

What Can We Infer From Hashes?

Economics of Information: Part 2B

Steph Macurdy
Institute of Thermoeconomics

October 2026

Abstract

What can we infer about the hidden activity of a distributed computational process from the constraints imposed and the statistical properties of its observed outputs? This paper develops the relationship between a known proof of work success probability and otherwise unobserved computation. Beginning with individual hash attempts, we connect proof counts and waiting times to an estimate of interval average hashes and its sampling uncertainty. The observations do not reveal the exact number of failed attempts or reconstruct an arbitrary computational history. Combined with an independently specified energy cost per hash, they support a conditional estimate of physical energy expenditure.

1 Introduction

Proof of work is the engineering application of interest.

Proof of work uses cryptographic hashes whose outputs, for distinct mining inputs, are modeled as independent and uniformly distributed over the range of possibilities. Under this model, no hash attempt has a greater probability of success than any other. The protocol imposes a difficulty threshold that distinguishes valid proofs from invalid proofs and determines the probability that any individual attempt will succeed.

While the majority of computation behind each proof goes “unseen,” the observed proofs provide statistical evidence about the variable of interest—total hash. Those unsuccessful attempts constitute the search effort that makes producing valid proofs costly. This search cost is central to the security rationale of proof of work [10], although a valid proof does not reveal how many unsuccessful attempts preceded it.

This gives rise to the central question of our inquiry:

Given a known probability of success and an observed sequence of valid proofs, what can we infer about the otherwise hidden computational process that produced them?

The answer requires distinguishing what is known and observed from what must be inferred. The difficulty threshold establishes the probability of success. The number and timing of valid proofs

provide observable evidence. Together, these measurements allow us to estimate the underlying quantity of computation, even though the individual hash attempts remain unobserved.

Although probability alone cannot determine energy expenditure, an inferred total hash quantity combined with an estimate of energy per hash enables us to estimate the physical expenditure of the process. The observed proofs do not supply that efficiency information.

1.1 Scope and Observation Assumptions

Unless stated otherwise, difficulty remains fixed throughout the observation interval. We assume independent trials on distinct mining inputs, complete observation of successful proofs in the modeled process, and an accurately measured observation duration. Where arrival times are used, they are assumed to represent the timing of proof discovery sufficiently well for the model.

A canonical ledger need not contain every successful proof—competing valid blocks and withheld discoveries may be absent. Miner provided block timestamps also need not equal actual discovery times [10, 12]. Applying the model to ledger data therefore requires accounting for these differences or treating them as explicit approximations. Without those assumptions, the estimate cannot automatically be identified with total network hashes.

1.2 This Paper’s Objective

Our objective is to derive these relationships, beginning with the probability of a single hash attempt and proceeding toward inference about an unobserved physical process. Five probability distributions describe successive aspects of that process:

$$\text{Difficulty} \rightarrow \text{Bernoulli} \rightarrow \begin{cases} \text{Binomial} \\ \text{Geometric} \end{cases} \rightarrow \begin{cases} \text{Poisson} \\ \text{Exponential} \end{cases} \rightarrow \text{Inferring Hash} \rightarrow \text{Physical energy}$$

The binomial and geometric distributions describe proof counts and search effort in hash attempts. The Poisson and exponential distributions then describe proof counts and waiting times in physical time.

2 Difficulty: The Probability of a Proof

Consider a cryptographic hash function with a 256-bit output. Although the hash function itself is deterministic, we model the outputs generated from distinct mining inputs as independent and uniformly distributed over the output space

$$\Omega = \{0, 1\}^{256}, \quad |\Omega| = 2^{256}.$$

Suppose the proof of work rule declares an output valid when its first d bits are zero. This requirement defines the simplified difficulty condition used throughout the paper. Of the 2^{256} possible outputs, exactly 2^{256-d} satisfy this condition. Therefore,

$$p = \frac{2^{256-d}}{2^{256}} = 2^{-d}.$$

Thus, d is the number of constrained output bits, and each additional constrained bit halves the probability of success.

For example,

$$d = 4 \implies p = \frac{1}{16} = 0.0625, \quad d = 40 \implies p \approx 9.095 \times 10^{-13}.$$

Difficulty fixes the probability of success for each attempt; it does not prescribe a deterministic amount of computation. A valid proof may be found on the first attempt, while another search may require far more attempts than the expected number. Difficulty adjustment is treated separately in Appendix B.

