input $(0, 1)$ or input $(1, 0)$ releases too much heat in expectation. We also define $Q_\varepsilon^{(0,1)}$ parameterized by $\varepsilon \leq 1/2$ such that $Q_\varepsilon^{(0,1)}(0, 0) = Q_\varepsilon^{(0,1)}(1, 1) = 0$, $Q_\varepsilon^{(0,1)}(0, 1) = \log(1/(1 - \varepsilon))$, and $Q_\varepsilon^{(0,1)}(1, 0) = \log(1/\varepsilon)$. Intuitively, these functions are used to reduce the expected heat released by gates where the first input is heavily biased towards input 0 while the second input is heavily biased towards input 1. In particular, if a gate receives a marginal induced input distribution where the probability of input $(1, 0)$ is at most $\varepsilon \leq 1/2$, then the expected heat released by this gate is at most $(1 - \varepsilon) \log(1/(1 - \varepsilon)) + \varepsilon \log(1/\varepsilon) \leq 2\varepsilon \log(1/\varepsilon)$. Likewise we define $Q_\varepsilon^{(1,0)}$ such that $Q_\varepsilon^{(1,0)}(a, b) = Q_\varepsilon^{(0,1)}(b, a)$ for $\varepsilon \leq 1/2$. These gates are used when the first input is heavily biased towards 1 while the second input is heavily biased towards 0. By the same reasoning as $Q_\varepsilon^{(0,1)}$, if a

gate receives a marginal induced input distribution where the probability of input $(0, 1)$ is at most $\varepsilon \leq 1/2$, then the expected heat released by this gate is at most $2\varepsilon \log(1/\varepsilon)$. In each layer ℓ , we split gates up into one of three regions depending on the value i for the gate $(i, i + 2^{k_\ell})$ and use the same heat function for every gate in the region. We use closed intervals on regions to cleanly handle non-integer values; this has the side-effect of counting some wires as belonging to multiple regions and thus using different heat functions. We will analyze such wires as being a part of both regions and account for costs associated with both heat functions. This only increases our upper bound and significantly simplifies the analysis. Let α be some value in $(0, 1)$ that we pick later and for now assume that $n \geq 32$.

Layer 1 In this layer, we have that $2^{k_\ell} = n/2$. The *upper region* contains all gates with $i \in [0, (n - n^{(1+\alpha)/2})/4]$ and uses the heat function Q^{unif} . Since we know that $w_0(0) \leq \dots \leq w_{n/2-1}(0)$ and $w_{n/2}(0) \leq \dots \leq w_{n-1}(0)$ and the input was distributed uniformly at random, Proposition 2.14 gives us that the probability that $w_i(0) = 1$ as well as the probability that $w_{i+n/2}(0) = 1$ are each at most $e^{-n^\alpha/4}$. Thus by a union bound, the probability that gate $(i, i + n/2)$ receives input $(0, 0)$ is at least $1 - 2 \cdot e^{-n^\alpha/4}$ and so the expected released heat by every such gate is at most $2e^{-n^\alpha/4}$.

The *middle narrow band* contains all gates with $i \in [(n - n^{(1+\alpha)/2})/4, (n + n^{(1+\alpha)/2})/4]$ and uses the heat function Q^{unif} . There are a total of $n^{(1+\alpha)/2}/2 + 1$ gates in this band that each release at most 1 expected heat.

The *lower region* contains all gates with $i \in [(n + n^{(1+\alpha)/2})/4, n/2 - 1]$ and uses the heat function Q^{unif} . Using a symmetrical argument to the upper region case, Proposition 2.14 gives us that the expected released heat of every gate in this region is at most $2e^{-n^\alpha/4}$.

Putting all of this together, the heat released in the first layer is at most $n^{(1+\alpha)/2}/2 + 2ne^{-n^\alpha/4} + 1$.

Layer 2 In this layer, we have that $2^{k_\ell} = n/4$. The *upper narrow band* contains all gates with $i \in [n/4, (n + n^{(1+\alpha)/2})/4]$ and uses the heat function Q^{unif} . There are at most $n^{(1+\alpha)/2}/4 + 1$ gates in this band that each only release at most 1 expected heat.

The *middle region* contains all gates with $i \in [(n + n^{(1+\alpha)/2})/4, (n - n^{(1+\alpha)/2})/2]$ and uses the heat function $Q_\varepsilon^{(1,0)}$ for some $\varepsilon \leq 1/2$ specified later. By the construction of Batchier's odd-even mergesort, we know that $w_{i+n/4}(1) = \max(w_{i+n/4}(0), w_{i-n/4}(0))$. Thus, by Proposition 2.14, the probability that $w_{i+n/4}(1) = 1$ is at most the probability that $w_{i+n/4}(0) = 1$ or $w_{i-n/4}(0) = 1$ which is at most $2e^{-n^\alpha/4}$. Thus the probability each gate in this region receives the input $(0, 1)$ is also at most $2e^{-n^\alpha/4}$. Picking $\varepsilon = 2e^{-n^\alpha/4}$ and $\alpha = 1/2$ gives us that $\varepsilon \leq 1/2$ and thus the expected heat released by any gate in this region is at most $\log(e)e^{-n^\alpha/4}n^\alpha$.

The *bottom narrow band* contains all gates with $i \in [(n - n^{(1+\alpha)/2})/2, n/2 - 1]$ and uses the heat function Q^{unif} . There are only $n^{(1+\alpha)/2}/2$ gates in this band that each only release at most 1 expected heat.

Putting all of this together, the heat released in the second layer is at most $n^{(1+\alpha)/2}/2 + \log(e)e^{-n^\alpha/4}n^{1+\alpha} + 1$.

Layer 3 and beyond The following claim gives us that, after layer 2, most 0's are already in the top half of wires and most 1's are already in the bottom half of wires.

Claim 5.3. *Let $\alpha \in (0, 1)$. Except with probability $3ne^{-n^\alpha/4}/2$, the following hold for every $\ell \geq 2$:*

1. The number of i such that $i < n/2$ and $w_i(\ell) = 1$ is at most $n^{(1+\alpha)/2}/2$.
2. The number of i such that $i \geq n/2$ and $w_i(\ell) = 0$ is at most $n^{(1+\alpha)/2}/2$.

Proof. We start by proving this for $\ell = 2$.

For $i < n/4$, $w_i(2) = w_i(1) = \min(w_i(0), w_{i+n/2}(0))$. We have already established using Proposition 2.14 that when $i \leq (n - n^{(1+\alpha)/2})/4$, each $w_i(0) = 0$ except with probability at most $e^{-n^\alpha/4}$. By a union bound, the probability that there exists such an i with $w_i(2) = 1$ is at most $ne^{-n^\alpha/4}/4$. We give no bounds on the values of $w_i(2) = w_i(1)$ for $i \in [(n - n^{(1+\alpha)/2})/4, n/4 - 1]$, but we note that there are only $n^{(1+\alpha)/2}/4$ such wires.

For $i \in [n/4, n/2 - 1]$, $w_i(2) = \min(w_i(1), w_{i+n/4}(1)) = \min(w_i(1), \max(w_{i+n/4}(0), w_{i-n/4}(0)))$. Again we have already established using Proposition 2.14 that, when $i \in [n/4, n/2 - n^{(1+\alpha)/2}/4]$, both $w_{i+n/4}(0)$ and $w_{i-n/4}(0)$ each equal 0 except with probability at most $e^{-n^\alpha/4}$. Thus by a union bound, their max is 0 except with probability at most $2e^{-n^\alpha/4}$. By a union bound, the probability that there exists such an i with $w_i(2) = 1$ is at most $ne^{-n^\alpha/4}/2$. We give no bounds on the values of $w_i(2)$ when $i \in [n/2 - n^{(1+\alpha)/2}/4, n/2 - 1]$, but we again note that there are only $n^{(1+\alpha)/2}/4$ such wires.

For $i \in [n/2, n - 1]$ the proof is symmetric. By a union bound, the total failure probability across both halves of the argument is at most $3ne^{-n^\alpha/4}/2$ and we have excluded only $n^{(1+\alpha)/2}/2$ wires from each half of the analysis. Note that after applying each comparator gate, the number of 1's in the first $n/2$ wires can only decrease while the number of 1's in the last $n/2$ wires can only increase. Thus the bound will automatically extend to all $\ell > 2$. $\square$

Next, we partition the gates in layer $\ell \geq 3$ into three groups each. The *upper gates* are $(i, i + 2^{k_\ell})$ such that $i + 2^{k_\ell} < n/2$ and they will use the heat function Q^{unif} . By Claim 5.3, except with probability $3ne^{-n^\alpha/4}/2$, at most $n^{(1+\alpha)/2}/2$ upper gates receive an input other than $(0, 0)$. Thus, except with probability at most $3ne^{-n^\alpha/4}/2$, the upper gates collectively release at most $n^{(1+\alpha)/2}/2$ expected heat. With probability at most $3ne^{-n^\alpha/4}/2$, every upper gate releases at most 1 heat. Since there are at most $n/2$ such gates in a layer, the expected heat released by all upper gates is at most $n^{(1+\alpha)/2}/2 + 3n^2e^{-n^\alpha/4}/4$.

The *lower gates* are $(i, i + 2^{k_\ell})$ such that $i \geq n/2$ and they will use the heat function Q^{unif} . The reasoning is symmetric to the upper gates, so we get that the expected heat released by all lower gates is also at most $n^{(1+\alpha)/2}/2 + 3n^2e^{-n^\alpha/4}/4$.

The *crossover gates* are $(i, i + 2^{k_\ell})$ where $i < n/2$ but $i + 2^{k_\ell} \geq n/2$ and they will use the heat function $Q_\varepsilon^{(0,1)}$ for some $\varepsilon \leq 1/2$ that we will specify later. By Claim 5.3, except with probability $3ne^{-n^\alpha/4}/2$, there are at most $n^{(1+\alpha)/2}/2$ crossover gates that receive the input $(1, 0)$. When this is the case, the expected heat released by all $< n$ such gates is less than $n \log(1/(1-\varepsilon)) + n^{(1+\alpha)/2} \log(1/\varepsilon)/2$. When this is not the case, there are $< n$ gates that each release at most $\log(1/\varepsilon)$ expected heat. Thus the overall expected heat released by the crossover gates is less than $n \log(1/(1-\varepsilon)) + n^{(1+\alpha)/2} \log(1/\varepsilon)/2 + 3n^2e^{-n^\alpha/4} \log(1/\varepsilon)/2 \leq 2 \log(e)\varepsilon n + n^{(1+\alpha)/2} \log(1/\varepsilon)/2 + 3n^2e^{-n^\alpha/4} \log(1/\varepsilon)/2$. Picking $\varepsilon = n^{(\alpha-1)/2}$ when $\alpha = 1/2$ makes $\varepsilon \leq 1/2$ and therefore this expression at most $2 \log(e)n^{(1+\alpha)/2} + n^{(1+\alpha)/2} \log(n^{(1-\alpha)/2})/2 + 3n^2e^{-n^\alpha/4} \log(n^{(1-\alpha)/2})/2$.

Thus, in total, the expected heat released in each layer $\ell \geq 3$ is less than $(\log(n^{(1-\alpha)/2})/2 + 4 \log(e))n^{(1+\alpha)/2} + (\log(n^{(1-\alpha)/2}) + 1)3n^2e^{-n^\alpha/4}/2$.

Putting it all together The total expected heat released in layers 1 and 2 is $O(n^{(1+\alpha)/2})$ while the total for layers $\ell \geq 3$ are each $O(n^{(1+\alpha)/2} \log(n))$, which gives a total of $O(n^{(1+\alpha)/2} \log^2 n)$ across all non-recursive layers. Fixing our choice of $\alpha = 1/2$ makes this expression $O(n^{3/4} \log^2 n)$. Since layer 0 is two recursive calls on the halves of the input, we have that if $Q(n)$ is the heat released for inputs of size n , then $Q(n) = 2Q(n/2) + O(n^{3/4} \log^2 n)$ for all $n \geq 32$. When $n < 32$ we use Q^{unif} for all gates and, since the number of gates in this circuit is constant, we also get a constant upper bound for the expected heat released in this base case. Since the last term in the recursive case is sub-linear, the master theorem gives us that $Q(n)$ is $O(n)$. We use heat function Q^{unif} across all recursive calls and we use both $Q_\varepsilon^{(0,1)}$ and $Q_\varepsilon^{(1,0)}$ for at most 1 unique value of ε each per recursive call. Thus there are at most $2 \log(n) + 1$ unique heat functions used in this construction. $\square$

In Theorem B.1 of the appendix, we give a similar proof showing that in the overwriting circuit model (see Section 2), Batcher's sorting network can still be implemented with only $O(n)$ total expected heat released.

The bound in Theorem 5.2 is asymptotically tight. We can show that the circuit entropy production of sorting n bits is also $\Omega(n)$.

Theorem 5.4. *When given the uniform distribution over inputs in $\{0,1\}^n$, for any choice of valid heat functions for each gate in Batcher's odd-even mergesort, the circuit entropy production is $\Omega(n)$.*

Proof. We show that each base case of sorting $n = 4$ inputs has constant circuit entropy production. Since this base case occurs $n/4$ times in the overall sorting network and circuit entropy production is always non-negative, the entropy production of the entire circuit must be $\Omega(n)$. We now analyze the $n = 4$ sorting network, as shown in Figure 3, on the uniform distribution over inputs. The two gates in the first layer each receive inputs 00, 01, 10, and 11 with probability $1/4$. By Lemma 3.1 this means that the expected heat release of each of these gates when given the optimal heat function is $1/2$ and the expected heat released in the first layer is 1. After the first layer, the first and third wires independently have value 1 with probability $1/4$ each while the second and fourth wires independently have value 0 with probability $1/4$ each. Thus by Lemma 3.1 we get that the expected heat release of these gates when given the optimal heat function is $3/8$ and the expected heat released in the second layer is $3/4$. The final gate is a bit more tricky to analyze. Looking at all 16 possible inputs to the circuit, this gate receives input 00 on all inputs to the circuit with 3 or more 0's. Likewise the gate receives input 11 on all inputs to the circuit with 3 or more 1's. The gate receives input 01 on inputs 0011 and inputs 1100 to the circuit while the gate receives input 10 on each of the four remaining inputs with 2 zeroes and 2 ones to the circuit. Thus by Lemma 3.1 the expected heat release of this gate is $(3/8)H_2(1/3)$. Putting all of this together, the expected heat release of a 4 input sorting networks is $7/4 + (3/8)H_2(1/3)$. Meanwhile the circuit maps a uniformly random length 4 bit-string to a binomial distribution on 4 unbiased coin flips. The entropy of this output equals

$$(2 \log(16) + 8 \log(4) + 6 \log(8/3))/16 = (8 + 16 + 18 + 6 \log(1/3))/16 = 21/8 - (3/8) \log(3)$$

and so the drop in entropy is $4 - (21/8 - (3/8) \log(3)) = 11/8 + (3/8) \log(3)$. Since the released heat is $7/4 + (3/8)H_2(1/3)$, the circuit entropy production of the 4 input Batcher's sorting network with optimal heat function is at precisely $1/8$. Since the general Batcher's odd-even mergesort network features this sub-circuit on this input distribution $n/4$ times, the overall circuit entropy production of the network is at least $n/32$ which is $\Omega(n)$. $\square$

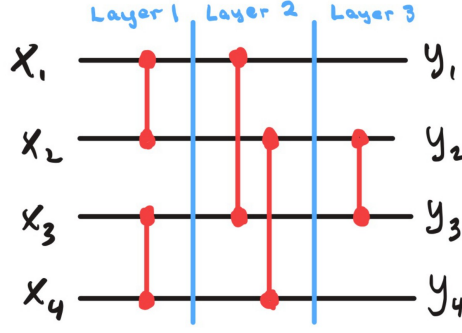


Figure 3: The $n = 4$ base case of Batcher's odd-even mergesort.

5.2.2 Arbitrary inputs

We now consider using Batcher's odd even mergesort for sorting a random permutation of the set $\{0, \dots, n-1\}$. This case is actually sufficient to upper bound the minimal heat dissipation for sorting arbitrary list of n distinct items from an ordered set D .