3 Bernoulli: One Hash Trial

We begin with one hash. Define the random variable

$$X_i = \begin{cases} 1 & \text{if hash } i \text{ is valid,} \\ 0 & \text{otherwise.} \end{cases}$$

Then $X_i \sim \text{Bernoulli}(p)$, with probability mass function

$$\mathbb{P}(X_i = x) = p^x(1-p)^{1-x}, \quad x \in \{0, 1\}.$$

Its expectation, variance, and standard deviation are

$$\mathbb{E}[X_i] = p, \quad \text{Var}(X_i) = p(1-p), \quad \sigma_X = \sqrt{p(1-p)}.$$

For $p = 1/16$, each attempt produces an average of $1/16$ valid proofs. This does not mean a fractional proof is ever produced. A single hash produces either zero or one valid proof; the expectation describes an average over repeated experiments.

4 Binomial: Count Proofs

Now suppose we perform exactly n independent hashes. How many valid proofs should we expect? Define

$$K = \sum_{i=1}^n X_i.$$

Because K is the sum of independent, identically distributed Bernoulli variables,

$$K \sim \text{Binomial}(n, p), \quad \mathbb{P}(K = k) = \binom{n}{k} p^k (1-p)^{n-k}, \quad k = 0, \dots, n.$$

The binomial coefficient counts the different arrangements of k successes among n attempts. By linearity of expectation and independence,

$$\mathbb{E}[K] = np, \quad \text{Var}(K) = np(1-p), \quad \sigma_K = \sqrt{np(1-p)}.$$

Suppose $n = 16$ and $p = 1/16$. The expected number of proofs is one, but one expected proof does not mean exactly one proof:

$$\mathbb{P}(K = 0) = \left(\frac{15}{16}\right)^{16} \approx 0.356, \quad \mathbb{P}(K \geq 1) = 1 - \left(\frac{15}{16}\right)^{16} \approx 0.644.$$

Thus there is about a 35.6% chance of observing no valid proof after 16 hashes.

The standard deviation is approximately 0.968 proofs, nearly as large as the expected count. More generally, for $np > 0$,

$$\frac{\sigma_K}{\mathbb{E}[K]} = \sqrt{\frac{1-p}{np}}.$$

Precision therefore depends on the expected number of successful proofs, not merely on a large number of hash attempts.

5 Geometric: Wait for a Proof

The binomial distribution counts valid proofs in a fixed number of hashes. The geometric distribution reverses the question—how many hashes are required to find the first valid proof?

Let N be the number of hash attempts, including the successful attempt. Since each attempt succeeds independently with probability p ,

$$N \sim \text{Geometric}(p), \quad \mathbb{P}(N = n) = (1-p)^{n-1}p, \quad n = 1, 2, \dots$$

The factor $(1-p)^{n-1}$ represents the preceding failures, and the final factor p represents success on attempt n . Equivalently,

$$\mathbb{P}(N > n) = (1-p)^n, \quad \mathbb{P}(N \leq n) = 1 - (1-p)^n.$$

The expectation, variance, and standard deviation are

$$\mathbb{E}[N] = \frac{1}{p}, \quad \text{Var}(N) = \frac{1-p}{p^2}, \quad \sigma_N = \frac{\sqrt{1-p}}{p}.$$

Thus an expected wait of $1/p$ attempts is an average, not a deadline. For rare events, $1-p \approx 1$, so $\sigma_N \approx \mathbb{E}[N]$: the number of attempts required for an individual proof remains highly variable.

With $p = 2^{-d}$, the expected number of hashes is 2^d . At $d = 40$, both the expectation and standard deviation are approximately 1.10×10^{12} hashes. Adding one difficulty bit halves the success probability and doubles the expected search effort.

6 Poisson Distribution: Proofs in Time

We now move from counting discrete hash attempts to observing successful proofs over physical time. The Poisson distribution describes the discrete number of events observed during a fixed interval. A Poisson process extends this description to event counts indexed by continuous time.

Throughout this section, difficulty is assumed constant and publicly known. Hashrate is unknown and may vary throughout the observation interval.

6.1 From Binomial Trials to Poisson Arrivals

Suppose the network performs an enormous number of independent hash attempts, while the probability of success per attempt is extremely small. In the rare-event limit,

$$n \longrightarrow \infty, \quad p \longrightarrow 0, \quad np \longrightarrow \mu,$$

the binomial distribution converges to the Poisson distribution:

$$\mathbb{P}(K = k) = \frac{\mu^k e^{-\mu}}{k!}, \quad k = 0, 1, 2, \dots$$

Here, μ is the expected number of successful proofs, and k is the number actually observed.

This convergence concerns event counts. To model arrival times, we must additionally describe how hash attempts occur over time. The limiting derivation is given in Appendix A.

6.2 Cumulative Computational Activity

Let $H(s)$ represent an underlying attempt-rate trajectory, measured in hashes per second. We treat this as an unknown deterministic trajectory. In a continuous approximation, cumulative computational activity over $[0, t]$ is

$$N(t) = \int_0^t H(s) ds.$$

The observer does not know $N(t)$ because unsuccessful attempts are not publicly visible.

Define the instantaneous proof-arrival intensity $\lambda(s) = pH(s)$ and cumulative intensity

$$\Lambda(t) = \int_0^t \lambda(s) ds = p \int_0^t H(s) ds = pN(t).$$

The cumulative intensity represents the expected number of valid proofs associated with a particular computational trajectory. It does not require constant hashrate.

Under an independent-trial, rare-event approximation, with attempts distributed sufficiently regularly over time, proof arrivals can be modeled by a nonhomogeneous Poisson process:

$$K(t) \mid H(\cdot), d \sim \text{Poisson}\left(p \int_0^t H(s) ds\right).$$

Equivalently,

$$\mathbb{P}(K(t) = k \mid \Lambda(t)) = \frac{\Lambda(t)^k e^{-\Lambda(t)}}{k!}.$$

This describes the randomness of proof discovery conditional on a possible computational trajectory. It does not imply that the observer knows that trajectory.

6.3 Expectation, Variance, and an Example

For a specified cumulative intensity,

$$\mathbb{E}[K(t) \mid \Lambda(t)] = \text{Var}(K(t) \mid \Lambda(t)) = \Lambda(t).$$

The conditional standard deviation is $\sqrt{\Lambda(t)}$, and for positive cumulative intensity,

$$\frac{\sigma_{K|\Lambda}}{\mathbb{E}[K(t) | \Lambda(t)]} = \frac{1}{\sqrt{\Lambda(t)}}.$$

For expected counts of one, ten, and 100, the corresponding relative standard deviations are 100%, approximately 31.6%, and 10%.

For illustration, suppose $\Lambda(5) = 1$. This means one expected proof during a five-second interval, without requiring constant hashrate. Then

$$\mathbb{P}(K(5) = 0 | \Lambda(5) = 1) = e^{-1} \approx 36.79\%,$$

and

$$\mathbb{P}(K(5) = 2 | \Lambda(5) = 1) = \frac{e^{-1}}{2} \approx 18.39\%.$$

These probabilities are conditional on the illustrative intensity; they are not unconditional predictions about an unknown network.

6.4 From Proof Counts to Waiting Times

Let T denote the waiting time until the first proof, measured from time zero. Conditional on an intensity trajectory, waiting longer than t is equivalent to observing no proof before t :

$$\mathbb{P}(T > t | \lambda(\cdot)) = \mathbb{P}(K(t) = 0 | \lambda(\cdot)) = e^{-\Lambda(t)}.$$

For a continuous intensity function, the corresponding waiting-time density is

$$f_T(t | \lambda(\cdot)) = \lambda(t)e^{-\Lambda(t)}.$$

If hashrate is constant, $\lambda = Hp$ is constant and

$$T \sim \text{Exponential}(\lambda), \quad \mathbb{P}(T > t) = e^{-\lambda t}, \quad f_T(t) = \lambda e^{-\lambda t}, \quad t \geq 0.$$

The expected waiting time and its standard deviation are both $1/\lambda$, while the variance is $1/\lambda^2$. For a rate of one proof every five seconds on average, the probability of a proof within five seconds is

$$\mathbb{P}(T \leq 5) = 1 - e^{-1} \approx 63.21\%.$$

A single short or long interval does not by itself establish a change in hashrate.