Lemma 5.5. *Let $\mathcal{C}$ be a sorting network on n inputs. Let $\mathcal{D}_n$ be the uniform distribution over permutations of $\{0, \dots, n-1\}$ and $\mathcal{D}_D$ be the distribution obtained by sampling n elements of D uniformly at random without replacement. Then the minimum expected heat released by any gate g in $\mathcal{C}$ on input distribution $\mathcal{D}_n$ is at most the minimum expected heat released by g in $\mathcal{C}$ on input distribution $\mathcal{D}_D$.*

Proof. Let X_g be a random variable over the inputs to gate g induced by $\mathcal{C}$ receiving input X sampled from $\mathcal{D}_D$ and $V(X)$ be the unordered set of values in X . Using Lemma 3.1, the optimal heat function for g would give a minimum expected heat release of

$$\begin{aligned} \sum_{x_1, x_2} \Pr[X_g = (x_1, x_2)] Q(X_g) &= \sum_{x_1, x_2} \Pr[X_g = (x_1, x_2)] \log \left(\frac{\Pr[\text{Comp}(X_g) = \text{Comp}(x_1, x_2)]}{\Pr[X_g = (x_1, x_2)]} \right) \\ &= \mathcal{S}(X_g \mid \text{Comp}(X_g)) \\ &\geq \mathcal{S}(X_g \mid \text{Comp}(X_g), V(X)). \end{aligned}$$

After conditioning on $V(X)$, we can re-label the elements in D so that the elements in $V(X)$ get mapped to $0, \dots, n-1$ in their relative rank order and all remaining elements get mapped to distinct values that are all at least n . After this relabeling, X_g conditioned on $V(X)$ is just a random variable for the induced marginal input distribution when $\mathcal{C}$ has input distribution $\mathcal{D}$. Thus the expected minimum heat dissipation on $\mathcal{D}_D$ is at least that on $\mathcal{D}_n$. $\square$

Thus we will be assuming input distribution $\mathcal{D}_n$ in the rest of this section. By Lemma 3.1 the optimal heat function for our comparator gates on marginal induced input distribution $\mathcal{D}$ when $x_1 \neq x_2$ is given by:

$$Q(x_1, x_2) = \log \left(\frac{\Pr_{\text{Comp}(\mathcal{D})}[\text{Comp}(x_1, x_2)]}{\Pr_{\mathcal{D}}[(x_1, x_2)]} \right)$$

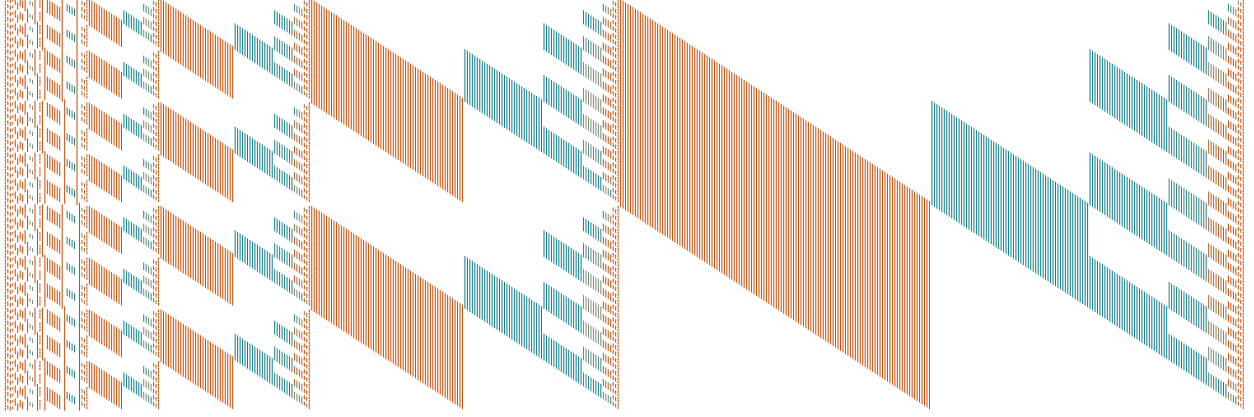


Figure 4: A plot of empirically calculated expected released heat per gate for sorting randomly permuted integers $0, \dots, n - 1$. The algorithm in Lemma 4.8 is used to find approximate heat functions for each gate and then a fresh source of randomness is used to compute the empirical expected heat release. Teal indicates low heat release and orange high heat release. While it appears that there are many orange gates at the beginning of each recursive step, in the later layers this number is only $O(n)$. Thus we use the centrally located orange gates in the later layers of each recursive step to obtain our bound.

$$\begin{aligned}
&= \log \left(\frac{\Pr_{\mathcal{D}}[(x_1, x_2)] + \Pr_{\mathcal{D}}[(x_2, x_1)]}{\Pr_{\mathcal{D}}[(x_1, x_2)]} \right) \\
&= \log(1 + \frac{\Pr_{\mathcal{D}}[(x_2, x_1)]}{\Pr_{\mathcal{D}}[(x_1, x_2)]}).
\end{aligned} \tag{5}$$

This means that to prove a $\Omega(n \log^2 n)$ lower bound on the expected heat release, it is sufficient to show that there are $\Omega(n \log^2 n)$ comparator gates where, on their induced marginal input distribution $\mathcal{D}$, for every pair $(x_1, x_2) \in \text{supp}(\mathcal{D})$ we have that there are universal constants $a, b > 0$ such that $a \leq \frac{\Pr_{\mathcal{D}}[(x_1, x_2)]}{\Pr_{\mathcal{D}}[(x_2, x_1)]} \leq b$. In particular, the gates that we show have this property will be ones that are centrally located and in the later layers. An empirical plot of the expected heat released per gate in Batchers odd-even mergesort when $n = 256$ is shown in Fig. 4.

Notation and network properties In order to show that these gates have similar probabilities for inputs (x_1, x_2) and (x_2, x_1) , we need some properties about the Batchers odd-even mergesort network. Some of the stated properties below are already established (see, e.g., [Bat68, Knu98]), but the key technical lemma that we require (i.e., Lemma 5.11) appears to be new.

Fact 5.6. *For every $0 \leq r < 2^{k_\ell}$, we have that*

$$\{w_j(0) \mid j \equiv r \pmod{2^{k_\ell}}\} = \{w_j(\ell) \mid j \equiv r \pmod{2^{k_\ell}}\}.$$

Proof. For every $\ell' \leq \ell$ we know that every $(i, j) \in G_{\ell'}$ has $2^{k_{\ell}} \mid (j - i)$. Thus, comparator gates up to level ℓ can only shift the index of elements within these congruency classes. $\square$

The following basic fact about Batchers odd-even sorting network follows directly from the original proof sketch of the network's correctness by Batchers [Bat68]. Hinze and Martin [HM18] give a proof of this result similar to the one we include here.

Proposition 5.7. *After layer ℓ of Batchers odd-even mergesort, the values stored on wires whose indices are congruent mod 2^{k_ℓ} are in their relative sorted order.*

Proof. We prove this by induction on ℓ increasing. In the base case of $\ell = 1$, we have that $2^{k_\ell} = n/2$ and the congruent pairs are the ones that were just compared. Now assume that this holds after layer $\ell - 1$ and let $m = n/2^{k_{\ell-1}}$. Fix any r such that $0 \leq r < 2^{k_\ell}$. We will show that this holds for the wires whose indices are congruent to $r \bmod 2^{k_\ell}$. Let e_j be the j 'th wire whose index is congruent to $r \bmod 2^{k_{\ell-1}}$ and o_j be the j 'th wire whose index is congruent to $r + 2^{k_\ell} \bmod 2^{k_{\ell-1}}$. In other words e_j is the wire at index $r + 2j \cdot 2^{k_\ell}$ and o_j is the wire at index $r + (2j + 1)2^{k_\ell}$. Note that for all $0 \leq j < m - 1$, $(o_j, e_{j+1}) \in G_\ell$. By the inductive hypothesis, we know that

$$e_j(\ell - 1) \leq e_{j+1}(\ell - 1) \text{ and } o_j(\ell - 1) \leq o_{j+1}(\ell - 1)$$

for all j . We also know that $e_j(0) \leq o_j(0)$ since they are on the same half of the input after the recursive call in Batchers. Since the $e_j(\ell - 1)$ are a sorted version of the $e_j(0)$'s and likewise the $o_j(\ell - 1)$ are a sorted version of the $o_j(0)$'s due to Fact 5.6, this implies that

$$e_j(\ell - 1) \leq o_j(\ell - 1)$$

for all j . By a similar argument we know when $j \neq m/2 - 1$ it must be the case that $o_j(0) \leq e_{j+1}(0)$ by the recursive call in Batchers. This maximum of one violation comparing $o_j(0)$ to $e_{j+1}(0)$ gives us that in the sorted orders before applying layer ℓ ,

$$o_j(\ell - 1) \leq e_{j+2}(\ell - 1)$$

for all $0 \leq j < m - 2$. Putting this together we have that:

$$\begin{aligned} e_j(\ell) &\leq \max(o_{j-1}(\ell - 1), e_j(\ell - 1)) \\ &\leq \min(o_j(\ell - 1), e_{j+1}(\ell - 1)) = o_j(\ell) \\ &\leq \max(o_j(\ell - 1), e_{j+1}(\ell - 1)) = e_{j+1}(\ell) \\ &\leq \min(o_{j+1}(\ell - 1), e_{j+2}(\ell - 1)) \leq o_{j+1}(\ell) \end{aligned}$$

for all $1 \leq j < m - 2$. We now have that $e_1(\ell) \leq o_1(\ell) \leq \dots \leq e_{m-2}(\ell) \leq o_{m-2}(\ell)$. We know $o_0(\ell) \leq e_1(\ell)$ and $o_{m-2}(\ell) \leq e_{m-1}(\ell)$ since pairs $o_0(\ell), e_1(\ell)$ and o_{m-2}, e_{m-1} are compared in layer ℓ . Also we know that $e_0(\ell) \leq o_0(\ell)$ and $e_{m-1}(\ell) \leq o_{m-1}(\ell)$ since $e_0(\ell - 1) = e_0(\ell)$ is smaller than both possible values of $o_0(\ell)$ and $o_{m-1}(\ell - 1) = o_{m-1}(\ell)$ is larger than both possible values of $e_{m-1}(\ell)$. Doing this analysis for all r completes the proof. $\square$

Lemma 5.8. *Let g be the gate $(i, i + 2^{k_\ell}) \in G_\ell$ for $\ell \geq 2$ and $r \equiv i \pmod{2^{k_\ell}}$. If Batchers odd even mergesort receives some permutation of $[n]$ as its input, then the smaller input to g is in the set $S_g \subseteq \{i + r - 2^{k_\ell} + 1, \dots, i + r\}$ and the larger input to g is in the set $L_g \subseteq \{i + r + 1, \dots, i + r + 2^{k_\ell}\}$.*

Proof. We prove this by induction on ℓ decreasing. When $\ell = \log n$ we have that the smaller input to g must be i and the larger input to g must be $i + 1$ since the output after this layer is sorted. Now assume the claim holds for all gates in layer $\ell + 1$. Now we split into two cases.

When $r < 2^{k_{\ell-1}}$ then in layer $\ell + 1$ we know that wire i is an input to the gate $g_1 = (i - 2^{k_{\ell-1}}, i)$ and wire $i + 2^{k_\ell}$ is an input to the gate $g_2 = (i + 2^{k_{\ell-1}}, i + 2^{k_\ell})$. By our inductive hypothesis, this means that $S_g \subseteq S_{g_1} \cup L_{g_1} = \{i + r - 2^{k_\ell} + 1, \dots, i + r\}$ and $L_g \subseteq S_{g_2} \cup L_{g_2} = \{i + r + 1, \dots, i + r + 2^{k_\ell}\}$.

When $2^{k_{\ell-1}} \leq r < 2^{k_\ell}$ then in layer $\ell + 1$ we know that wire i is an input to the gate $g_1 = (i, i + 2^{k_{\ell-1}})$ and wire $i + 2^{k_\ell}$ is an input to the gate $g_2 = (i + 2^{k_{\ell-1}}, i + 2^{k_\ell} + 2^{k_{\ell-1}})$. Again our inductive hypothesis gives us that $S_g \subseteq S_{g_1} \cup L_{g_1} \subseteq \{i + r - 2^{k_\ell} + 1, \dots, i + r\}$ and $L_g \subseteq S_{g_2} \cup L_{g_2} = \{i + r + 1, \dots, i + r + 2^{k_\ell}\}$. $\square$

Next we state the key technical lemma that we will need to prove our lower bound on EP_C . We need the following definition of *central gates* in order to state our key lemma, and will use the definition of the hypergeometric distribution in the proof:

Definition 5.9. We say a gate $(i, i + 2^{k_\ell}) \in G_\ell$ is central if $\frac{3n}{8} \leq i \leq \frac{5n}{8}$.

Definition 5.10. Let $HG_n^{N,K}[k]$ denote the probability that when n items are taken without replacement from a set of N items of which exactly K are marked that exactly k marked items are taken. This is given by the hypergeometric distribution:

$$HG_n^{N,K}[k] = \binom{N-K}{n-k} \binom{K}{k} / \binom{N}{n},$$

which has mean nK/N . The distribution may have one or two modes, which differ by at most 1 from the mean.

Lemma 5.11. Let n be a sufficiently large power of two and $\ell > \lceil \frac{1}{2} \log n \rceil + 1$ be a layer in Batchier's odd-even mergesort on n inputs. Let $g = (i, i + 2^{k_\ell}) \in G_\ell$ be a central gate and $\mathcal{D}$ be the marginal distribution over input pairs (x_1, x_2) to g induced by giving the sorting network a uniformly random permutation of the set $[n]$ as input. Then there is some constant $\alpha > 1$ (independent of n) such that for every $(x_1, x_2) \in \text{supp}(\mathcal{D})$,

$$\frac{1}{\alpha} \leq \frac{\Pr_{\mathcal{D}}[(x_2, x_1)]}{\Pr_{\mathcal{D}}[(x_1, x_2)]} \leq \alpha.$$

Before we more formally prove this lemma, we give a detailed and more intuitive outline of the strategy. We want to show that if there are k permutations of $[n]$ resulting in a gate $g = (i, i + 2^{k_\ell})$ receiving any feasible input x_1, x_2 , then there must be at least k/α permutations of $[n]$ resulting in g receiving input x_2, x_1 . To prove this, we first note that due to the structure of Batchier's odd even mergesort, if $r \equiv i \pmod{2^{k_\ell}}$ then the top wire inputs to g will have some fixed rank t among the values of all wires after layer zero (the recursive step, which separately sorts the first and second half of the inputs independently) whose indices are congruent to $r \pmod{2^{k_\ell}}$, and bottom wire inputs to g will have rank $t + 1$ among the values of all wires after layer zero whose indices are congruent to $r \pmod{2^{k_\ell+1}}$. This observation is an immediate consequence of Proposition 5.7. We call these sets $\mathcal{O}$ and $\mathcal{E}$ respectively, since they correspond to indices r plus either odd or even multiples of 2^{k_ℓ} respectively.

With this, we now know that the top input to g is always the t 'th ranked element of $\mathcal{O}$ while the bottom input to g is always the $t + 1$ 'st ranked element of $\mathcal{E}$. To prove our lemma, it is sufficient to show that the number of permutations of $[n]$ that lead to x_1 being the t 'th ranked element of $\mathcal{O}$ and x_2 being the $t + 1$ 'st ranked element of $\mathcal{E}$ is within a factor of α of the number of permutations resulting in x_2 being the t 'th ranked element of $\mathcal{O}$ and x_1 being the $t + 1$ 'st ranked element of $\mathcal{E}$.

Unfortunately, this is complicated by the fact that the indices of elements after layer zero (and hence our sets $\mathcal{O}$ and $\mathcal{E}$) are the result of independently sorting the first half and the second half of the input. Thus the sets $\mathcal{O}$ and $\mathcal{E}$ are determined entirely by which inputs started on which half of the wires. We introduce sets A and B containing the values stored on the first and second half of the wires at the beginning of the circuit respectively. Note that, for the uniformly random permutation of $[n]$ as input, the induced equipartition of $[n]$ into A and B is selected uniformly at random. Then the sets $\mathcal{O}$ and $\mathcal{E}$ are entirely determined by A and B .

Let $x_{\min}$ and $x_{\max}$ be the smaller and larger values respectively among x_1, x_2 . We observe that shifting exactly 2^{k_ℓ} elements from A to B (or from B to A) that are smaller than $x_{\min}$ and moving 2^{k_ℓ} elements larger than $x_{\max}$ back results in flipping the sets $\mathcal{O} \cap [x_{\min}, x_{\max}]$ and $\mathcal{E} \cap [x_{\min}, x_{\max}]$, which has the effect of swapping the order of the two inputs to g . Moreover for any A and B resulting in g receiving input x_1, x_2 , swapping any number of elements smaller than $x_{\min}$ between A and B and doing likewise with elements larger than $x_{\max}$ will maintain the fact that g receives input x_1, x_2 . Thus we want to compare the number of possible A with s elements smaller than $x_{\min}$ and a fixed subset I of $\{x_{\min}, \dots, x_{\max}\}$ in A with the number of possible A' with $s \pm 2^{k_\ell}$ elements smaller than $x_{\min}$ and the same fixed subset of I in A' . The remainder of the proof is a (somewhat cumbersome) argument that these values are within a factor of α of one another.

We use $\mathcal{A}_I^s$ to denote the set of all A that have $A \cap \{x_{\min}, \dots, x_{\max}\} = I$ and exactly s elements smaller than $x_{\min}$ in A . The relative size of these sets for different values of s are hyper-geometrically distributed; the ‘marked’ items are elements smaller than $x_{\min}$ and we want to count how many of them end up in set A . Since the probability mass function of the hyper-geometric distribution starts off monotonically non-decreasing until hitting some mode s^* and then is monotonically non-increasing, we immediately have that when $|s - s^*| > 2^{k_\ell}$ it must be the case that $|\mathcal{A}_I^s| \leq \max(|\mathcal{A}_I^{s-2^{k_\ell}}|, |\mathcal{A}_I^{s+2^{k_\ell}}|)$ and we are done.

When $|s - s^*| < 2^{k_\ell}$ the analysis is more involved. We show that the ratio of the squared geometric mean of $|\mathcal{A}_I^{s-2^{k_\ell}}|$ and $|\mathcal{A}_I^{s+2^{k_\ell}}|$ to the value of $|\mathcal{A}_I^s|^2$ is at least some constant, $1/\alpha^2$, by expressing $|\mathcal{A}_I^{s-2^{k_\ell}}| |\mathcal{A}_I^{s+2^{k_\ell}}| / |\mathcal{A}_I^s|^2$ as a telescoping product. With this lower bound in hand, we can use the arithmetic, geometric mean inequality to deduce that $|\mathcal{A}_I^{s-2^{k_\ell}}| + |\mathcal{A}_I^{s+2^{k_\ell}}| \geq |\mathcal{A}_I^s| (2/\alpha)$. This in turn gives us that if $C(a, b)$ is the number of choices for A resulting in g receiving input a, b then we always have that $1/\alpha \leq \frac{C(x_2, x_1)}{C(x_1, x_2)} \leq \alpha$ as desired.

Proof of Lemma 5.11. Let $r \equiv i \pmod{2^{k_\ell}}$. As in the proof of Proposition 5.7, we define wires e_j indexed at $r + 2j \cdot 2^{k_\ell}$ and o_j indexed at $r + (2j + 1)2^{k_\ell}$. Let $\mathcal{E}$ and $\mathcal{O}$ be the set of values found in the $e_j(0)$ and the $o_j(0)$ respectively. By Proposition 5.7 we know that the $e_j(\ell - 1)$ and the $o_j(\ell - 1)$ contain the values in $\mathcal{E}$ and $\mathcal{O}$ respectively in sorted orders. Let t be the index such that $g = (o_t, e_{t+1})$. Thus we know that the input to g is the pair $(o_t(\ell - 1), e_{t+1}(\ell - 1))$.

Next we characterize the sets $\mathcal{E}$ and $\mathcal{O}$. Let A and $B = [n] \setminus A$ be the uniformly random equipartition of our input elements into the first and second halves of the wires respectively at the beginning of the circuit. If $a_0 < a_1 < \dots < a_{n/2-1}$ are the elements of A and $b_0 < b_1 < \dots < b_{n/2-1}$ are the elements of B , then we know that $w_i(0) = a_i$ for all $i < n/2$ and $w_i(0) = b_{i-n/2}$ for all $i \geq n/2$ at the end of layer zero. Then we can characterize our sets as

$$\mathcal{E} = \{a_j \mid j \equiv r \pmod{2^{k_\ell+1}}\} \cup \{b_j \mid j \equiv r \pmod{2^{k_\ell+1}}\}$$

and

$$\mathcal{O} = \{a_j \mid j \equiv r + 2^{k_\ell} \pmod{2^{k_\ell+1}}\} \cup \{b_j \mid j \equiv r + 2^{k_\ell} \pmod{2^{k_\ell+1}}\}.$$

Thus we know that any $x \in \mathcal{E}$ exactly when $x = a_j$ or $x = b_j$ for some $j \equiv r \pmod{2^{k_\ell+1}}$. Likewise $x \in \mathcal{O}$ exactly when $x = a_j$ or $x = b_j$ for some $j \equiv r + 2^{k_\ell} \pmod{2^{k_\ell+1}}$.

Now define $x_{\min}$ and $x_{\max}$ as the min and the max respectively of x_1 and x_2 . It follows from Lemma 5.8 that

$$x_{\min} = i \pm O(\sqrt{n}) \quad x_{\max} = i \pm O(\sqrt{n}), \quad \text{and } 1 \leq x_{\max} - x_{\min} \leq 2\sqrt{n}. \quad (6)$$

Since $g = (i, i + 2^{k_\ell})$ is central, we have that

$$3n/8 - O(\sqrt{n}) \leq x_{\min} \leq x_{\max} \leq 5n/8 + O(\sqrt{n}), \quad (7)$$

and so both $x_{\min}$ and $x_{\max}$ are $\Theta(n)$.

Observe that moving precisely 2^{k_ℓ} elements from A to B (or from B to A) that are smaller than $x_{\min}$ and moving 2^{k_ℓ} elements larger than $x_{\max}$ back results in flipping the sets $\mathcal{E} \cap [x_{\min}, x_{\max}]$ and $\mathcal{O} \cap [x_{\min}, x_{\max}]$.³³ In particular such a change would swap the values of $o_t(\ell - 1)$ and $e_{t+1}(\ell - 1)$, as the even and the odd sets swapped their values in this interval and the moved elements made the number of elements smaller than $x_{\min}$ now in $\mathcal{E}$ one more than the number of elements smaller than $x_{\min}$ previously in $\mathcal{O}$.³⁴ This swap would have the effect of replacing input (x_1, x_2) with (x_2, x_1) and vice versa.

Let $\mathcal{A}_I^s$ be the (possibly empty) subset of equipartitions A where $I = [x_{\min}, x_{\max}] \cap A$ and $s = |[0, x_{\min}] \cap A|$. Importantly, all equipartitions in $\mathcal{A}_I^s$ result in g receiving the same input. Then we can precisely characterize the size of these sets by:

$$|\mathcal{A}_I^s| = HG_{n/2-|I|}^{(n-x_{\max}+x_{\min}-1), x_{\min}}[s] \cdot \binom{n-x_{\max}+x_{\min}-1}{n/2-|I|} = \binom{x_{\min}}{s} \binom{n-x_{\max}-1}{n/2-|I|-s}. \quad (8)$$

To see why the hypergeometric distribution applies, note that $|I|$ elements of A are already fixed, leaving $n/2 - |I|$ elements of A that can be chosen from the set $[0, n-1] - [x_{\min}, x_{\max}]$, which has size $n - x_{\max} + x_{\min} - 1$. Mark the $x_{\min}$ elements in the range $[0, x_{\min} - 1]$; from these, s must be chosen. The remaining $n/2 - |I| - s$ items are chosen from the unmarked elements in the set $[x_{\max} + 1, n - 1]$ which has size $n - x_{\max} - 1$.

From Definition 5.10, the mean of this distribution is

$$\frac{x_{\min}(n/2 - |I|)}{n - (x_{\max} - x_{\min} + 1)}.$$

Since $|I| \leq x_{\max} - x_{\min} + 1 \leq 2^{k_\ell+1} \leq 2\sqrt{n}$, and since $x_{\min} = \Theta(n)$, the mean and modes (which differ from the mean by at most 1) are $x_{\min}/2 \pm O(\sqrt{n})$. We denote the larger mode of this distribution by s^* , and so we have

$$s^* = x_{\min}/2 \pm O(\sqrt{n}) \quad (9)$$

For any s and I where g receives the input pair (x_1, x_2) we know that both $\mathcal{A}_I^{s+2^{k_\ell}}$ and $\mathcal{A}_I^{s-2^{k_\ell}}$ results in g receiving the input pair (x_2, x_1) . We will want to show that there is some constant $\alpha > 1$ such that for every choice of s and I ,

$$|\mathcal{A}_I^{s-2^{k_\ell}}| + |\mathcal{A}_I^{s+2^{k_\ell}}| \geq |\mathcal{A}_I^s| (2/\alpha). \quad (10)$$

Since the hyper-geometric distribution is unimodal, i.e., strictly non-decreasing to s^* , the larger mode, and then strictly non-increasing after that point, for any $s \leq s^* - 2^{k_\ell}$ we immediately have that $|\mathcal{A}_I^s| \leq |\mathcal{A}_I^{s+2^{k_\ell}}|$. Likewise for any $s \geq s^* + 2^{k_\ell}$ we immediately have that $|\mathcal{A}_I^s| \leq |\mathcal{A}_I^{s-2^{k_\ell}}|$.

³³When shifting small elements from B to A , each old a_j with a value in this interval becomes $a_{j+2^{k_\ell}}$ and each old b_j with a value in this interval becomes $b_{j-2^{k_\ell}}$.

³⁴As an illustrative example of this, consider $\ell = 0$ so $\mathcal{E}$ and $\mathcal{O}$ are just the even and odd indices respectively of A and B when sorted. Fix $A = \{0, 1, 2, 3\}$ and $B = \{4, 5, 6, 7\}$. Then $\mathcal{E} = \{0, 2, 4, 6\}$ and $\mathcal{O} = \{1, 3, 5, 7\}$. If we swap values 0 and 7 between A and B , then we get $A' = \{1, 2, 3, 7\}$, $B' = \{0, 4, 5, 6\}$ and $\mathcal{E}' = \{0, 1, 3, 5\}$, $\mathcal{O}' = \{2, 4, 6, 7\}$. We have that $e'_{t+1} = o_t$ and $o'_t = e_{t+1}$ for $t \in \{0, 1, 2\}$.

When $|s - s^*| < 2^{k_\ell}$ we give a lower bound on $\frac{|\mathcal{A}_I^{s-2^{k_\ell}}| |\mathcal{A}_I^{s+2^{k_\ell}}|}{|\mathcal{A}_I^s|^2}$ and then use the arithmetic geometric mean equality to deduce that $\frac{|\mathcal{A}_I^{s-2^{k_\ell}}| |\mathcal{A}_I^{s+2^{k_\ell}}|}{|\mathcal{A}_I^s|^2} \geq (2/\alpha) |\mathcal{A}_I^s|$. Using a telescoping product, we have that:

$$\frac{|\mathcal{A}_I^{s-2^{k_\ell}}| |\mathcal{A}_I^{s+2^{k_\ell}}|}{|\mathcal{A}_I^s|^2} = \prod_{j=1}^{2^{k_\ell}} \frac{|\mathcal{A}_I^{s-j}|}{|\mathcal{A}_I^{s-j+1}|} \frac{|\mathcal{A}_I^{s+j}|}{|\mathcal{A}_I^{s+j-1}|}.$$

We first show that the telescoping product is well-defined, i.e., that all terms $|\mathcal{A}_I^w|$, $s - 2^{k_\ell} \leq w \leq s + 2^{k_\ell}$, are positive. The left binomial term of Eq. (8) for $|\mathcal{A}_I^w|$ is $\binom{x_{\min}}{w}$. From Eq. (7) we have that $x_{\min} = \Theta(n)$. Also, $|w - s^*| \leq |w - s| + |s - s^*| \leq 2\sqrt{n}$ and so, since $s^* = x_{\min}/2 + O(\sqrt{n})$ by Eq. (9), we have

$$w = x_{\min}/2 \pm O(\sqrt{n}).$$

Therefore for sufficiently large n , since $x_{\min} = \Theta(n)$, w is positive and less than the upper index, $x_{\min}$, so the first binomial term, $\binom{x_{\min}}{w}$, is positive.

Now consider the right binomial term of Eq. (8) for $|\mathcal{A}_I^w|$, which is $\binom{n-x_{\max}-1}{n/2-|I|-w}$. We have that $x_{\max} \leq 5n/8 + O(\sqrt{n})$, so the upper index of the binomial is at least $3n/8 - O(\sqrt{n})$. Using the expression for w above, as well as the fact that $|I| \leq x_{\max} - x_{\min} + 1 = O(\sqrt{n})$, Eq. (6), we can bound the lower index as follows:

$$n/2 - |I| - w = n/2 - x_{\min}/2 \pm O(\sqrt{n}) \geq n/2 - x_{\max}/2 \pm O(\sqrt{n}),$$

which again is positive and is less than the upper index for sufficiently large n .

We now expand the telescoping product above:

$$\begin{aligned} & \frac{|\mathcal{A}_I^{s-2^{k_\ell}}| |\mathcal{A}_I^{s+2^{k_\ell}}|}{|\mathcal{A}_I^s|^2} \\ &= \prod_{j=1}^{2^{k_\ell}} \frac{|\mathcal{A}_I^{s-j}|}{|\mathcal{A}_I^{s-j+1}|} \frac{|\mathcal{A}_I^{s+j}|}{|\mathcal{A}_I^{s+j-1}|} \\ &= \prod_{j=1}^{2^{k_\ell}} \frac{(s-j+1)(n/2 - x_{\max} + |I| + s-j)}{(x_{\min} - s + j)(n/2 - |I| - s + j)} \frac{|\mathcal{A}_I^{s+j}|}{|\mathcal{A}_I^{s+j-1}|} \\ &= \prod_{j=1}^{2^{k_\ell}} \frac{(s-j+1)(n/2 - x_{\max} + |I| + s-j)}{(x_{\min} - s + j)(n/2 - |I| - s + j)} \frac{(x_{\min} - s - j + 1)(n/2 - |I| - s - j + 1)}{(s+j)(n/2 - x_{\max} + |I| + s + j - 1)} \\ &= \prod_{j=1}^{2^{k_\ell}} \left(\frac{s-j+1}{s+j} \right) \left(\frac{x_{\min} - s - j + 1}{x_{\min} - s + j} \right) \left(\frac{n/2 - |I| - s - j + 1}{n/2 - |I| - s + j} \right) \left(\frac{n/2 - x_{\max} - 1 + |I| + s - j + 1}{n/2 - x_{\max} - 1 + |I| + s + j} \right). \end{aligned}$$

In the last line, inside the product there are four terms, each of the form $\frac{z-j+1}{z+j}$, where, from our previous derivations, $z = \Theta(n)$ and $j \leq \sqrt{n}$. Therefore for some positive constant c , each of these terms is at least $1 - c/\sqrt{n}$. Therefore, since $k_\ell \leq \sqrt{n}$,

$$\frac{|\mathcal{A}_I^{s-2^{k_\ell}}| |\mathcal{A}_I^{s+2^{k_\ell}}|}{|\mathcal{A}_I^s|^2} \geq (1 - c/\sqrt{n})^{4\sqrt{n}} \geq (1/4)^{4c},$$

where for the last inequality we use the fact that $1/4 \leq (1 - 1/k)^k$ for all $k \geq 2$.

Using the arithmetic geometric mean inequality,

$$\begin{aligned} |\mathcal{A}_I^{s-2^{k_\ell}}| + |\mathcal{A}_I^{s+2^{k_\ell}}| &\geq 2\sqrt{|\mathcal{A}_I^{s-2^{k_\ell}}| |\mathcal{A}_I^{s+2^{k_\ell}}|} \\ &\geq 2(1/4)^{2^c} |\mathcal{A}_I^s| = (2/\alpha) |\mathcal{A}_I^s|, \end{aligned}$$

where the constant α is 4^{2^c} . We have now proven (10) in all cases, Let $C(x_1, x_2)$ denote the number of choices of set A such that g will receive input (x_1, x_2) . Then:

$$C(x_1, x_2) = \sum_{\substack{s, I \\ \mathcal{A}_I^s \text{ gives input } (x_1, x_2)}} |\mathcal{A}_I^s| \leq (\alpha/2) \sum_{\substack{s, I \\ \mathcal{A}_I^s \text{ gives input } (x_1, x_2)}} |\mathcal{A}_I^{s-2^{k_\ell}}| + |\mathcal{A}_I^{s+2^{k_\ell}}| \leq \alpha C(x_2, x_1),$$

where the last inequality comes from the fact that this sum is counting each possible A at most twice. By symmetry the same argument gives that $C(x_2, x_1) \leq \alpha C(x_1, x_2)$. Since every equipartition is equally likely we can conclude that:

$$1/\alpha \leq \frac{C(x_2, x_1)}{C(x_1, x_2)} = \frac{\Pr_{\mathcal{D}}[(x_2, x_1)]}{\Pr_{\mathcal{D}}[(x_1, x_2)]} \leq \alpha$$

as desired. $\square$

Thermodynamic bounds Now that we have proven Lemma 5.11, we are ready to prove our thermodynamic bounds.

Lemma 5.12. *Let D be an ordered set and $\mathcal{D}$ be the distribution resulting from picking n elements of D without replacement uniformly at random. Let n be sufficiently large so that Lemma 5.11 applies. Any gate in Batcher's odd-even mergesort where Lemma 5.11 can be applied must release $\Omega(1)$ expected heat.*

Proof. Assume for now that $D = \{0, \dots, n-1\}$. By (5) and Lemma 5.11 we know that the optimal heat function for such a gate with induced marginal input distribution $\mathcal{D}$ is given by $Q(x_1, x_2) = \log(1 + \frac{\Pr_{\mathcal{D}}[(x_2, x_1)]}{\Pr_{\mathcal{D}}[(x_1, x_2)]}) \geq \log(1 + 1/\alpha) = \Omega(1)$. Since this applies to every input pair x_1, x_2 , the expected heat dissipation is $\Omega(1)$. This lower bound also applies to arbitrary sets D due to Lemma 5.5. $\square$

Theorem 5.13. *Let D be an ordered set and $\mathcal{D}$ be the distribution resulting from picking n elements of D without replacement uniformly at random. Batcher's odd-even mergesort must release $\Omega(n \log^2 n)$ expected heat on input distribution $\mathcal{D}$. When using the optimal heat function for each gate, this value is $\Theta(n \log^2 n)$.*

Proof. We do this by counting the number of gates where we can apply Lemma 5.12. Let $C(n)$ denote the count of such gates in the circuit of size n . In the non recursive step, when n is sufficiently large, the admissible gates are those in layers $\ell > \lceil (\log n)/2 \rceil + 1$ that are central. As in the proof of Lemma 5.11, let $k_\ell = (\log n) - \ell$ be the width of gates in layer ℓ . There are at least $n/4$ choices of i such that $3n/8 \leq i \leq 5n/8$. By (4), at least half of these i have the correct value of $\lfloor i/2^{k_\ell} \rfloor$ to be either the top or the bottom node of a gate. Of these $n/8$ potential gates, at most 2^{k_ℓ} of them

do not actually exist due to the constraint $2^{k_\ell} \leq i < n - 2^{k_\ell}$ on the starting point of gates. After all this, the number of gates in layer ℓ is at least $n/8 - 2^{k_\ell} \geq n/9$ for sufficiently large n . The number of considered layers is $(\log n) - (\lceil (1/2) \log n \rceil + 1) \geq (\log n)/4$. Thus the number of relevant gates in the non recursive step is at least $(n \log(n))/36$ or $\Theta(n \log n)$ with the asymptotic upper bound coming from the total number of gates in the non-recursive step. Considering in the recursive step, we have:

$$C(n) = 2C(n/2) + \Theta(n \log n)$$

Which is $\Theta(n \log^2 n)$ by the Master theorem.

By Lemma 5.12, each of these gates must release $\Omega(1)$ heat in expectation. Thus the total heat dissipation of the circuit is $\Omega(n \log^2 n)$. To see that this bound is asymptotically tight, observe that the heat function $Q(x_1, x_2) = 1$ is valid. Since there are $O(n \log^2 n)$ total gates in Batchers odd even mergesort, this choice for Q gives a total of $O(n \log^2 n)$ expected heat released. $\square$

Corollary 5.14. *Let D be an ordered set and $\mathcal{D}$ be the distribution resulting from picking n elements of D without replacement uniformly at random. Batchers odd-even mergesort has $\Omega(n \log^2 n)$ circuit entropy production on input distribution $\mathcal{D}$.*

Proof. By Theorem 5.13 we know this circuit releases $\Omega(n \log^2 n)$ expected heat. The drop in entropy by sorting an input is at most $\log(n!)$ which is only $O(n \log n)$. Thus the released heat dominates the drop in entropy term for this circuit. $\square$

We have thus shown that Batchers odd even mergesort network is thermodynamically inefficient.

5.3 A more thermodynamically efficient sorting network

In Section 5.2.2 we showed that Batchers odd even mergesort is a poor choice if one is interested in optimizing the thermodynamic costs of sorting more than just bits, as the minimal expected heat released is asymptotically larger than the drop in entropy. If one is only interested in asymptotic bounds, then there is a trivial way to get a sorting network with $O(n \log n)$ expected heat release by using the AKS sorting network which has only $O(n \log n)$ gates [AKS83]. However, this construction has such a large constant hidden in the O notation that it is unsuitable for actual implementation.

A network sorting almost all inputs In [LP98] the authors present a comparator network that sorts almost every permutation of $[n]$ using only $(3.72 + o(1))n \log n$ comparator gates. We will show how this construction can be leveraged to give a sorting network that releases only $O(n \log n)$ expected heat. To start, we cite the relevant result from their paper.

Proposition 5.15 ([LP98]). *Let $n = 2^d$. For every $\varepsilon(d) = 2^{-2^{o(\sqrt{d})}}$ there is a deterministic (non-uniform) network of depth $(7.44 + o(1)) \cdot d$ that sorts a uniformly random permutation of $[n]$ with probability at least $1 - \varepsilon(d)$.*

Remark 5.16. *In [LP98] the authors actually prove a bound that works for every $\varepsilon(d) = 2^{-2^{o(d)}}$ however, this stronger error bound requires invoking AKS on small subproblems. Thus the enormous constant associated with AKS would still be present in the $o(1)$ term in the depth bound. Instead, we use a weaker version of their result that they explicitly mention at the end of section 6 where AKS is replaced by bitonic sort. This slightly weakens the error bounds but avoids the large constant factor in the overall cost.*

In addition to a depth bound that is linear in d , Proposition 5.15 has only a linear number of gates per layer. Thus the overall gate count is only $O(n \log n)$. We will only need to instantiate this with $\varepsilon(d) = 2^{-\log(d) - \log \log(d)} = 1/(\log(n) \log \log(n))$. The following corollary suffices for our use case.