Conditional on a fixed rate value, the exponential distribution is memoryless:

$$\mathbb{P}(T > s + t | T > s) = \mathbb{P}(T > t).$$

A failed hash does not make the next hash more likely to succeed. There is, however, an important distinction between the physical process and our knowledge of that process. If the rate is unknown, observing an unusually long delay may provide evidence that it is lower than previously estimated. Previous waiting times do not change the success probability of an individual hash, but they can change an observer's inference about the hidden hashrate.

7 Inference from Observed Proofs

Difficulty supplies the known probability p , and observed proofs supply the evidence. The underlying interval-average hashrate is

$$\overline{H} = \frac{1}{t} \int_0^t H(s) ds, \quad \Lambda(t) = pt\overline{H}.$$

7.1 Inferring Hidden Computation from Proof Counts

Suppose k proofs are observed during a fixed interval of duration $t > 0$. The count likelihood for cumulative intensity $\Lambda = \Lambda(t)$ is

$$L(\Lambda \mid k) \propto \Lambda^k e^{-\Lambda}.$$

For $k > 0$, taking the logarithm and differentiating gives

$$\ell(\Lambda) = k \ln \Lambda - \Lambda + C, \quad \frac{\partial \ell}{\partial \Lambda} = \frac{k}{\Lambda} - 1.$$

Setting the derivative to zero gives $\hat{\Lambda} = k$. For $k = 0$, the likelihood is maximized at the boundary $\hat{\Lambda} = 0$. Therefore,

$$\boxed{\hat{N} = \frac{k}{p} = k2^d, \quad \hat{\overline{H}} = \frac{k}{pt} = \frac{k2^d}{t}.$$

Estimating this average does not require assuming that instantaneous hashrate remained constant.

The estimate is nevertheless subject to sampling uncertainty. In particular, observing zero proofs does not establish that no hashes were performed: for every finite positive Λ ,

$$\mathbb{P}(K = 0 \mid \Lambda) = e^{-\Lambda} > 0.$$

For example, ten proofs over 50 seconds at $d = 40$ give

$$\hat{\overline{H}} = \frac{10(2^{40})}{50} \approx 2.20 \times 10^{11} \text{ hashes/s} = 220 \text{ GH/s}.$$

The estimate concerns an aggregate rate; it does not reconstruct the exact number or sequence of failed hashes.

7.2 Estimator Precision and Confidence Bounds

Under the Poisson count model,

$$\text{Var}(\hat{\overline{H}}) = \frac{\text{Var}(K)}{(pt)^2} = \frac{\overline{H}}{pt}, \quad \widehat{\text{SE}}(\hat{\overline{H}}) = \frac{\sqrt{k}}{pt}.$$

If $\mu = pt\overline{H}$ is the expected proof count and $\overline{H} > 0$, the relative standard error is

$$\frac{\widehat{\text{SE}}(\hat{\overline{H}})}{\overline{H}} = \frac{1}{\sqrt{\mu}}.$$

Replacing μ with k , 100 observed proofs give about 10% estimated relative standard error, and 10,000 give about 1%. Halving the relative standard error requires four times as many expected proofs. These describe sampling uncertainty under the model; they are not 95% error bounds or allowances for errors in the observation assumptions.

For small counts, confidence intervals are more informative than the plug-in standard error. Their endpoints can be obtained directly from Poisson probabilities. For the ten-proof example, the lower endpoint is the expected count at which observing ten or more proofs has probability 2.5%; the upper endpoint is the expected count at which observing ten or fewer has probability 2.5%. The 95% confidence level is a conventional choice, not a property of proof of work.

These endpoints are approximately 4.80 and 18.39 expected proofs. Dividing by pt gives the confidence interval

$$105 \text{ GH/s} \leq \bar{H} \leq 404 \text{ GH/s}.$$

The wide interval makes the uncertainty behind the 220 GH/s point estimate explicit. Across repeated observations under the model, this procedure produces intervals containing the true interval-average hashrate at least 95% of the time. Appendix C gives the general calculation.