Corollary 5.17. *Let n be a power of two, D be an arbitrary ordered set and $\mathcal{D}$ be the distribution over inputs given by sampling n elements of D uniformly at random without replacement. Then there exists a deterministic (non-uniform) network with $O(n \log n)$ gates that correctly sorts inputs $x_1, \dots, x_n \sim \mathcal{D}$ except with probability at most $1/(\log(n) \log \log(n))$.*

A sorting network with low expected released heat Now we show that the network in Corollary 5.17 can be used as a subroutine in a thermodynamically efficient sorting network. Intuitively, this network sorts most inputs using only a small number of gates. This allows it to be followed by a larger sorting network which uses a heat function that takes advantage of the fact that its input is often already in sorted order.

Theorem 5.18. *Let D be an arbitrary ordered set and $\mathcal{D}$ be the distribution over inputs given by sampling n elements of D uniformly at random without replacement. There is a thermodynamic sorting network $\mathcal{C}$ with $O(n \log^2 n)$ gates that are each augmented with one of two heat functions Q_0, Q_1 such that, on input distribution $\mathcal{D}$, the expected heat dissipation of $\mathcal{C}$ is $O(n \log n)$.*

Proof. We fix $\mathcal{C}$ to be the concatenation of the sorting network in Corollary 5.17 on all n wires (call this $\mathcal{C}_0$) followed by Batcher's odd even mergesort on the same n wires (call this $\mathcal{C}_1$). Since Batcher's odd even mergesort is a sorting network, we know that $\mathcal{C}$ is also a sorting network. We augment every gate in $\mathcal{C}_0$ with the heat function $Q_0(x_1, x_2) = 1$. Since $\mathcal{C}_0$ has $O(n \log n)$ gates, the expected heat dissipation of this component is also $O(n \log n)$. Importantly, after running $\mathcal{C}_0$, the input is already sorted except with probability at most $1/(\log(n) \log \log(n))$.

We augment every gate in $\mathcal{C}_1$ with the heat function:

$$Q_1(x_1, x_2) = \begin{cases} 1/\log(n) & x_1 \leq x_2 \\ \log\left(\frac{1}{1-2^{-1/\log(n)}}\right) & x_1 > x_2 \end{cases}$$

The validity of Q_1 follows from Lemma 2.22. When $\mathcal{C}_1$ receives an already sorted input from $\mathcal{C}_0$, this choice of Q_1 implies that every gate will dissipate only $1/\log(n)$ heat. Since $\mathcal{C}_1$ has $O(n \log^2 n)$ total gates, the expected released heat conditioned on $\mathcal{C}_0$ sorting its input is only $O(n \log n)$. Conditioned on $\mathcal{C}_0$ failing to sort its input, some gates in the network might release up to

$$\begin{aligned} -\log(1 - 2^{-1/\log n}) &= -\log(1 - e^{-\frac{\ln 2}{\log n}}) \\ &= -\log\left(\frac{\ln 2}{\log n}\right) - \log\left(\frac{1 - e^{-\frac{\ln 2}{\log n}}}{\frac{\ln 2}{\log n}}\right) \\ &= \log \log(n) - \log(\ln(2)) - \log\left(\frac{1 - e^{-\frac{\ln 2}{\log n}}}{\frac{\ln 2}{\log n}}\right) \\ &= \log \log(n) - \log(\ln(2)) + o(1) \quad \text{since } \lim_{x \rightarrow 0} \frac{1 - e^{-x}}{x} = 1 \\ &= O(\log \log n) \end{aligned}$$

expected heat. Even if every gate releases this much, the expected released heat conditioned on $\mathcal{C}_0$ failing to sort its input is only $O(n \log^2(n) \log \log(n))$. Thus the total expected heat released by $\mathcal{C}_1$ is at most:

$$\left(1 - \frac{1}{\log(n) \log \log(n)}\right) \cdot O(n \log^2(n)) / \log(n) + \frac{1}{\log(n) \log \log(n)} \cdot O(n \log^2(n)) \cdot \log \log(n) = O(n \log n)$$

as desired. Since both $\mathcal{C}_0$ and $\mathcal{C}_1$ release only $O(n \log n)$ expected heat, so does $\mathcal{C}$. $\square$

It is important to note that this construction is heavily reliant on the input distribution $\mathcal{D}$. If $\mathcal{D}$ gives heavy weight to inputs that $\mathcal{C}_0$ fails to sort, then the analysis presented here falls apart.

6 Clean zero entropy circuits for compaction and sorting

We start by considering *bit compaction* as a special case of sorting with only two possible values for each key. We give a detailed overview of this circuit with zero circuit entropy production and then show how the idea can be extended to more general sorting.

6.1 A clean compaction circuit with zero entropy production

We describe a bit-sorting (a.k.a., *bit-compaction*) circuit with zero $\text{EP}_{\mathcal{C}}$ (assuming the distribution of the inputs is *exchangeable*, explained below). For any binary string x , write $|x|_1$ to denote the number of 1's in x and $|x|_0 = |x| - |x|_1$ to denote the number of 0's.

First, we note that we are considering compaction *circuits*, rather than sorting networks: these can use more general gates than the comparators used in Sections 5.2, 5.2.2 and 5.3, and they are allowed ancilla wires beyond those needed to store inputs and outputs. Although the problem of sorting bits with a sorting network as in Section 5.2 is nontrivial due to Proposition 5.1, general circuits can perform compaction without payloads by simply counting the number of times each key appears in the input. Thus, we add payloads that must be sorted alongside their keys to make the problem non-trivial.

This problem was studied by Asharov, Lin, and Shi [ALS21], who gave an $o(n \log n)$ sized circuit for compaction (when the keys are sufficiently small) without considering the thermodynamic costs.³⁵ The input is n key-value pairs $(b_i, V_{i,:})$ called *records*, where $b_i \in \{0, 1\}$ is a one-bit *key* and

³⁵ See also [LS22]. The problem has a longer history in other models of computation, going back to Pippenger's self-routing superconcentrators [Pip96].³⁶ With comparator gates only, compaction is equivalent to sorting: by Proposition 5.1, a comparator network that sorts every 0/1 key vector sorts every input, so $\Theta(n \log n)$ comparators are necessary and sufficient [AKS83]. With general Boolean gates, Asharov, Lin, and Shi [ALS21] gave the first compaction circuits of size $O(nm)$ for m -bit payloads, up to a $\text{poly}(\log^* n - \log^* m)$ factor that disappears when $m = O(1)$ or $m \geq \log^{(c)} n$ for any constant c , but with superpolylogarithmic depth; Koucký and Král [KK21] reduced the depth to $O(\log^3 n)$ with size $O(nm(1 + \log^* n - \log^* m))$, and Lin and Shi [LS22] to the optimal $O(\log n + \log m)$ with size $O(nm) \cdot \text{poly}(\log^* n - \log^* m)$. (Pippenger's self-routing superconcentrators [Pip96] already give $O(n \log n + nm)$ gates [ALS21]; in the RAM model, oblivious compaction takes linear time [AKL⁺23], but word operations cost $\Omega(\log n)$ gates, so this does not yield linear-size circuits.) All of these circuits are non-stable, necessarily so: stable compaction needs $\Omega(n \log n)$ selector gates if payloads are only routed and never computed on [LSX19], and $\Omega(nm \log n)$ gates in general if the *Li-Li network coding conjecture* [LL04] holds [AFKL19, ALS21]. To our knowledge it remains open whether the $\text{poly}(\log^* n - \log^* m)$ factor can be removed for every m , and no unconditional superlinear lower bound is known for stable compaction, or for sorting keys of length $\Omega(\log n)$, with general gates. The circuits in this paper are much larger than these; they are designed for zero circuit entropy production, not size.

³⁶ A *superconcentrator* is a network of 2-input, 2-output *switches*, with n inputs and n outputs, such that for every

$V \in \{0, 1\}^{n \times m}$ is the payload matrix, whose i 'th row $V_{i,:}$ is the m -bit *payload* value in record i . The goal is to sort the records by the key; for example sorting $(1, a), (0, b), (1, c), (0, d)$ could result in sorted order $(0, d), (0, b), (1, c), (1, a)$. (There is no requirement that the sort be stable, i.e., preserve the original order between two records with equal keys.) The requirement to carry the payload values along with the keys means that we must really track the order of the records as we sort them; simply counting the 1's as above is not sufficient.

Theorem 6.1 assumes the distribution on the input record list is *exchangeable*, which means that any permutation of the n record positions gives the same distribution over inputs. This allows payloads to be arbitrarily correlated with their keys but prevents, for example, a bias towards keys that are almost already sorted or payloads that encode the initial indices of their record.

Our circuit is based on first computing the mapping $(b, V) \mapsto (b, s, W)$, where $b \in \{0, 1\}^n$ represents the unsorted 1-bit keys, $V \in \{0, 1\}^{n \times m}$ represents the associated m -bit payloads, $s \in \{0, 1\}^n$ is the sorted keys, and $W \in \{0, 1\}^{n \times m}$ is the sorted payload matrix. To make the computation clean, we then erase the bits b with zero EP, which requires determining the probability p that a bit b_i is 1 and then routing it to an erasure gate optimal for that probability. It turns out that there are $\Omega(n^2)$ many such probabilities, which implies that this technique requires $\Omega(n^2)$ erasure gates. It remains an open question whether one can erase n bits, which in total have quadratically many possible distributions, using $o(n^2)$ gates, or to compute the desired mapping $(b, V) \mapsto (s, W)$ with zero circuit entropy production using a different technique to obtain $o(n^2)$ total gates. We note that for any register r , both copying $(r, \vec{0}) \mapsto (r, r)$ and moving $(r, \vec{0}) \mapsto (\vec{0}, r)$ can be done reversibly.

The next theorem turns an arbitrary sorting network into a compaction circuit with zero circuit entropy production.

Theorem 6.1. *For all positive integers n and m , for any sorting network C with $t(n)$ gates and depth $d(n)$, there is a circuit $\mathcal{C}$ cleanly solving the 1-bit key, m -bit payload value, compaction problem on n inputs, with $\text{EP}_{\mathcal{C}}(\mathcal{D}) = 0$ for every exchangeable distribution $\mathcal{D}$ over input record lists, using $O(t(n)m + n^2)$ gates, $O(d(n) \log m + n \log n)$ depth, and $O(t(n) + nm)$ ancilla wires.*

Proof. Denote the input $x = (b, V)$, where $b \in \{0, 1\}^n$ represents the n 1-bit keys, and $V \in \{0, 1\}^{n \times m}$ is the payload matrix whose row $V_{i,:}$ is the value associated with key b_i . The correct output of the compaction problem on input (b, V) is $C(b, V) = (s, W)$, where $s = 0^{n-h}1^h$ represents the compacted keys (assuming there are h 1's in b), and $W \in \{0, 1\}^{n \times m}$ represents the associated values, sorted according to the keys. (That is, the final h rows $W_{n-h+1,:}, \dots, W_{n,:}$ are exactly those associated with 1 keys.)

Most of the task of doing compaction with zero circuit entropy production will use standard reversible computing primitives to compute the mapping $(b, V) \mapsto (b, s, W)$. The main difficulty is erasing b with zero EP, once the rest of the output (s, W) is correct and the other input V is reversibly erased. This last erasure of b is the only part of the circuit that is not logically reversible and thus the only part which requires a heat function with non-zero entries. In order to obtain zero circuit entropy production, we must pick these heat functions, per Lemma 3.1, in a way that exactly matches the induced distribution over inputs for each bit being erased. Unfortunately, this induced input distribution depends on values other than the bit being erased (see Appendix D

k and every choice of k inputs and k outputs, some setting of the switches joins them by k vertex-disjoint paths; *self-routing* means the network computes that setting itself from flags presented at its inputs, rather than being handed it by an external routing algorithm. A switch differs from a comparator in that it is told whether to swap by a separate control wire and never inspects the payloads it carries, so compaction in this model splits into a control computation over the n keys and a data path moving the m -bit payloads as opaque cargo.

for why this causes positive entropy production). Thus, it is necessary to determine the induced marginal probability distribution over the bit being erased conditioned on the values stored on all other wires to obtain zero circuit entropy production.

First we explain how to compute the mapping $(b, V) \mapsto (b, s, W)$ reversibly. In the sorting network C , replace each comparator gate between wires i and j with a reversible comparator between records that also outputs (for reversibility) whether a swap occurred:

$$\begin{aligned} & ((b_i, V_{i,:}), (b_j, V_{j,:}), 0) \\ \mapsto & ((b'_i, V'_{i,:}), (b'_j, V'_{j,:}), g), \end{aligned}$$

where the primed values are the identity if $b_i \leq b_j$ and swapped otherwise, with $g = 1$ if a swap occurred and 0 otherwise. To implement this record comparator using constant-size gates, we reversibly distribute g to m clean ancilla wires using a balanced tree of copy gates of depth $O(\log m)$, use those copies to swap all m pairs of payload bits in parallel, and then reverse the copy tree. Thus each comparator of C becomes $O(m)$ gates of depth $O(\log m)$. All record comparators belonging to one layer of C act on disjoint records, so their copy trees and bit swaps can be performed in parallel. The resulting reversible sorting computation has gate complexity $O(t(n)m)$ and depth $O(d(n) \log m)$. Running this modified sorting network C on the records (b, V) results in the sorted output (s, W) described above, along with “garbage” data G indicating which swaps occurred, reversibly computing the mapping

$$\begin{aligned} & (b, V, \vec{0}) \\ \mapsto & (s, W, G). \end{aligned}$$

We now reversibly copy s :

$$\begin{aligned} & (s, W, G, \vec{0}) \\ \mapsto & (s, W, G, s). \end{aligned}$$

Now, “uncompute” C via the mirror-image reverse of the first sorting step, but using swap gates that *only* swap the keys, leaving the sorted payloads in place:

$$\begin{aligned} & (s, W, G, s) \\ \mapsto & (b, W, \vec{0}, s). \end{aligned}$$

This completes the mapping $(b, V) \mapsto (b, s, W)$, using $O(t(n)m)$ gates and $O(d(n) \log m)$ depth.

At this point we *almost* have the clean output $y = (s, W)$, but it remains to erase the unsorted key bits b .

Write $b_{\leq i}$ to denote the i ’th prefix $b_{1..i}$ of b (similarly $b_{< i} = b_{1..i-1}$, $b_{\geq i} = b_{i..n}$, and $b_{> i} = b_{i+1..n}$). Let $h = |b|_1 = |s|_1$. We erase the bits one at a time starting with the first, successively producing $b \mapsto 0b_{\geq 2} \mapsto 00b_{\geq 3} \mapsto \dots \mapsto 0^{n-2}b_{n-1}b_n \mapsto 0^{n-1}b_n \mapsto 0^n$. Suppose we currently have $0^{i-1}b_{\geq i}$ and now need to erase b_i ; call this *stage i*. The correct distribution to assume when erasing b_i when we apply Lemma 3.1 is such that the probability of a one is $|b_{\leq i}|_1/i$; we justify this at the end of the proof. The denominator i is a fixed constant in stage i of this circuit, so the key is to make the numerator $z_i = |b_{\leq i}|_1$ reversibly available in a binary register Z , which will then be used to route b_i to a “controlled erasure” gate with an optimal heat function for this probability.³⁷ We explain below how to maintain Z incrementally as we iterate through the bits of b .

³⁷A controlled erasure gate takes two input bits: c and b . If the control bit $c = 0$, then it leaves b alone (and releases 0 heat), otherwise it erases b . We need these because there will be a “bank” of several that might be used, but only one will actually be used to erase b_i , i.e., will have its $c = 1$; the other gates will have $c = 0$.

An example of the erasure subcircuit for stage i is shown in Figure 5. There are $i + 1$ pairs of clean wires, one pair for each possible numerator $j \in \{0, \dots, i\}$. Set a clean wire to 1 with a NOT gate and, using a reversible $(i + 1)$ -way demultiplexer,³⁸ route the pair $(1, b_i)$ (the first bit is a control bit c for the controlled erasure gate downstream) to output wires addressed by the binary value z_i in register Z . Thus the selected pair contains $(1, b_i)$ and every other pair contains $(0, 0)$. On pair j , apply a controlled erasure gate to each such pair for $j \in \{0, \dots, i\}$, using a heat function optimal for probability j/i . Consequently only the selected gate performs an erasure, using the heat function optimal for the probability z_i/i ; all other gates act reversibly with zero heat since their control bits are 0. After these gates act in parallel, reverse the demultiplexer (a.k.a., multiplexing) using the same address z_i . It returns the pair $(1, 0)$, after which a NOT gate restores the control wire to 0, while the original b_i wire and all routing wires are also 0.

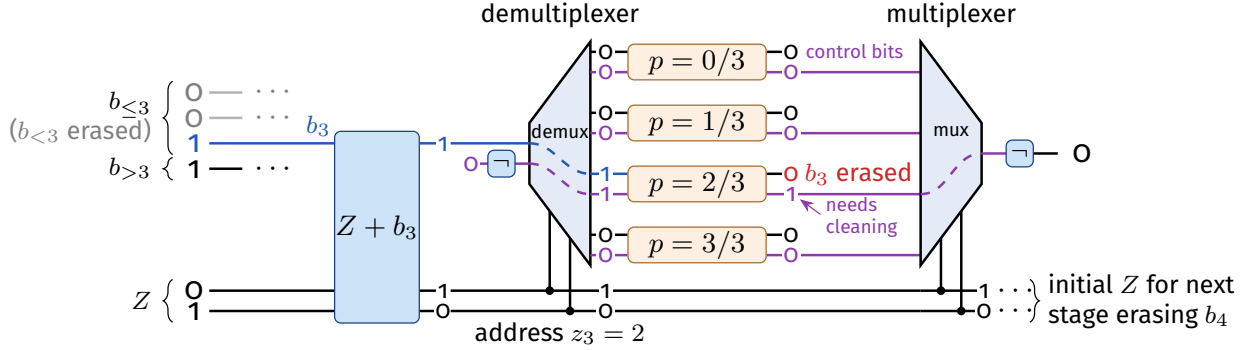


Figure 5: The subcircuit for erasing bit b_3 when $b = 1011$, so $z_3 = |b_{\leq 3}|_1 = 2$ and the gate must assume $\Pr[b_3 = 1] = 2/3$. Reversible demultiplexer drawn without extra clean input wires needed to supply 0's to unaddressed output wires, and similarly for unshown output wires of multiplexer that simply pass through unaddressed inputs.

A binary-tree demultiplexer uses $O(i) = O(n)$ constant-size gates and $O(\log i) = O(\log n)$ depth. Together with the $i + 1$ controlled erasure gates, stage i therefore uses $O(n)$ gates and $O(\log n)$ depth for routing and erasure.

Computing z_i from scratch with a straightforward sequential implementation would require $O(n)$ depth, and since this must be done sequentially for each bit erasure, the total depth would be $O(n^2)$ if computed naively. Instead, we observe that z_i can be updated incrementally using only $O(\log n)$ depth as we erase bits. Maintain a register Z , initially 0. Just before stage i , maintain the invariant that $Z = |b_{< i}|_1$, the number of 1's among the already-erased prefix bits. At stage i , do the reversible increment $(b_i, Z) \mapsto (b_i, Z + b_i)$ to update Z to represent $|b_{\leq i}|_1 = h - |b_{> i}|_1 = z_i$.³⁹ This requires only $O(\log n)$ depth per bit erasure (using a standard ripple-carry increment),⁴⁰ so $O(n \log n)$ total depth for all n bit erasures. After the final erasure, $Z = h = |s|_1$. To restore this

³⁸A demultiplexer takes as input a value v (one or more bits), and an address a interpreted as a binary integer, and routes v to one of several output lines based on the value of a .

³⁹The increment must precede the demultiplexing that selects the erasure gate. If the pre-increment value $Z = z_i - b_i$ were used as the address instead, then (s, W) , $b_{> i}$, and Z would together determine $b_i = z_i - Z$, since $z_i = |s|_1 - |b_{> i}|_1$ is determined by the first two. Erasing b_i would then destroy the mutual information between b_i and the rest of the circuit, which by Proposition D.1 produces entropy no matter which heat function the erasure gate uses. After the increment, $Z = z_i$ carries no information about b_i beyond what (s, W) and $b_{> i}$ already determine.

⁴⁰We could actually do better than $O(\log n)$ depth to increment Z , using $O(\log \log n)$ depth by using a parallel prefix circuit [LF80]. However, we forsake such optimizations, since we are already using $\Omega(\log n)$ depth for the

work register to 0, as required for clean output, apply for each $i \in \{1, \dots, n\}$ in turn the reversible decrement $(Z, s_i) \mapsto (Z - s_i, s_i)$, the inverse of the increment above, with the retained sorted key bit s_i as the control. Since Z starts at $|s|_1$, Z ends at 0. This uses n decrements of $O(\log n)$ gates and depth per bit, i.e., $O(n \log n)$ gates and depth total.

The erasure portion uses $\sum_{i=1}^n (i+1) = \Theta(n^2)$ controlled erasure gates, one for each possible numerator z_i at each stage i ; the forward and reverse demultiplexers use $O(n^2)$ additional reversible gates in total. Together with the reversible sorting computation, the total gate complexity is $O(t(n)m)$ (for the sorting network) plus $O(n^2)$, i.e., $O(t(n)m + n^2)$. The depth is $O(d(n) \log m + n \log n)$ since we have n layers of single-bit erasure, one for each b_i , and each layer requires $O(\log n)$ depth as justified above (plus the $O(d(n) \log m)$ depth of the reversible sorting computation.) The circuit uses $O(t(n) + nm)$ wires beyond the $n(m+1)$ input wires: $n(m+1)$ for the register that receives (s, W) ; $t(n)$ for the garbage bits G ; $O(nm)$ for the copy trees of the comparators in one layer of C , which are returned to 0 after each layer and reused by the next; and $O(n)$ for the register Z , the demultiplexers, and the pairs of wires feeding the controlled erasure gates, which are likewise returned to 0 after each stage and reused by the next.

Now we show that this gives us zero circuit entropy production. Overall, we are mapping $(b, V) \sim \mathcal{D}$ to $C(b, V) = (s, W)$.

Since C is a deterministic function, we have that $\mathcal{S}((b, V)) - \mathcal{S}(C((b, V))) = \mathcal{S}((b, V) \mid (s, W)) = \mathcal{S}(b \mid (s, W))$. Since $\mathcal{D}$ is exchangeable, W is irrelevant to the conditional probability distribution,⁴¹ so this equals $\mathcal{S}(b \mid s) = \mathcal{S}(b \mid |b|_1) = \sum_{k \leq n} \Pr[|b|_1 = k] \cdot \mathcal{S}(b \mid |b|_1 = k)$. Since $\mathcal{D}$ is exchangeable, each b with a fixed value of $|b|_1$ is equally likely, so $\mathcal{S}(b \mid |b|_1 = k) = \log \binom{n}{k} = \log \frac{n!}{k!(n-k)!}$ and the total drop in entropy is:

$$\sum_{k \leq n} \Pr[|b|_1 = k] \log \frac{n!}{k!(n-k)!}. \quad (11)$$

We will now show that our choices of heat function Q_i for erasing each bit of any fixed b with $|b|_1 = k$ result in the circuit releasing exactly this much heat in expectation. We now fix k and any b such that $|b|_1 = k$. As a reminder, we are picking the optimal heat function per Lemma 3.1 for erasing bit b_i based on the probability that b_i is 1 is $|b_{\leq i}|_1 / i$. This gives us the heat function

$$Q_i(b_i, |b_{\leq i}|_0, |b_{\leq i}|_1) = \begin{cases} \log(i / |b_{\leq i}|_0) & b_i = 0 \\ \log(i / |b_{\leq i}|_1) & b_i = 1 \end{cases}$$

for erasing the i 'th bit of b . Let $I = i_1, \dots, i_{n-k}$ denote the indices of b where $b_{i_\ell} = 0$ and $J = j_1, \dots, j_k$ denote the indices of b where $b_{j_\ell} = 1$. Then the expected heat released on input b is exactly

$$\sum_{i_\ell \in I} \log(i_\ell / \ell) + \sum_{j_\ell \in J} \log(j_\ell / \ell) = \sum_{\ell \in [n]} \log(\ell) + \sum_{i \in [n-k]} \log(1/i) + \sum_{j \in [k]} \log(1/j) = \log \frac{n!}{k!(n-k)!}.$$

demultiplexer used to route the bits to erase, so there is no asymptotic benefit in optimizing the increment depth. In fact, given our setup, there is no way to demultiplex in $o(\log n)$ depth, since we need a single value b_i to reach one of $\Omega(n)$ different wires, but due to $O(1)$ size gates, each layer only enables it to reach a constant multiple more wires than the previous. This does not rule out a more sophisticated approach that can achieve $O(1)$ amortized depth per bit erasure, using some sort of “pipelining” for instance, or perhaps some way to erase $\omega(1)$ bits in parallel.

⁴¹The distribution over payloads must be independent of their input index and b is a permutation of s .

Due to the linearity of expectation, the total expected heat released during compaction is exactly

$$\sum_{k \leq n} \Pr[|b|_1 = k] \log \frac{n!}{k!(n-k)!}. \quad (12)$$

Since the drop in entropy (11) and the expected heat release (12) are identical, we can conclude that $\text{EP}_C = 0$. $\square$

Remark 6.2. *The analysis in Theorem 6.1 has a natural extension to compaction on k -bit keys by instead applying Lemma 3.1 to pick heat functions where, for every $d \in \{0, 1\}^k$, the chosen distribution over inputs has $\Pr[b_i = d] = |b_{\leq i}|_d / i$ where $|x|_d$ denotes the count of key d in x .*

6.2 A clean sorting circuit with zero entropy production

Our method of zero circuit entropy production erasure in Section 6.1 can be extended to give a circuit with zero entropy production for sorting. As we are now doing general sorting, we can always assume that payloads are just low order bits of the keys and therefore ignore them altogether. Thus the task is now, given an input $x = x_1, \dots, x_n$ with each $x_i \in \{0, 1\}^m$, output $\text{Sort}(x) = y_1, \dots, y_n$ where the y_i are a permutation of the x_i such that, for every i , $y_i \leq y_{i+1}$. For simplicity, we will assume that our larger keys can be compared with a single comparator gate, which is implemented using multiple constant-arity gates.

Theorem 6.3. *Let $\mathcal{D}$ be any exchangeable distribution over $(\{0, 1\}^m)^n$. There exists a polynomial sized (in terms of n and m) circuit that sorts n inputs that are each m bits with zero circuit entropy production.*

Proof. We will use the same circuit as in Section 6.1 to map input x to the state $(x, \text{Sort}(x))$ using only logically reversible gates. Now, the task is to erase the copy of x with zero circuit entropy production. For $b = b_1, \dots, b_\ell$, and $b_i, v \in \{0, 1\}^k$, we use $|b|_v$ to denote the number of b_i such that $b_i = v$.

We will erase the highest order bit of every x_i before moving on to the next highest order bit and so on. In other words, the order in which we erase bits will be $x_1^1, x_2^1, \dots, x_n^1, x_1^2, x_2^2, \dots, x_n^m$. Let x_i^j denote the j 'th bit of x_i . Analogous to before, we now use $x_{\leq i}^{\geq j}$ to denote the suffixes starting on the j 'th bit of $x_{i'}$ for all $i' \leq i$. When erasing x_i^j , we will use the optimal heat function, per Lemma 3.1, for the distribution with:

$$\Pr[x_i^j = 1] = \frac{|x_{\leq i}^{\geq j}|_{1 \cdot x_i^{\geq j+1}}}{|x_{\leq i}^{\geq j+1}|_{x_i^{\geq j+1}}},$$

where $1 \cdot x_i^{\geq j}$ denotes the suffix of x_i beginning with index j with a 1 appended to the front. In other words, the probability of a 1 is exactly the probability one would get a 1 by sampling uniformly at random the j 'th bit of any $x_{i'}$ with $i' \leq i$ that shares its suffix after index j with x_i . While it would be unfeasible to track these counts for every possible suffix, this probability can be computed on the fly for each erasure by paying a polynomial cost in the depth of the circuit and the number of gates. This follows from the fact that these counts can be computed (and uncomputed) in a

logically reversible fashion even after the actual value of x_i^j has been erased.⁴² We will use the same reversible demultiplexer / multiplexer gadget as in the compaction circuit with controlled erasure gates to selectively pick the optimal heat function to use for each erasure.

Now we argue that this approach gives us a circuit with zero circuit entropy production. First we consider the drop in entropy from the mapping $x \mapsto \text{Sort}(x)$. Let $C(x) \in \mathbb{Z}_{\geq 0}^{\{0,1\}^m}$ be the vector of counts for the number of times each possible $b \in \{0,1\}^m$ appears in the input x . Then we have that $\mathcal{S}(\mathcal{D}) - \mathcal{S}(\text{Sort}(\mathcal{D})) = \mathcal{S}(\mathcal{D} \mid \text{Sort}(\mathcal{D})) = \mathcal{S}(\mathcal{D} \mid C(\mathcal{D}))$ and thus when $x \sim \mathcal{D}$:

$$\begin{aligned} \mathcal{S}(\mathcal{D}) - \mathcal{S}(\text{Sort}(\mathcal{D})) &= \sum_{c \in \mathbb{Z}_{\geq 0}^{\{0,1\}^m}} \Pr_{x \sim \mathcal{D}}[C(x) = c] \mathcal{S}(x \mid C(x) = c) \\ &= \sum_{c \in \mathbb{Z}_{\geq 0}^{\{0,1\}^m}} \Pr_{x \sim \mathcal{D}}[C(x) = c] \log \frac{n!}{\prod_{b \in \{0,1\}^m} c_b!} \end{aligned} \quad (13)$$

Note that this term only has support on c where the 1-norm (i.e. Manhattan norm) of c is exactly n .

As before, we will now show that our choices of heat functions $Q_{i,j}$ for erasing each bit release $\log(n! / \prod_b c_b!)$ expected heat on every input with count vector c . Our heat function for erasing the j 'th bit of the i 'th input will take the form:

$$\begin{aligned} Q_{i,j} \left(x_i^j, \left| x_{\leq i}^{\geq j+1} \right|_{x_i^{\geq j+1}}, \left| x_{\leq i}^{\geq j} \right|_{0 \cdot x_i^{\geq j+1}}, \left| x_{\leq i}^{\geq j} \right|_{1 \cdot x_i^{\geq j+1}} \right) &= \begin{cases} \log \left(\frac{\left| x_{\leq i}^{\geq j+1} \right|_{x_i^{\geq j+1}}}{\left| x_{\leq i}^{\geq j} \right|_{0 \cdot x_i^{\geq j+1}}} \right) & x_i^j = 0 \\ \log \left(\frac{\left| x_{\leq i}^{\geq j+1} \right|_{x_i^{\geq j+1}}}{\left| x_{\leq i}^{\geq j} \right|_{1 \cdot x_i^{\geq j+1}}} \right) & x_i^j = 1 \end{cases} \\ &= \log \left(\frac{\left| x_{\leq i}^{\geq j+1} \right|_{x_i^{\geq j+1}}}{\left| x_{\leq i}^{\geq j} \right|_{x_i^{\geq j}}} \right) \end{aligned}$$

Let's fix any x and let $c = C(x)$. Then the total expected heat released on input x is exactly

$$\begin{aligned} \sum_{i \in [n], j \in [m]} Q_{i,j} \left(x_i^j, \left| x_{\leq i}^{\geq j+1} \right|_{x_i^{\geq j+1}}, \left| x_{\leq i}^{\geq j} \right|_{0 \cdot x_i^{\geq j+1}}, \left| x_{\leq i}^{\geq j} \right|_{1 \cdot x_i^{\geq j+1}} \right) &= \sum_{i \in [n], j \in [m]} \log \left(\frac{\left| x_{\leq i}^{\geq j+1} \right|_{x_i^{\geq j+1}}}{\left| x_{\leq i}^{\geq j} \right|_{x_i^{\geq j}}} \right) \\ &= \sum_{i \in [n]} \sum_{j \in [m]} \log \left(\left| x_{\leq i}^{\geq j+1} \right|_{x_i^{\geq j+1}} \right) - \log \left(\left| x_{\leq i}^{\geq j} \right|_{x_i^{\geq j}} \right) \\ &= \sum_{i \in [n]} \log \left(\left| x_{\leq i}^{\geq m+1} \right|_{x_i^{\geq m+1}} \right) - \log \left(\left| x_{\leq i}^{\geq 1} \right|_{x_i^{\geq 1}} \right) \\ &= \sum_{i \in [n]} \log(i) - \log(C_{x_i}(x_{\leq i})) \quad \text{since } x_i^{\geq m+1} \text{ is the empty string} \end{aligned}$$

⁴²The counts of 0's versus 1's for erased prefixes can be deduced from the sorted output and the non erased part of the input. For example, $\left| x_{\leq i}^{\geq j} \right|_{0 \cdot x_i^{\geq j+1}} = \left| \text{Sort}(x)^{\geq j} \right|_{0 \cdot x_i^{\geq j+1}} - \left| x_{\leq i}^{\geq j} \right|_{1 \cdot x_i^{\geq j+1}}$ and the terms on the right can be computed after x_i^j has been erased. Control gates can be used to ensure that we only count the number of 0's and 1's for inputs with the desired suffix.

$$= \log \frac{n!}{\prod_{b \in \{0,1\}^m} c_b!}.$$

Due to the linearity of expectation, the total expected heat released during sorting is exactly:

$$\sum_{c \in \mathbb{Z}_{\geq 0}^{\{0,1\}^m}} \Pr_{x \sim \mathcal{D}}[C(x) = c] \log \frac{n!}{\prod_{b \in \{0,1\}^m} c_b!}. \quad (14)$$

Since the drop in entropy (13) exactly matches the expected heat released (14) by our circuit, we can conclude that the circuit entropy production is zero. $\square$

7 Conclusion

In this paper we posit that the concepts of entropy production and heat release provide a principled foundation for understanding inherent energy costs of computations. We introduce a thermodynamic logic circuit model in which gates (e.g., Boolean logic gates or comparator gates) are augmented with heat functions, which determine a circuit's entropy flow and entropy production with respect to a given input distribution. We show that in general, determining optimal heat functions is computationally intractable. In contrast, we show that for specific problems (i.e. sorting) our model can be used both to analyze the thermodynamic costs of existing circuits and to design new circuits that are thermodynamically efficient.

There are many directions for future work. Our analysis of Batchier's odd-even mergesort on bits uses a set of gate heat functions that scales with the circuit size; what are the penalties for using only a constant number of distinct heat functions for this problem? How can we analyze heat dissipation of circuits in other problem domains, such as graph algorithms? What are good general design principles for energy-efficient computation?

Theorem 6.1 shows a zero EP circuit for bit-compaction, the problem of sorting 1-bit keys with m -bit payloads, using $O(n \log n)$ depth (as long as $\log m$ is $O(n)$). Can we design a zero EP compaction circuit with $\text{polylog}(n)$ depth? This rules out erasing key bits one at a time as we do, which requires $\Omega(n)$ depth.⁴³ Is it possible to show a lower bound of $\Omega(n)$ depth for the clean compaction problem, arguing that zero EP erasure with constant arity gates must do so sequentially?

Another limitation of our technique to do compaction in Theorem 6.1 is that we require $\Omega(n^2)$ erasure gates. This is necessary given our specific erasure technique: we get the circuit to a point where it has the intended output (compacted key-payload records), as well as the uncompacted keys $b \in \{0,1\}^n$, and we erase each key bit one at a time. There are $\Omega(n^2)$ conditional probabilities that are possible for each bit to be a 1 across all stages during this process, necessitating $\Omega(n^2)$ erasure gates. However, this does not rule out a more sophisticated way to erase the bits in which we do not encounter so many different conditional probabilities. Perhaps some efficient reversible circuit computation could be done on b to transform it into a different bit string b' of length $O(n)$, such that while erasing the bits of b' in order, each bit would only have one of, say, $O(\log n)$ different (possibly position-dependent) probabilities. In this case $O(n \log n)$ total erasure gates would suffice to erase all bits.