When $k = 0$, the plug-in standard error is zero and should not be used as an uncertainty bound. An absence of proofs nevertheless supplies evidence, because

$$\mathbb{P}(K = 0 \mid \bar{H}) = e^{-pt\bar{H}}.$$

Setting this probability to 0.05 gives a one-sided 95% upper confidence limit:

$$\bar{H}_{\text{upper}} = \frac{-\ln(0.05)}{pt} \approx \frac{3}{pt}.$$

Higher rates would produce an observation window with no proofs less than 5% of the time. These are statistical confidence bounds conditional on the model, not deterministic guarantees of realized computation or energy expenditure.

7.3 What Proof Counts Cannot Reveal

There is a fundamental limit to what can be inferred from proof counts alone. Two different hashrate trajectories may satisfy

$$\int_0^t H_1(s) ds = \int_0^t H_2(s) ds.$$

They therefore have the same cumulative intensity and the same conditional distribution of total proof counts over that interval.

Consequently, a single proof count cannot uniquely identify how computational activity was distributed across time. The observer can estimate cumulative computation, but cannot reconstruct its precise temporal history from the count alone. Individual proof timestamps can provide additional information about the temporal structure of a varying-rate process. However, inference about an arbitrary time-varying hashrate still requires statistical assumptions or constraints.

The distinction is between the forward process

$$p, H(\cdot) \longrightarrow \Lambda(t) \longrightarrow K(t)$$

and the inverse problem

$$p, K(t) \longrightarrow L(\Lambda \mid K(t)) \longrightarrow \hat{\Lambda} \longrightarrow \hat{H}.$$

The first describes proof production; the second describes what an observer can infer from it.

8 From Computation to Physical Energy

The preceding sections connect a known success probability to an estimate of otherwise unobserved computation. We now distinguish three further quantities: the statistical rarity of a valid proof, the electrical energy consumed by a particular implementation, and the thermodynamic consequences of specified physical operations within that implementation. These quantities can be related only after the relevant physical assumptions and system boundary have been stated.

Let

$$\epsilon_h = \text{electrical joules consumed per hash.}$$

Initially assume that ϵ_h is approximately constant and applies to every attempted hash. For a search stopped at its first valid proof, let N_{proof} denote the number of attempted hashes, including the successful attempt. Since

$$N_{\text{proof}} \sim \text{Geometric}(p),$$

the realized electrical energy is the random variable

$$E_{\text{proof}} = \epsilon_h N_{\text{proof}}.$$

Consequently,

$$\mathbb{E}[E_{\text{proof}}] = \frac{\epsilon_h}{p}, \quad \text{SD}(E_{\text{proof}}) = \epsilon_h \frac{\sqrt{1-p}}{p}.$$

With $p = 2^{-d}$,

$$\mathbb{E}[E_{\text{proof}}] = \epsilon_h 2^d.$$

The coefficient of variation is $\sqrt{1-p}$, which is close to one for a rare success. Thus the expected energy per proof is an average over repeated searches, not a prediction of the energy consumed in producing an individual proof.

Using the surprisal of a successful trial,

$$I_{\text{success}} = -\ln p = d \ln 2,$$

measured in nats, the expected engineering cost may be written

$$\mathbb{E}[E_{\text{proof}}] = \epsilon_h e^{I_{\text{success}}}.$$

The exponential appears because $e^{I_{\text{success}}} = 1/p$ is the expected number of trials. It is not a thermodynamic conversion factor; the physical energy scale is supplied by ϵ_h .

Several information quantities must also be distinguished. Let Y be an idealized uniform w -bit trial output, let A be the valid target set, and let $B = \mathbf{1}\{Y \in A\}$, with $p = |A|/2^w$. Then

$$\begin{aligned} H_2(Y) &= w, \\ I(B=1) &= -\log_2 p = d, \\ H_2(B) &= h_2(p), \\ H_2(Y \mid B=1) &= w - d, \\ H_2(Y \mid B=0) &= w + \log_2(1-p), \end{aligned}$$

where h_2 is the binary entropy function. These information quantities appear because the rarity of a valid proof, the uncertainty of a hash output, and the entropy removed by a physical reset are distinct concepts.

These distinctions prevent difficulty bits from being interpreted automatically as physically erased bits. Their purpose is to identify the additional step required for an energetic conclusion: one must specify what information a physical implementation stores and resets. The relevant entropy reduction can then be represented by s_f and, under an explicit reset model, related to a conditional Landauer heat bound.