Suppose the number of erasure gates could be reduced to $O(n \text{polylog}(n))$, or if $m \gg n$ so that the $O(n^2)$ erasure gates are negligible. (To simplify discussion below assume $m = O(1)$, so that

⁴³It is even worse for our circuit, $\Theta(n \log n)$, due to requiring $\Theta(\log n)$ depth per bit to route it through a $\Theta(\log(n))$ -layer demultiplexer to the erasure gate with appropriate heat function.

swapping m -bit payloads takes $O(1)$ gates in our model.) Then it becomes more important to reduce the size of the compaction/sorting circuit implementing the underlying routing logic in Theorems 6.1 and 6.3. For instance, Theorem 6.1 can very likely be improved to use the $O(n \text{polylog}^*(n))$ -gate compaction circuit of Asharov, Lin, and Shi [ALS21], instead of a sorting network that requires $\Omega(n \log n)$ gates by the 0/1 principle.⁴⁴

Rather than insisting on $\text{EP} = 0$, one can ask for the minimum EP achievable by compaction circuits of near-linear size or polylogarithmic depth. We know of no lower bound of the form “depth d forces $\text{EP} \geq g(n, d)$ ” for any $g > 0$; such a bound would be the first general quantitative trade-off between a standard circuit resource and entropy production.

Theorem 6.1 uses a single circuit, with finitely many heat functions, that has zero EP for every exchangeable input distribution simultaneously, because the conditional probability that each erased bit is 1 depends only on exchangeability. For which other classes of distributions is this possible? If, for instance, the keys are independent but position-dependent, the conditional probabilities depend on the bias of every position, and it is unclear whether finitely many heat functions suffice. Conversely, what is the EP of our circuit when the input is only approximately exchangeable, say as a function of a divergence between the actual distribution and the nearest exchangeable one?

Analogous questions about the gate and depth complexity for zero circuit entropy production sorting are natural. In general, a more detailed study of gate-depth-entropy tradeoffs for sorting is worthy of further investigation.

Our results concern expected EP and expected heat. What are the variance, and the worst case over inputs, of the heat released by our circuits, and is there a trade-off between the fluctuations of heat and circuit size or depth, in the spirit of the thermodynamic uncertainty relations of stochastic thermodynamics [HG20]? Can we obtain concentration guarantees on the distribution of expected heat and entropy production?

Section 5.2 shows that Batchier’s network has $\text{EP} = \Omega(n)$ on uniformly random bits and $\Omega(n \log^2 n)$ on uniformly random distinct inputs, and Section 5.3 gives a network releasing only $O(n \log n)$ expected heat, but, as noted in Section 5.2, it is open whether every sorting network has $\text{EP} = \Omega(n)$, or even $\text{EP} > 0$, on uniformly random bits. (Every sorting network must release $\Omega(n)$ heat, since that is the Landauer cost of sorting bits; zero EP means releasing exactly that much.) A positive lower bound would show that comparators alone cannot achieve zero EP, so that the more general gates of Theorem 6.1 are necessary.⁴⁵ In the other direction, Theorem 6.1 uses $O(t(n) + nm)$ ancilla wires, where $t(n)$ is the size of the underlying sorting network. What is the minimum number of ancilla wires in a zero EP sorting or compaction circuit? In particular, can zero EP be achieved in place, with no ancilla wires, using $O(1)$ -size gates more general than comparators?

Our thermodynamic circuit model is deterministic; we are interested in adapting it to circuits with probabilistic gates, where the output of a gate on a given input is specified as a distribution over the possibilities. We also would like to understand heat dissipation in other models of computation, such

⁴⁴Our main reason for using sorting networks is that it is more straightforward to transform them to be reversible in a black-box way, whereas more sorting circuits allowing more general gates than comparators require more complex hypotheses about how the sorting circuit treats its keys and payloads. For example it is important for our technique that routing decisions are made based only on keys, never payloads, which is true of the networks of Asharov, Lin, and Shi [ALS21] and Lin and Shi [LS22].

⁴⁵Ancilla wires alone would not help a comparator network on bits: a comparator between a data wire and a wire holding a known constant is a data-independent wire crossing (with constant 0 it maps $(x, 0) \mapsto (0, x)$, and with constant 1 it maps $(x, 1) \mapsto (x, 1)$), which is a bijection on its reachable inputs, so it can be assigned zero heat and removed by renaming wires, without changing the outputs or increasing EP.

as random access machines. Can we show trade-offs between time, space, and entropy production? Might it be possible to show average-case hardness results for heat dissipation?

8 Acknowledgments

Niels Kornerup’s contribution to work was supported by the Laboratory Directed Research and Development program (project 240650) at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. The publisher acknowledges that the U.S. Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this written work or allow others to do so, for U.S. Government purposes. The DOE will provide public access to results of federally sponsored research in accordance with the DOE Public Access Plan.

David Wolpert thanks the Santa Fe Institute for their support.

8.1 Discussion of substantive LLM use

In Section 4.2, we used ChatGPT 5.6 to help us strengthen Lemma 4.7 to inputs where it is promised that $\mathbb{E}_{x \sim \mathcal{D}}[Q^*(x)] > 0$. In particular, the LLM pointed us towards Lemma 4.6 as a black-box way to convert any SAT instance ϕ into a new one ψ where (1) there is always some satisfying assignment and (2) there is a programmable gap between the number of satisfying assignments for ϕ depending on whether or not ϕ had some satisfying assignment. We uncovered that the LLM likely arrived at this exact construction based on a similar construction by Blass and Gurevich [BG82] designed to handle (1) and then observing that (2) follows directly from adding additional dummy variables. The LLM also generated a full proof of Lemma 4.7 which we have heavily simplified into the version presented in this paper.

We suspected that the add-one smoothing estimator would be a natural candidate to use in the proof of Lemma 4.8, as it is a simple estimator which avoids the catastrophic pitfall of assigning zero probability on any event with positive probability, and asked the LLM to show that this estimator gives our polynomial time additive approximation. The LLM observed that the excess heat dissipation associated with using an estimator is upper-bounded by the expected KL-divergence between the input distribution and its estimated version. We identified that this is an immediate consequence of what is now known as *mismatch costs* that were first identified by Kolchinsky and Wolpert [KW17]. Next, the LLM proceeded to give a rather convoluted proof that upper bounds the expected KL-divergence between an input distribution and its add-one smoothed estimator. We instead identified that Braess, Forster, Sauer, and Simon [BFSS02] directly proved that the add-one smoothing estimator is well suited for minimizing the expected KL-divergence.

We independently discovered proofs of Lemma 5.12 and Theorem 5.13 where the heat function was restricted to only depend on the value of the comparison (i.e. is $x_1 \leq x_2$?) rather than the actual input values. The LLM helped us find that shifting exactly 2^{k_ℓ} small elements between sets A and B causes a central gate to receive input x_2, x_1 instead of input x_1, x_2 , which was the key ingredient in our proof of Lemma 5.11, that let us extend our lower bound to general heat functions.

The high level strategy in Section 5.3 of combining an $O(n \log n)$ gate almost sorting network with an $O(n \log^2 n)$ gate network to reduce the expected heat dissipation was the result of interactions between the authors and the LLM when searching for a more thermodynamically efficient sorting network on integers than Batchers.

ChatGPT 5.6 and ChatGPT 6 Astra were used to help proofread this document and catch basic errors, which were fixed by the authors. Fig. 4 was generated by code produced with the assistance of Claude code. All text, proofs, and the remaining figures in this paper were produced by the authors, who take full responsibility for the correctness and originality of all content therein.

References

- [AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: A modern approach*, volume 1. Cambridge University Press Cambridge, 2009.
- [AFK⁺98] M. Josephine Ammer, Michael Frank, Tom Knight, Nicole Love, Norman Margolus, and Carlin Vieri. A scalable reversible computer in silicon. In *Unconventional Models of Computation*, pages 183–200. Springer-Verlag, 1998.
- [AFKL19] Peyman Afshani, Casper Benjamin Freksen, Lior Kamma, and Kasper Green Larsen. Lower bounds for multiplication via network coding. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *LIPIcs*, pages 10:1–10:12, 2019.
- [AKL⁺23] Gilad Asharov, Ilan Komargodski, Wei-Kai Lin, Kartik Nayak, Enoch Peserico, and Elaine Shi. OptORAMA: Optimal oblivious RAM. *Journal of the ACM*, 70(1):4:1–4:70, 2023.
- [AKS83] Miklós Ajtai, János Komlós, and Endre Szemerédi. An $O(n \log n)$ sorting network. In David S. Johnson, Ronald Fagin, Michael L. Fredman, David Harel, Richard M. Karp, Nancy A. Lynch, Christos H. Papadimitriou, Ronald L. Rivest, Walter L. Ruzzo, and Joel I. Seiferas, editors, *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 1–9. ACM, 1983.
- [ALS21] Gilad Asharov, Wei-Kai Lin, and Elaine Shi. Sorting short keys in circuits of size $o(n \log n)$. In *Proceedings of the 2021 ACM–SIAM Symposium on Discrete Algorithms (SODA)*, pages 2249–2268. SIAM, 2021.
- [AW22] Shyan Akmal and Ryan Williams. MAJORITY-3SAT (and related problems) in polynomial time. In *FOCS 2021: Proceedings of the 62nd IEEE Annual Symposium on Foundations of Computer Science*, pages 1033–1043. IEEE, 2022.
- [Bat68] K. E. Batchers. Sorting networks and their applications. In *Proceedings of the April 30–May 2, 1968, Spring Joint Computer Conference, AFIPS ’68 (Spring)*, page 307–314, New York, NY, USA, 1968. Association for Computing Machinery.
- [Ben73] Charles H. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532, November 1973.

- [Ben82] Charles H. Bennett. The thermodynamics of computation—a review. *International Journal of Theoretical Physics*, 21(12):905–940, 1982.
- [Ben89] Charles H. Bennett. Time/space trade-offs for reversible computation. *SIAM Journal on Computing*, 18(4):766–776, 1989.
- [BFSS02] Dietrich Braess, Jürgen Forster, Tomas Sauer, and Hans Ulrich Simon. How to achieve minimax expected kullback-leibler distance from an unknown finite distribution. In Nicolò Cesa-Bianchi, Masayuki Numao, and Rüdiger Reischuk, editors, *Algorithmic Learning Theory, 13th International Conference, ALT 2002, Lübeck, Germany, November 24-26, 2002, Proceedings*, volume 2533 of *Lecture Notes in Computer Science*, pages 380–394. Springer, 2002.
- [BG82] Andreas Blass and Yuri Gurevich. On the unique satisfiability problem. *Inf. Control.*, 55(1-3):80–88, 1982.
- [BJO21] Rory A. Brittain, Nick S. Jones, and Thomas E. Ouldrige. What would it take to build a thermodynamically reversible universal turing machine? computational and thermodynamic constraints in a molecular design, 2021.
- [BTV01] Harry Buhrman, John Tromp, and Paul M. B. Vitányi. Time and space bounds for reversible simulation. *Journal of Physics A: Mathematical and General*, 34(35):6821–6830, 2001.
- [CT06] Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006.
- [DKL⁺24] David Doty, Niels Kornerup, Austin Luchsinger, Leo Orshansky, David Soloveichik, and Damien Woods. Harvesting Brownian motion: Zero energy computational sampling. [arXiv:2309.06957](https://arxiv.org/abs/2309.06957), 2024.
- [DLMT16] Erik D. Demaine, Jayson Lynch, Geronimo J. Mirano, and Nirvan Tyagi. Energy-efficient algorithms. In *Proceedings of the 7th Annual ACM Conference on Innovations in Theoretical Computer Science (ITCS 2016)*, pages 321–332. ACM, 2016.
- [FA17] Michael P. Frank and M. Josephine Ammer. Relativized separation of reversible and irreversible space-time complexity classes. *CoRR*, abs/1708.08480, 2017.
- [FBT⁺20] Michael P. Frank, Robert W. Brocato, Brian D. Tierney, Nancy A. Missert, and Alexander H. Hsia. Reversible computing with fast, fully static, fully adiabatic CMOS. In *International Conference on Rebooting Computing, ICRC 2020, Atlanta, GA, USA, December 1-3, 2020*, pages 1–8. IEEE, 2020.
- [Fra02] Michael P. Frank. The physical limits of computing. *Comput. Sci. Eng.*, 4(3):16–26, 2002.
- [Fra17] Michael P Frank. Foundations of generalized reversible computing. In *International Conference on Reversible Computation*, pages 19–34. Springer, 2017.

- [Fra18] Michael P. Frank. Physical foundations of landauer’s principle. In Jarkko Kari and Irek Ulidowski, editors, *Reversible Computation - 10th International Conference, RC 2018, Leicester, UK, September 12-14, 2018, Proceedings*, volume 11106 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2018.
- [FT82] Edward Fredkin and Tommaso Toffoli. Conservative logic. *International Journal of Theoretical Physics*, 21(3–4):219–253, 1982.
- [Gil74] John T. Gill. Computational complexity of probabilistic Turing machines. In *Proceedings of the 6th Annual ACM Symposium on Theory of Computing*, STOC ’74, pages 91–95. Association for Computing Machinery, 1974.
- [HG20] Jordan M. Horowitz and Todd R. Gingrich. Thermodynamic uncertainty relations constrain non-equilibrium fluctuations. *Nature Physics*, 16(1):15–20, 2020.
- [HM18] Ralf Hinze and Clare Martin. Batcher’s odd-even merging network revealed. *J. Funct. Program.*, 28:e14, 2018.
- [Hoe63] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.
- [Int26] International Energy Agency. Key questions on energy and AI. Technical report, International Energy Agency, Paris, 2026. Licensed under CC BY 4.0.
- [KK21] Michal Koucký and Karel Král. Sorting short integers. In *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *LIPIcs*, pages 88:1–88:17, 2021.
- [KMPA14] Jonne V. Koski, Ville F. Maisi, Jukka P. Pekola, and Dmitri V. Averin. Experimental realization of a szilard engine with a single electron. *Proceedings of the National Academy of Sciences*, 111(38):13786–13789, 2014.
- [Knu98] Donald E. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Addison-Wesley, 2nd edition, 1998.
- [Kor25] Niels Kornerup. *Time, Space, and Energy in Computation*. PhD thesis, The University of Texas at Austin, 2025.
- [KW17] Artemy Kolchinsky and David H Wolpert. Dependence of dissipation on the initial distribution over states. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(8), August 2017.
- [Lan61] R. Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5(3):183–191, July 1961.
- [LF80] Richard E Ladner and Michael J Fischer. Parallel prefix computation. *Journal of the ACM (JACM)*, 27(4):831–838, 1980.
- [LL04] Zongpeng Li and Baochun Li. Network coding: The case of multiple unicast sessions. In *Proceedings of the 42nd Annual Allerton Conference on Communication, Control, and Computing*, 2004.

- [LMT00] Klaus-Jörn Lange, Pierre McKenzie, and Alain Tapp. Reversible space equals deterministic space. *Journal of Computer and System Sciences*, 60(2):354–367, 2000.
- [LP98] Frank Thomson Leighton and C. Greg Plaxton. Hypercubic sorting networks. *SIAM J. Comput.*, 27(1):1–47, 1998.
- [LS90] Robert Y. Levine and Alan T. Sherman. A note on Bennett’s time-space tradeoff for reversible computation. *SIAM Journal on Computing*, 19(4):673–677, 1990.
- [LS22] Wei-Kai Lin and Elaine Shi. Optimal sorting circuits for short keys. In *SODA 2022: Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 3583–3629. SIAM, 2022.
- [LSX19] Wei-Kai Lin, Elaine Shi, and Tiancheng Xie. Can we overcome the $n \log n$ barrier for oblivious sorting? In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2419–2438, 2019.
- [LTV98] Ming Li, John Tromp, and Paul M. B. Vitányi. Reversible simulation of irreversible computation. *Physica D: Nonlinear Phenomena*, 120(1–2):168–176, 1998.
- [LV96] Ming Li and Paul M. B. Vitányi. Reversibility and adiabatic computation: Trading time and space for energy. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 452(1947):769–789, 1996.
- [Mor08] Kenichi Morita. Reversible computing and cellular automata—a survey. *Theoretical Computer Science*, 395(1):101–131, 2008.
- [Nor13] John D Norton. Brownian computation is thermodynamically irreversible. *Foundations of Physics*, 43:1384–1410, October 2013.
- [Pip96] Nicholas Pippenger. Self-routing superconcentrators. *Journal of Computer and System Sciences*, 52(1):53–60, 1996.
- [PP21] Luca Peliti and Simone Pigolotti. *Stochastic Thermodynamics: An Introduction*. Princeton University Press, 2021.
- [Sag14] Takahiro Sagawa. Thermodynamic and logical reversibilities revisited. *Journal of Statistical Mechanics: Theory and Experiment*, 2014(3), March 2014.
- [SCSB15a] Philipp Strasberg, Javier Cerrillo, Gernot Schaller, and Tobias Brandes. Thermodynamics of stochastic turing machines. *Physical Review E*, 92(4), October 2015.
- [SCSB15b] Philipp Strasberg, Javier Cerrillo, Gernot Schaller, and Tobias Brandes. Thermodynamics of stochastic turing machines. *Phys. Rev. E*, 92:042104, Oct 2015.
- [Sei05] Udo Seifert. Entropy production along a stochastic trajectory and an integral fluctuation theorem. *Physical Review Letters*, 95(4):040602, 2005.
- [Sei12] Udo Seifert. Stochastic thermodynamics, fluctuation theorems and molecular machines. *Reports on Progress in Physics*, 75(12):126001, 2012.

- [Sei25] Udo Seifert. *Stochastic Thermodynamics*. Cambridge University Press, 2025.
- [SFS18] Naoto Shiraishi, Ken Funo, and Keiji Saito. Speed limit for classical stochastic processes. *Phys. Rev. Lett.*, 121:070601, Aug 2018.
- [Szi29] L. Szilard. über die entropieverminderung in einem thermodynamischen system bei eingriffen intelligenter wesen. *Zeitschrift für Physik*, 53(11):840–856, 1929.
- [Tof77] Tommaso Toffoli. Computation and construction universality of reversible cellular automata. *Journal of Computer and System Sciences*, 15(2):213–231, 1977.
- [Tof80] Tommaso Toffoli. Reversible computing. In Jaco W. de Bakker and Jan van Leeuwen, editors, *ICALP 1980: International Colloquium on Automata, Languages and Programming*, volume 85 of *Lecture Notes in Computer Science*, pages 632–644. Springer, 1980.
- [Tur09] Sadi Turgut. Relations between entropies produced in nondeterministic thermodynamic processes. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 79(4):041102, 2009.
- [VdBE15] Christian Van den Broeck and Massimiliano Esposito. Ensemble and trajectory thermodynamics: A brief introduction. *Physica A: Statistical Mechanics and its Applications*, 418:6–16, 2015.
- [Vit05] Paul M. B. Vitányi. Time, space, and energy in reversible computing. In *Proceedings of the 2nd Conference on Computing Frontiers*, pages 435–444. ACM, 2005.
- [WK20] David H Wolpert and Artemy Kolchinsky. Thermodynamics of computing with circuits. *New Journal of Physics*, 22(6), June 2020.
- [WKL⁺24] David H. Wolpert, Jan Korbelt, Christopher W. Lynn, Farita Tasnim, Joshua A. Grochow, Gülce Kardeş, James B. Aimone, Vijay Balasubramanian, Eric De Giuli, David Doty, Nahuel Freitas, Matteo Marsili, Thomas E. Ouldrige, Andréa W. Richa, Paul Riechers, Édgar Roldán, Brenda Rubenstein, Zoltan Toroczkai, and Joseph Paradiso. Is stochastic thermodynamics the key to understanding the energy costs of computation? *Proceedings of the National Academy of Sciences*, 121(45), 2024.
- [Wol19a] David H Wolpert. The stochastic thermodynamics of computation. *Journal of Physics A: Mathematical and Theoretical*, 52(19), 2019. See <https://arxiv.org/abs/1905.05669> for a more recent version.
- [Wol19b] David H Wolpert. Uncertainty relations and fluctuation theorems for Bayes nets. *arXiv preprint arXiv:1911.02700*, 2019.
- [YCW26] Abhishek Yadav, Francesco Caravelli, and David Wolpert. Entropy production bounds for systems running computer programs, 2026.
- [YYW26] Abhishek Yadav, Mahran Yousef, and David Wolpert. A strictly positive lower bound on the thermodynamic cost of running a Boolean circuit, 2026.
- [Zhu24] Jihai Zhu. Realizability of heat released in nondeterministic processes derived from a fluctuation theorem. [arXiv:2311.16269](https://arxiv.org/abs/2311.16269), 2024.

A Basic lemmas

Lemma 2.12. *Let $f : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ be a function. For any set $S \subseteq D$ where $(\mathcal{G}_f)_{|S}$ is defined, it is the unique choice that minimizes $\mathbb{E}_{x \sim \mathcal{D}_{|S}}[f(x)] - \mathcal{S}(\mathcal{D}_{|S})$ over all distributions supported on S .*

Proof. Observe that

$$\begin{aligned} \mathbb{E}_{x \sim \mathcal{D}_{|S}}[f(x)] - \mathcal{S}(\mathcal{D}_{|S}) &= \mathbb{E}_{x \sim \mathcal{D}_{|S}}[f(x) + \log \Pr_{\mathcal{D}_{|S}}[x]] \\ &= \mathbb{E}_{x \sim \mathcal{D}_{|S}}[\log(\Pr_{\mathcal{D}_{|S}}[x]/2^{-f(x)})] \\ &= \mathbb{E}_{x \sim \mathcal{D}_{|S}}[\log(\Pr_{\mathcal{D}_{|S}}[x]/\Pr_{(\mathcal{G}_f)_{|S}}[x])] - \log\left(\sum_{x \in S} 2^{-f(x)}\right) \\ &= D_{\text{KL}}(\mathcal{D}_{|S} \| (\mathcal{G}_f)_{|S}) - \log\left(\sum_{x \in S} 2^{-f(x)}\right). \end{aligned}$$

By Proposition 2.10, we know that $D_{\text{KL}}(\mathcal{D}_{|S} \| (\mathcal{G}_f)_{|S})$ and hence $\mathbb{E}_{\mathcal{D}_{|S}}[f(x)] - \mathcal{S}(\mathcal{D}_{|S})$ is minimized exactly when $\Pr_{\mathcal{D}_{|S}}[x] = \Pr_{(\mathcal{G}_f)_{|S}}[x]$ for every $x \in S$. $\square$

Lemma 2.22. *Let g be a gate in a circuit computing a function $f : D \rightarrow R$. A heat function $Q : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ is valid for g if and only if, for every $y \in R$,*

$$\sum_{x \in f^{-1}(y)} 2^{-Q(x)} \leq 1.$$

Proof. By Definition 2.21 we know that Q is valid precisely when

$$\max_{\mathcal{D}} \sum_{x \in \text{supp}(\mathcal{D})} \Pr_{\mathcal{D}}[x] \left(\log(\Pr_{f(\mathcal{D})}[f(x))] - \log(\Pr_{\mathcal{D}}[x]) - Q(x) \right) \leq 0.$$

Or equivalently when

$$\max_{\mathcal{D}} \sum_{y \in \text{supp}(f(\mathcal{D}))} \Pr_{f(\mathcal{D})}[y] \left(\mathcal{S}(\mathcal{D}_{|f^{-1}(y)}) - \mathbb{E}_{\mathcal{D}_{|f^{-1}(y)}}[Q(x)] \right) \leq 0.$$

By convexity, we can assume without loss of generality that the maximum $\mathcal{D}$ has $\Pr_{f(\mathcal{D})}[y] = 1$ for some fixed choice of y . We can also assume that the maximum $\mathcal{D}$ has no probability mass on any x where $Q(x) = \infty$ unless Q is ∞ on every point and the validity condition is trivial. Additionally, by Lemma 2.12 we know that the maximal $\mathcal{D}_{|f^{-1}(y)}$ is $(\mathcal{G}_Q)_{|f^{-1}(y)}$. Thus the above is equivalent to the claim

$$\max_y \log \left(\sum_{x \in f^{-1}(y)} 2^{-Q(x)} \right) \leq 0.$$

Taking 2 to the power of both sides of this inequality proves our desired bound. $\square$

B Analyzing entropy flow in Batcher's sorting network: overwriting model with binary inputs

Here we show that in the overwriting model, the expected heat dissipation of Batcher's sorting network is $O(n)$, matching our bound for our thermodynamic circuit model. We first describe useful heat functions for our analysis, then use these to bound the expected heat dissipation for layers 1 through $\log n$ of Batcher's network, and finish with a recurrence to bound the total heat dissipation. This proof is very similar to the one for our standard circuit model (i.e. Theorem 5.2).

The overwriting model and its heat functions In the overwriting model, we view a circuit as a DAG where every node represents a gate and edges indicate input dependence. Unlike in our standard circuit model, the gates in the overwriting model retain their output values in a persistent storage between runs and are not required to erase their inputs. As such, the old output to every gate must be *overwritten* by a new value during each run of the circuit. For this section, we will be assuming that inputs to our overwriting circuit are independently and identically distributed and will use $\tilde{Q}$ to denote heat functions for circuits in the overwriting model. If a gate in the overwriting model computes a function $f : D \rightarrow R$, then it is actually computing the function $h(x, y) = (x, f(x))$ for every $x \in D, y \in R$. As such, the possible heat functions for this gate is given by functions $\tilde{Q} : D \times R \rightarrow \mathbb{R}_{\geq 0}$ and the validity criteria matches that of Definition 2.21 when the function being computed is h .

Since our inputs are independently and identically distributed, there is a significantly simpler expression for the optimal heat function given by Lemma 3.1. Let $\mathcal{D}$ be the marginal induced distribution over inputs to our gate (including the output from the previous run), $\mathcal{D}_D$ be the marginal distribution over new input $x \in D$, and $\mathcal{D}_R$ be the marginal distribution over the old output $y \in R$. Then, since $\mathcal{D}$ is independently and identically distribution, we have that $\mathcal{D} = \mathcal{D}_D \otimes \mathcal{D}_R$. Thus the optimal heat function for our gate is given by:

$$\tilde{Q}(x, y) = \log \frac{\Pr_{x, y \sim h(\mathcal{D})}[(x, f(x))]}{\Pr_{x, y \sim \mathcal{D}}[x, y]} = \log \frac{\Pr_{x \sim \mathcal{D}_D}[x]}{\Pr_{x \sim \mathcal{D}_D}[x] \Pr_{y \sim \mathcal{D}_R}[y]} = \log \frac{1}{\Pr_{y \sim \mathcal{D}_R}[y]}.$$

So the optimal heat function for gates run on independently identically distributed inputs in the overwriting model is equivalent to the optimal heat function for erasing the value of $f(x)$. Thus we will be specifying heat functions in the overwriting model whose only inputs are the values that are about to be overwritten.

Batcher's odd-even mergesort on bits Rather than use the optimal heat function for every gate, in our analysis, we'll use the three heat functions (parameterized by ε) shown in the following table:

x_1, x_2	y'_1, y'_2	Pr	$\tilde{Q}_\varepsilon^{(0,0)}$	Pr	$\tilde{Q}_\varepsilon^{(0,1)}$	Pr	$\tilde{Q}_\varepsilon^{(1,1)}$
0,0	0,0	$1 - \varepsilon$	$\log_2(1/(1 - \varepsilon))$	$\varepsilon/2$	$\log_2(2/\varepsilon)$	$\varepsilon/2$	$\log_2(2/\varepsilon)$
0,1 or 1,0	0,1	$\varepsilon/2$	$\log_2(2/\varepsilon)$	$1 - \varepsilon$	$\log_2(1/(1 - \varepsilon))$	$\varepsilon/2$	$\log_2(2/\varepsilon)$
1,1	1,1	$\varepsilon/2$	$\log_2(2/\varepsilon)$	$\varepsilon/2$	$\log_2(2/\varepsilon)$	$1 - \varepsilon$	$\log_2(1/(1 - \varepsilon))$

In the table, preceding the column for each heat function $\tilde{Q}_\varepsilon^{(b,b')}$, are the probabilities for which that function is optimal. For example. $\tilde{Q}_\varepsilon^{(0,0)}$ is optimal when $(y'_1, y'_2) = (0, 0)$ with probability $1 - \varepsilon$,

$(0, 1)$ with probability $\varepsilon/2$, and $(1, 1)$ with probability $\varepsilon/2$ (or equivalently when $(x_1, x_2) = (0, 0)$ with probability $1 - \varepsilon$, $(0, 1)$ or $(1, 0)$ with probability $\varepsilon/2$, and $(1, 1)$ with probability $\varepsilon/2$). The expected heat release of a gate with one of these heat functions, when the input distribution is optimal, is

$$(1 - \varepsilon) \log_2(1/(1 - \varepsilon)) + \varepsilon \log_2(2/\varepsilon) = O(\varepsilon \log(1/\varepsilon)),$$

using the fact that $\log_2(1/(1 - \varepsilon)) = O(\varepsilon)$ by application of Taylor's expansion of $\log(1 - \varepsilon)$. Moreover, on distributions where $(x_1, x_2) = (b, b')$ with probability *at least* $1 - \varepsilon$, the expected heat release of a gate with heat function $\tilde{Q}_\varepsilon^{(b, b')}$ is $O(\varepsilon \log 1/\varepsilon)$.

We will also use heat function $\tilde{Q}^{(\text{unif})}$ that is optimal on the uniform input distribution. Its values are $\tilde{Q}^{(\text{unif})}(0, 0) = \tilde{Q}^{(\text{unif})}(1, 1) = 2$, and $\tilde{Q}^{(\text{unif})}(0, 1) = 1$ and the expected heat release is

$$(1/4)2 + (1/4)2 + (1/2) = 3/2.$$

Moreover, for any input distribution, the expected heat release with respect to $\tilde{Q}^{(\text{unif})}$ is at most 2; it approaches 2 as the probability of inputs $(0, 1)$ or $(1, 0)$ approaches 0.

A key difference between the overwriting model and our circuit model is that *logically reversible* operations do not buy you anything extra in the overwriting model. In Section 5.2 we saw that we could always set $Q(0, 0) = Q(1, 1) = 0$ since these inputs give unique outputs. On the other hand, we need to dissipate heat when we write over these outputs in the overwriting model.

As in the analysis of Batcher's network for our model, we partition the comparator gates into $\log_2 n + 1$ layers, with layer 0 encompassing the recursive sub-networks, as in Figure 2. Within each layer ℓ , we will partition the gates into the same three regions as in the proof of Theorem 5.2, but will pick heat functions that are amenable to the overwriting model. Again $\alpha \in (0, 1)$ is a parameter we will specify later.

Layer 1 In this layer, there are $n/2$ comparator gates. The i 'th gate has input wires i and $i + n/2$.

- Upper region: $i \in [0, (n - n^{(1+\alpha)/2})/4]$. Since just after layer 0, bits $0, 1, \dots, n/2 - 1$ are sorted, and also the remaining $n/2$ bits are independently sorted, it follows from Proposition 2.14 that for i in this region,

$$\Pr[w_i(0) = 1] \leq \exp(-n^\alpha/4) \text{ and also } \Pr[w_{n/2+i}(0) = 1] \leq \exp(-n^\alpha/4). \quad (15)$$

Consequently, the i th comparator gate has the input $(0, 0)$ with probability at least $(1 - \exp(-n^\alpha/4))^2 \geq 1 - 2\exp(-n^\alpha/4)$. If the gate has heat function $\tilde{Q}_{2\exp(-n^\alpha/4)}^{(0,0)}$, the expected heat release is $O(n^\alpha \exp(-n^\alpha/4)) = O(\exp(-n^\alpha/5))$.

- Lower region: $i \in [(n + n^{(1+\alpha)/2})/4, n/2 - 1]$. A similar analysis shows that the i th comparator gate in this region gets $(1, 1)$ as input with probability at least $1 - 2\exp(-n^\alpha/4)$. If the gate has heat function $\tilde{Q}_{2\exp(-n^\alpha/4)}^{(1,1)}$, the expected heat release is $O(\exp(-n^\alpha/5))$.
- Middle band: $i \in [(n - n^{(1+\alpha)/2})/4, (n + n^{(1+\alpha)/2})/4]$. The expected heat release each gate in this region is at most 2 if the gates have heat function $\tilde{Q}^{(\text{unif})}$.

Summing up over all three regions, the total expected heat dissipation for layer 1 is dominated by the middle band, and is $O(n^{(1+\alpha)/2})$, using three different types of heat functions.

Layer 2 There are $n/4$ comparator gates in layer 2; we number them by the index of their upper wire, from $n/4$ to $n/2 - 1$. The i 'th gate has input wires i and $i + n/4$. We partition the gates into three regions: a narrow band of gates at the top, a large middle region, and a narrow band at the bottom.

- Top narrow band: $i \in [n/4, (n + n^{(1+\alpha)/2})/4]$. Using the heat function $\tilde{Q}^{\text{unif}}$, the expected heat dissipation for each gate is at most 2.
- Middle region: $i \in [(n + n^{(1+\alpha)/2})/4, (n - n^{(1+\alpha)/2})/2]$. It follows from Proposition 2.14 that $\Pr[w_i(1) = 1] = \Pr[\min\{w_i(0), w_{n/2+i}(0)\} = 1] = \Pr[w_i(0) = 1]\Pr[w_i(0) = 1] \geq 1 - 2\exp(-n^\alpha/4)$, and $\Pr[w_{i+n/4}(1) = 0] = \Pr[\max\{w_{i-n/4}(0), w_{i+n/4}(0)\} = 0] \geq \Pr[w_{i-n/4}(0) = 0]\Pr[w_{i+n/4}(0) = 0] \geq 1 - 2\exp(-n^\alpha/4)$. So each comparator gate in this range has input $(1, 0)$ with probability at least $1 - 4\exp(-n^\alpha/4)$. Using the heat function $\tilde{Q}^{(0,1)}(4\exp(-n^\alpha/4))$, the expected heat release of each gate is $O(n^\alpha \exp(-n^\alpha/4)) = O(\exp(-n^\alpha/5))$.
- Bottom narrow band: $i \in [(n - n^{(1+\alpha)/2})/2, n/2 - 1]$. Using the heat function $\tilde{Q}^{\text{unif}}$, the expected heat dissipation for each gate is at most 2.

Therefore, the total expected heat dissipation for layer 2 is dominated by the narrow bands, and is $O(n^{(1+\alpha)/2})$, using two different types of heat functions.

Layers 3 and beyond We will reuse Claim 5.3 from the proof of Theorem 5.2 which shows that with high probability, the number of 1-valued wires in the range $[0, n/2 - 1]$, and the number of 0-valued wires in the range $[n/2, n - 1]$, is small.

Claim 5.3. *Let $\alpha \in (0, 1)$. Except with probability $3ne^{-n^\alpha/4}/2$, the following hold for every $\ell \geq 2$:*

1. *The number of i such that $i < n/2$ and $w_i(\ell) = 1$ is at most $n^{(1+\alpha)/2}/2$.*
2. *The number of i such that $i \geq n/2$ and $w_i(\ell) = 0$ is at most $n^{(1+\alpha)/2}/2$.*

We partition gates of each layer ℓ , $3 \leq \ell \leq \log_2 n$, into three groups: upper gates, lower gates, and crossover gates, defined in the cases below. In Figure 2, where $n = 16$, when $\ell = 3$ there are two upper gates, two lower gates, and two crossover gates.

- Upper gates: Both inputs are from wires in the range $[0, \dots, n/2 - 1]$. We assign them the heat function $\tilde{Q}^{(0,0)}(2\exp(-n^\alpha/4))$. On input $(0, 0)$, such gates have expected heat dissipation $O(\exp(-n^\alpha/4))$, and otherwise they have expected heat dissipation $O(n^\alpha \log_2(1/\exp(-n^\alpha/4))) = O(n^{2\alpha})$. It follows from Claim 5.3 that with probability $1 - O(\exp(-n^\alpha/5))$, the number of upper gates that don't have input $(0, 0)$ is $O(n^{(1+\alpha)/2})$. In this scenario, the total (conditional) expected heat dissipation of the upper gates of layer 3 is

$$O(n \exp(-n^\alpha/4)) + O(n^{(1+\alpha)/2} n^{2\alpha}) = O(n^{(1+5\alpha)/2}).$$

With the remaining probability $O(\exp(-n^\alpha/5))$, there may be more gates that don't have input $(0, 0)$; in the worst case none of the $n/8$ gates has input $(0, 0)$. In this alternative scenario the conditional expected heat dissipation is $O(n^{1+2\alpha})$. Taking the expectation over the two scenarios, the total heat dissipation for the upper gates of layer 3 is

$$(1 - O(\exp(-n^\alpha/5)))O(n^{(1+5\alpha)/2}) + O(\exp(-n^\alpha/5)n^{1+2\alpha}) = O(n^{(1+5\alpha)/2}).$$

- Lower gates: Both inputs are from wires in the range $[n/2, \dots, n-1]$. The reasoning is similar to that for the upper gates, with heat function $\tilde{Q}^{(1,1)}(2 \exp(-n^\alpha/4))$ and expected heat dissipation $O(n^{(1+5\alpha)/2})$.
- Crossover gates: One input wire is in the range $[0, \dots, n/2-1]$ and the other is in the range $[n/2, \dots, n-1]$. The reasoning is similar to the previous two cases, with heat function $\tilde{Q}^{(0,1)}(2 \exp(-n^\alpha/4))$ and expected heat dissipation $O(n^{(1+5\alpha)/2})$.