8.1 Inferred Energy Expenditure

Suppose k valid proofs are observed during an interval of duration t at fixed difficulty. The proof-count likelihood gives

$$\hat{N}_{\text{total}} = \frac{k}{p} = k2^d, \quad \hat{H} = \frac{k}{pt} = \frac{k2^d}{t}.$$

Under the constant efficiency model, the corresponding estimate of electrical energy is

$$\hat{E}_{\text{total}} = \epsilon_h \hat{N}_{\text{total}} = \epsilon_h k2^d.$$

This remains a statistical point estimate. It does not assert that exactly $k2^d$ hashes occurred. If $[H_L, H_U]$ is a confidence interval for interval-average hashrate and ϵ_h is treated as known, the associated interval for hashing energy is

$$[\epsilon_h t H_L, \epsilon_h t H_U].$$

Uncertainty in ϵ_h is additional to the sampling uncertainty represented by this interval. For a time-varying process, let $H(s)$ be the attempt rate and let $\epsilon_h(s)$ be the hash-weighted electrical cost per attempt. The active-hashing expenditure is

$$E_{\text{hashing}} = \int_0^t \epsilon_h(s) H(s) ds.$$

The simple product $\epsilon_h k2^d$ therefore presumes either constant efficiency or an independently justified hash-weighted average efficiency. Cooling, power conversion, communication, idle equipment, and other auxiliary expenditure must be included separately when they lie outside the definition of ϵ_h .

8.2 From Shannon Entropy to Physical Entropy

Shannon and Gibbs entropy share the same mathematical functional, but their physical identification requires that the probability distribution describe physical states [3, 1]. For a distribution ρ_a over physical microstates a ,

$$S_G(\rho) = -k_B \sum_a \rho_a \ln \rho_a.$$

This is k_B times the Shannon entropy of the *same* distribution. It does not identify the Shannon entropy of an arbitrary message or proof event with the thermodynamic entropy of the device representing it.

To make the distinction explicit, partition a memory's physical microstates into disjoint sets Ω_i , one for each logical value i . Define

$$p_i = \sum_{a \in \Omega_i} \rho_a, \quad r_{a|i} = \frac{\rho_a}{p_i}, \quad S_i = -k_B \sum_{a \in \Omega_i} r_{a|i} \ln r_{a|i}.$$

The physical entropy decomposes as [16]

$$S_G(\rho) = k_B H(p) + \sum_i p_i S_i.$$

The first term is uncertainty about the logical value; the second is physical entropy within the logical states. Only when the internal contribution is controlled appropriately does a change in logical entropy determine a change in physical entropy.

For a register coupled to a reservoir at temperature T_{phys} , let Q_{bath} be positive for heat transferred to the reservoir. Under the usual assumptions concerning the initial register–reservoir state and the treatment of auxiliary entropy sinks, the entropy balance gives

$$\Delta S_G + \frac{\mathbb{E}[Q_{\text{bath}}]}{T_{\text{phys}}} \geq 0.$$

An exact reset that reduces the physical entropy of the specified memory by $k_B \ln 2 s$ therefore satisfies the conditional mean heat bound

$$\mathbb{E}[Q_{\text{bath}}] \geq k_B T_{\text{phys}} \ln 2 s$$

[14, 17, 18].

The scope of the reset operation is essential. If h is a deterministic hash and the input X is retained, then

$$H_2(h(X) | X) = 0.$$

A reversible implementation may use retained information to uncompute intermediate or output registers without inverting the hash [15]. Statistical unpredictability to an observer therefore does not, by itself, establish a particular entropy reduction or heat cost inside the computing device.

8.3 A Conditional Failed-Trial Reset Model

The engineering coefficient ϵ_h describes electrical energy per attempted hash. A different question is whether a specified operation associated with each failed trial has a thermodynamic lower bound. To examine that question, let

$$F = N_{\text{proof}} - 1$$

be the number of failed trials before the first success. Then

$$\mathbb{E}[F] = \frac{1-p}{p}.$$

Assume that each failed trial is followed by a specified reset operation at temperature T_{phys} . Suppose that, after accounting for internal physical entropy and correlated side information, this operation

reduces the entropy of the specified memory by at least $k_B \ln 2 s_f$, where $s_f \geq 0$ is measured in bits. The conditional mean heat per failed-trial reset is then bounded by

$$q_0 = k_B T_{\text{phys}} \ln 2 s_f.$$

Under the independent-trial model, the mean heat associated with failed-trial resets before a successful proof satisfies

$$\mathbb{E}[Q_{\text{failed}}] \geq k_B T_{\text{phys}} \ln 2 s_f \frac{1-p}{p}.$$