Summing up over upper, lower and crossover gates, the total heat dissipation per layer in the range 3 to $\log_2 n$ is $O(n^{(1+5\alpha)/2})$, and three different heat functions are used.

Theorem B.1. *With respect to the overwriting circuit model, when given the uniform distribution over inputs in $\{0,1\}^n$, there exist valid heat functions for each gate in Batchier's odd-even mergesort such that the expected total heat release across all gates is $O(n)$.*

Proof. Adding up the expected heat dissipation for layers 1 through $\log_2 n$, we see that the total expected heat dissipation is

$$O(n^{(1+\alpha)/2}) + O(n^{(1+5\alpha)/2} \log_2 n),$$

where the first term accounts for layers 1 and 2, and the second term accounts for gates in layers 3 to $\log_2 n$. The total heat dissipation for the whole circuit is therefore given by the recurrence

$$Q(n) = 2Q(n/2) + O(n^{(1+5\alpha)/2} \log_2 n)$$

and so is $O(n)$ when $\alpha < 1/5$. The number of different heat functions used is $O(1)$ per layer, and so is $O(\log n)$ in total for the network. $\square$

C Re-deriving mismatch costs

In [WK20, YCW26, YYW26] the authors use the language of *mismatch costs* to lower bound the entropy production of implementing gates in a Boolean circuit. Instead of explicitly defining heat functions, they start with the assumption that some distribution over inputs $\mathcal{D}^*$ minimizes the entropy production of the desired computation. Their main observation is that entropy production can be decomposed as the sum of a term characterizing how far the actual input distribution is from $\mathcal{D}^*$ and a term characterizing the inefficiencies even when using $\mathcal{D}^*$ as the input distribution. We start by characterizing how $\mathcal{D}^*$ can be derived from a given heat function.

Lemma C.1. *Let f be an arbitrary function with domain D and $Q : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ be a valid heat function for computing f . Then any conditional input distribution $\mathcal{D}_{|f^{-1}(y)}^*$ that minimizes the entropy production of applying a gate g computing f with heat function Q must satisfy*

$$\mathcal{D}_{|f^{-1}(y)}^* = (\mathcal{G}_Q)_{|f^{-1}(y)}$$

on all y where $(\mathcal{G}_Q)_{|f^{-1}(y)}$ is defined. If $(\mathcal{G}_Q)_{|f^{-1}(y)}$ is undefined, then all conditional input distributions have infinite entropy flow.

Proof. By the definition of entropy production, we have that for every y :

$$\text{EP}_g(\mathcal{D}_{|f^{-1}(y)}) = -\mathcal{S}(\mathcal{D}_{|f^{-1}(y)}) + \mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}}[Q(x)].$$

By Lemma 2.12, we know that $(\mathcal{G}_Q)_{|f^{-1}(y)}$ is the distribution that minimizes this quantity. When $(\mathcal{G}_Q)_{|f^{-1}(y)}$ is undefined, it must be the case that $Q(x) = \infty$ for every $x \in f^{-1}(y)$. In such cases, any distribution over these inputs must release infinite expected heat. $\square$

Corollary C.2. *Let f be an arbitrary function with domain D and $Q : D \rightarrow \mathbb{R}_{\geq 0} \cup \{\infty\}$ be a valid heat function for computing f such that $Q(x) \in \mathbb{R}_{\geq 0}$ for some $x \in D$. Then any input distribution $\mathcal{D}^*$ over D that minimizes the entropy production of applying a gate computing f with heat function Q must have conditional distributions $\mathcal{D}_{|f^{-1}(y)}^* = (\mathcal{G}_Q)_{|f^{-1}(y)}$ and the distribution $f(\mathcal{D}^*)$ must only have support on the y where $(\mathcal{G}_Q)_{|f^{-1}(y)}$ is defined and that minimize $\text{EP}_g((\mathcal{G}_Q)_{|f^{-1}(y)})$.*

Proof. The entropy production on an input distribution $\mathcal{D}$ is given by:

$$\begin{aligned} \text{EP}_g(\mathcal{D}) &= \mathbb{E}_{x \sim \mathcal{D}}[Q(x)] + \mathcal{S}(f(\mathcal{D})) - \mathcal{S}(\mathcal{D}) \\ &= \mathbb{E}_{y \sim f(\mathcal{D})} \left[\log(1/\Pr_{f(\mathcal{D})}[y]) + \mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}} \left[Q(x) - \log(1/\Pr_{\mathcal{D}}[x]) \right] \right] \\ &= \mathbb{E}_{y \sim f(\mathcal{D})} \left[\mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}} [Q(x)] - \mathcal{S}(\mathcal{D}_{|f^{-1}(y)}) \right] \\ &= \mathbb{E}_{y \sim f(\mathcal{D})} [\text{EP}_g(\mathcal{D}_{|f^{-1}(y)})] \end{aligned}$$

The entropy production can be expressed as a convex combination of the entropy production for the different conditional distributions. Lemma C.1 exactly characterizes what these conditional distributions look like. Due to convexity, any optimal distribution must have all of its probability mass on x where $\text{EP}_g(\mathcal{D}_{|f^{-1}(f(x))})$ is minimized. $\square$

We now use heat functions to reprove the decomposition of entropy production into the sum of terms first presented in [WK20]. Intuitively, in Proposition C.3, $\mathcal{D}^*$ represents the distribution on inputs to gate g for which the heat function $\mathcal{G}$ is optimal (i.e., that minimizes entropy production), and $\mathcal{D}$ represents the true distribution, incurring a higher entropy production. The difference is known as the *mismatch cost*. The significance of Proposition C.3 is that this mismatch cost can be characterized purely information theoretically in terms of $\mathcal{D}$ and $\mathcal{D}^*$ (specifically by the KL-divergence terms $D_{\text{KL}}(\mathcal{D}||\mathcal{D}^*) - D_{\text{KL}}(f(\mathcal{D})||f(\mathcal{D}^*))$). In other words the way the physics of the device implementing the gate influences the extra entropy production is entirely contained in its optimal distribution $\mathcal{D}^*$. Our proof of this result uses the language of heat functions, but is similar to that in [WK20].

Proposition C.3 ([WK20]). *Let g be a gate computing function f with domain D , and valid heat function Q . Let $\mathcal{D}$ be any input distribution over D , and $\mathcal{D}^*$ be another input distribution over D where $\text{supp}(f(\mathcal{D})) \subseteq \text{supp}(f(\mathcal{D}^*))$ and $\mathcal{D}_{|f^{-1}(y)}^* = (\mathcal{G}_Q)_{|f^{-1}(y)}$ for every y where both of these distributions are defined. If $(\mathcal{G}_Q)_{|f^{-1}(y)}$ is undefined for some $y \in \text{supp}(f(\mathcal{D}))$ then $\text{EP}_g(\mathcal{D}) = \infty$. Otherwise, the entropy production of applying g on $\mathcal{D}$ is:*

$$\text{EP}_g(\mathcal{D}) = \mathbb{E}_{y \sim f(\mathcal{D})} [\text{EP}_g((\mathcal{G}_Q)_{|f^{-1}(y)})] + D_{\text{KL}}(\mathcal{D}||\mathcal{D}^*) - D_{\text{KL}}(f(\mathcal{D})||f(\mathcal{D}^*))$$

Proof. If $\text{supp}(\mathcal{D}) \not\subseteq \text{supp}(\mathcal{D}^*)$ then there must be some $x \in \text{supp}(\mathcal{D})$ where $Q(x) = \infty$. We would immediately have that $\text{EP}_g(\mathcal{D}) = \infty$ and also $D_{\text{KL}}(\mathcal{D}||\mathcal{D}^*) = \infty$ while the remaining terms on the right-hand side of the equality would be finite. Thus, we can assume without loss of generality that $\text{supp}(\mathcal{D}) \subseteq \text{supp}(\mathcal{D}^*)$.

$$\text{EP}_g(\mathcal{D}) = \mathcal{S}(f(\mathcal{D})) - \mathcal{S}(\mathcal{D}) + \mathbb{E}_{x \sim \mathcal{D}}[Q(x)] \tag{16}$$

$$\begin{aligned}
&= \mathbb{E}_{y \sim f(\mathcal{D})} \left[\mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}} [\log(\Pr_{\mathcal{D}}[x] / \Pr_{f(\mathcal{D})}[y]) + Q(x)] \right] \\
&= \mathbb{E}_{y \sim f(\mathcal{D})} \left[\mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}} [\log(\Pr_{\mathcal{D}_{|f^{-1}(y)}}[x]) + Q(x)] \right] \\
&= \mathbb{E}_{y \sim f(\mathcal{D})} \left[\mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}} [\log(2^{Q(x)} \Pr_{\mathcal{D}_{|f^{-1}(y)}}[x])] \right].
\end{aligned}$$

Since $\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)}) = \mathbb{E}_{x \sim (\mathcal{G}_Q)_{|f^{-1}(y)}} [\log(2^{Q(x)} \Pr_{(\mathcal{G}_Q)_{|f^{-1}(y)}}[x])]$ we can rewrite Eq. (16) as:

$$\begin{aligned}
&\mathbb{E}_{y \sim f(\mathcal{D})} \left[\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)}) + \mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}} [\log(2^{Q(x)} \Pr_{\mathcal{D}_{|f^{-1}(y)}}[x])] - \mathbb{E}_{x \sim (\mathcal{G}_Q)_{|f^{-1}(y)}} [\log(2^{Q(x)} \Pr_{(\mathcal{G}_Q)_{|f^{-1}(y)}}[x])] \right] \\
&= \mathbb{E}_{y \sim f(\mathcal{D})} \left[\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)}) \right] \\
&\quad + \mathbb{E}_{y \sim f(\mathcal{D})} \left[\mathbb{E}_{x \sim \mathcal{D}_{|f^{-1}(y)}} [\log(2^{Q(x)} \Pr_{\mathcal{D}_{|f^{-1}(y)}}[x])] - \mathbb{E}_{x \sim (\mathcal{G}_Q)_{|f^{-1}(y)}} [\log(2^{Q(x)} \Pr_{(\mathcal{G}_Q)_{|f^{-1}(y)}}[x])] \right].
\end{aligned} \tag{17}$$

Now by the definition of $(\mathcal{G}_Q)_{|f^{-1}(y)}$, for every $x \in \text{supp}((\mathcal{G}_Q)_{|f^{-1}(y)})$, $2^{Q(x)} \Pr_{(\mathcal{G}_Q)_{|f^{-1}(y)}}[x] = 1 / \sum_{x' \in f^{-1}(y)} 2^{-Q(x')}$. This implies that this quantity has the same value for every element of the set $f^{-1}(y)$. Therefore, any convex combination over the terms with the same values of y will give the same value. This lets us rewrite Eq. (17) as:

$$\begin{aligned}
&= \mathbb{E}_{y \sim f(\mathcal{D})} [\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)})] + \mathbb{E}_{x \sim \mathcal{D}} [\log(2^{Q(x)} \Pr_{\mathcal{D}_{|f^{-1}(f(x))}}[x]) - \log(2^{Q(x)} \Pr_{(\mathcal{G}_Q)_{|f^{-1}(f(x))}}[x])] \\
&= \mathbb{E}_{y \sim f(\mathcal{D})} [\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)})] + \mathbb{E}_{x \sim \mathcal{D}} [\log(\Pr_{\mathcal{D}_{|f^{-1}(f(x))}}[x] / \Pr_{(\mathcal{G}_Q)_{|f^{-1}(f(x))}}[x])] \\
&= \mathbb{E}_{y \sim f(\mathcal{D})} [\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)})] + \mathbb{E}_{x \sim \mathcal{D}} \left[\log \left(\frac{\Pr_{\mathcal{D}}[x] \Pr_{f(\mathcal{D}^*)}[f(x)]}{\Pr_{f(\mathcal{D})}[f(x)] \Pr_{\mathcal{D}^*}[x]} \right) \right] \\
&= \mathbb{E}_{y \sim f(\mathcal{D})} [\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)})] + D_{\text{KL}}(\mathcal{D} || \mathcal{D}^*) - D_{\text{KL}}(f(\mathcal{D}) || f(\mathcal{D}^*))
\end{aligned}$$

as desired. $\square$

There are actually some subtlety in our statement of Proposition C.3 that becomes obvious with the language of heat functions. In particular, we cannot always take $\mathcal{D}^*$ to be the distribution over inputs that minimizes the entropy production. Due to Corollary C.2, we know that such a $\mathcal{D}^*$ would only have support on y that minimize $\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)})$, and thus for many choices of $\mathcal{D}$ we will not have that $\text{supp}(f(\mathcal{D})) \subseteq \text{supp}(f(\mathcal{D}^*))$. Proposition C.3 actually has a nice intuitive interpretation in terms of heat functions. The term $D_{\text{KL}}(\mathcal{D} || \mathcal{D}^*) - D_{\text{KL}}(f(\mathcal{D}) || f(\mathcal{D}^*))$ corresponds to how well suited Q is for the input distribution $\mathcal{D}$ while $\mathbb{E}_{y \sim f(\mathcal{D})} [\mathbb{E} \Pr_g((\mathcal{G}_Q)_{|f^{-1}(y)})]$ corresponds to how close Q is to being optimal on the best possible distributions over the preimage of each y .

D Entropy production from modularity

Heat functions only consider the inputs to a specific gate to determine the corresponding heat release. Thus, they do not account for correlations between the inputs to this gate and the rest

of the circuit. This leads to a source of entropy production due to the locality of operations that are not injective. This source of entropy production was first identified in [WK20], where it was shown that the entropy production of applying a gate in a circuit is at least the drop in mutual information between the wires in this gate and the state of the remaining wires of the circuit. We prove essentially the same result, showing that it is an immediate consequence of the locality of valid heat functions.

Proposition D.1 ([WK20]). *Let $\mathcal{C}$ be a thermodynamic circuit with W wires, and let X, Y be a partition of $[W]$ and $\mathcal{C}_{t-1}[X], \mathcal{C}_{t-1}[Y] \sim \mathcal{D}_{t-1}$ be random variables for the state of the wires in X, Y after applying gate $t-1$ in $\mathcal{C}$. Let $\mathcal{D}_X$ be the marginal distribution of random variable $\mathcal{C}_{t-1}[X]$ according to distribution $\mathcal{D}_{t-1}$. Then if gate g_t computes the function $f^{g_t} : \Sigma^X \rightarrow \Sigma^X$ and has valid heat function Q_{g_t} , the entropy production of applying g_t as the next gate in $\mathcal{C}$ is $\text{EP}_{g_t}^{\text{Loc}}(\mathcal{D}_X) + \mathcal{I}(\mathcal{C}_{t-1}[X]; \mathcal{C}_t[Y]) - \mathcal{I}(\mathcal{C}_t[X]; \mathcal{C}_t[Y])$.*

Proof. By Definition 2.25,

$$\mathbb{E}_{x \sim \mathcal{D}_X}[Q_{g_t}(x)] = \text{EP}_{g_t}^{\text{Loc}}(\mathcal{D}_X) + \mathcal{S}(\mathcal{C}_{t-1}[X]) - \mathcal{S}(\mathcal{C}_t[X])$$

Let $\mathcal{D}_t$ be the distribution over states of the circuit after applying g_t . Since $\mathcal{C}_{t-1}[Y]$ is the same as $\mathcal{C}_t[Y]$, we know that according to Definition 2.24:

$$\begin{aligned} \text{EP}_{g_t}(\mathcal{D}_{t-1}) &= \mathbb{E}_{x \sim \mathcal{D}_{t-1}}[Q_{g_t}(x_X)] + \mathcal{S}(\mathcal{D}_t) - \mathcal{S}(\mathcal{D}_{t-1}) \\ &= \mathbb{E}_{x \sim \mathcal{D}_X}[Q_{g_t}(x)] + \mathcal{S}(\mathcal{D}_t) - \mathcal{S}(\mathcal{D}_{t-1}) \\ &= \text{EP}_{g_t}^{\text{Loc}}(\mathcal{D}_X) + \mathcal{S}(\mathcal{C}_{t-1}[X]) - \mathcal{S}(\mathcal{C}_t[X]) + \mathcal{S}(\mathcal{D}_t) - \mathcal{S}(\mathcal{D}_{t-1}) \\ &= \text{EP}_{g_t}^{\text{Loc}}(\mathcal{D}_X) + \mathcal{S}(\mathcal{C}_{t-1}[X]) - \mathcal{S}(\mathcal{C}_t[X]) + \mathcal{S}(\mathcal{C}_t[X], \mathcal{C}_t[Y]) - \mathcal{S}(\mathcal{C}_{t-1}[X], \mathcal{C}_t[Y]) \\ &= \text{EP}_{g_t}^{\text{Loc}}(\mathcal{D}_X) + \mathcal{S}(\mathcal{C}_{t-1}[X]) - \mathcal{S}(\mathcal{C}_t[X]) + \mathcal{S}(\mathcal{C}_t[X] \mid \mathcal{C}_t[Y]) - \mathcal{S}(\mathcal{C}_{t-1}[X] \mid \mathcal{C}_t[Y]) \\ &= \text{EP}_{g_t}^{\text{Loc}}(\mathcal{D}_X) + \mathcal{I}(\mathcal{C}_{t-1}[X]; \mathcal{C}_t[Y]) - \mathcal{I}(\mathcal{C}_t[X]; \mathcal{C}_t[Y]) \end{aligned}$$

as desired. □

Thus we can decompose entropy production into a term based off of sub-optimal computation on the marginal input distribution plus the drop in mutual information caused by the locality constraint on heat functions.