Using $p = 2^{-d}$,

$$\mathbb{E}[Q_{\text{failed}}] \geq k_B T_{\text{phys}} \ln 2 s_f (1 - 2^{-d}) 2^d.$$

The factor 2^d arises from the expected multiplicity of the search. The factor s_f arises from the physical entropy reduction of the specified reset operation. These factors have different origins: the result does not say that each hash erases d bits.

For illustration, suppose a failed w -bit output is physically distributed uniformly over the complement of the valid target set and is reset locally without using correlated input information. The logical entropy of that failed output is

$$s_f = H_2(Y \mid B = 0) = \log_2(2^w - |A|) = w + \log_2(1 - p).$$

The conditional bound becomes

$$\mathbb{E}[Q_{\text{failed}}] \geq k_B T_{\text{phys}} \ln 2 [w + \log_2(1 - p)] \frac{1-p}{p}.$$

For $p \ll 1$, its leading behavior is

$$\mathbb{E}[Q_{\text{failed}}] \gtrsim w k_B T_{\text{phys}} \ln 2 2^d.$$

This full-output reset model is an additional implementation assumption. If the output is not materialized, if retained information is used to uncompute it reversibly, or if the internal physical entropy changes differently, then the appropriate value of s_f must be derived from that implementation.

For k proofs observed during a fixed interval, the proof-count estimate implies

$$\hat{F} = \hat{N}_{\text{total}} - k = k \frac{1-p}{p}.$$

Thus a plug-in estimate of the conditional mean heat floor in this reset model is

$$\hat{Q}_{\text{failed,min}} = k_B T_{\text{phys}} \ln 2 s_f k \frac{1-p}{p}.$$

Because the failed attempts are unobserved, this is a model-based inferred mean contribution, not a deterministic lower bound on the heat realized in the observed interval.

8.4 From Conditional Heat to Power

A sustained proof rate converts the conditional mean heat per proof into a conditional average power requirement. Let R be the long-run aggregate trial rate. Under the independent repeated-trial model, the proof rate is

$$\lambda = R p.$$

If each failed trial undergoes the specified reset, the associated mean heat rate satisfies

$$P_{\text{failed}} \geq R(1-p)q_0 = \lambda k_B T_{\text{phys}} \ln 2 s_f \frac{1-p}{p}.$$

If useful operation requires a sustained mean proof rate of at least one proof per τ seconds, then

$$P_{\text{failed}} \geq \frac{k_B T_{\text{phys}} \ln 2 s_f}{\tau} \frac{1-p}{p}.$$

The interval τ is an aggregate mean interval, not a deadline by which every proof must arrive. A success probability of at least α by a deadline instead requires capacity for at least

$$q_{\min} = \left\lceil \frac{\ln(1-\alpha)}{\ln(1-p)} \right\rceil$$

trials during the window. This capacity is not the number necessarily executed by a search that stops at success.

The two energy statements in this section should not be conflated. The quantity

$$\hat{E}_{\text{total}} = \epsilon_h \frac{k}{p}$$

is an engineering estimate based on a measured or modeled electrical cost per hash. The Landauer expression is a conditional lower bound on one specified heat contribution under an explicit reset model. If ϵ_h already includes that reset expenditure, adding the two quantities would double count it. Neither difficulty nor observed proofs alone reveal the implementation's efficiency, reset architecture, or total physical expenditure.

9 Conclusion

Proof of work connects a known probability law to an expected amount of search and, under an explicit physical implementation, to an estimate of energy expenditure. The observed number and timing of proofs provide statistical evidence about hidden computation; they do not reveal the exact number of failed attempts or the total energy expended.

A successful proof converts an uncertain computational event into evidence that can be checked independently and shared across a global network. Its validity is publicly verifiable, while the inference about the effort behind it remains subject to sampling uncertainty and the stated observation assumptions.

Information theory quantifies uncertainty; statistical inference describes how evidence changes it; computation supplies procedures for resolving it; and thermodynamics constrains the physical systems in which those procedures are realized. Part III will examine the implications of these relationships for coordination and resource allocation.

A From Bernoulli Trials to Continuous-Time Processes

A.1 From Binomial to Poisson

Consider a sequence of binomial distributions in which the number of trials increases while the probability of success decreases, such that the expected number of successes remains constant.

Setting $p = \mu/n$ gives

$$\mathbb{P}(K_n = k) = \binom{n}{k} \left(\frac{\mu}{n}\right)^k \left(1 - \frac{\mu}{n}\right)^{n-k}.$$

For fixed k , as n grows without bound,

$$\binom{n}{k} \left(\frac{\mu}{n}\right)^k \longrightarrow \frac{\mu^k}{k!}, \quad \left(1 - \frac{\mu}{n}\right)^{n-k} \longrightarrow e^{-\mu}.$$

Combining these limits gives

$$\mathbb{P}(K_n = k) \longrightarrow \frac{\mu^k e^{-\mu}}{k!}.$$

The Poisson distribution counts discrete events even when those events occur in continuous time.

A.2 From Geometric to Exponential

Recall the geometric survival probability:

$$\mathbb{P}(N > n) = (1 - p)^n.$$

For a constant attempt rate H , the number of attempts during t seconds is approximately Ht . Using $\ln(1 - p) \approx -p$ for small p ,

$$\mathbb{P}(T > t) \approx (1 - p)^{Ht} = \exp(Ht \ln(1 - p)) \approx e^{-Hpt}.$$

More precisely, let $p_H = \lambda/H$ and $N_{p_H} \sim \text{Geometric}(p_H)$. Then, as $H \rightarrow \infty$,

$$\mathbb{P}\left(\frac{N_{p_H}}{H} > t\right) = \left(1 - \frac{\lambda}{H}\right)^{\lfloor Ht \rfloor} \longrightarrow e^{-\lambda t}.$$

Thus N_{p_H}/H converges in distribution to $\text{Exponential}(\lambda)$. The discrete number of attempts is thereby converted into a continuous waiting-time distribution. Within the resulting homogeneous Poisson model, the relationship between no arrivals and the exponential waiting time in Section 6.4 is exact.

B Extended Difficulty-Adjustment Example

Suppose the target block interval is $\tau = 5$ seconds, but the observed average over a measurement window is $\bar{T} = 4$ seconds. The target and observed arrival rates are

$$\lambda_{\text{target}} = \frac{1}{5} = 0.2, \quad \hat{\lambda} = \frac{1}{4} = 0.25 \quad \text{blocks/s.}$$

The observed rate is greater than the target by a factor of 1.25. Assuming hashrate remains unchanged, restoring the target rate requires

$$p_{\text{new}} = \frac{\lambda_{\text{target}}}{\hat{\lambda}} p_{\text{old}} = 0.8 p_{\text{old}}.$$

The expected hashes per proof, $1/p$, therefore increase by a factor of 1.25. In terms of effective difficulty bits,

$$d_{\text{new}} = d_{\text{old}} + \log_2(1.25) \approx d_{\text{old}} + 0.322.$$

The fractional-bit representation is mathematical shorthand: practical systems can use numerical hash targets to achieve finer adjustments than an integer leading-zero rule [12].

For m measured intervals,

$$\bar{T} = \frac{1}{m} \sum_{i=1}^m T_i.$$

If D is proportional to the expected number of hash attempts per proof, a simplified proportional update is

$$D_{\text{new}} = D_{\text{old}} \frac{\tau}{\bar{T}}.$$

Thus $\bar{T} < \tau$ increases difficulty, while $\bar{T} > \tau$ decreases it. Actual protocols may add rolling averages, adjustment limits, timestamp rules, and consensus-specific constraints; the calculation here isolates the statistical feedback principle.

C Exact Poisson Confidence Bounds

Let $K \sim \text{Poisson}(\mu)$, where $\mu = pt\bar{H}$. For an observed count $k \geq 1$, the endpoints of an exact equal-tailed 95% confidence interval for μ solve

$$\mathbb{P}_{\mu_L}(K \geq k) = 0.025, \quad \mathbb{P}_{\mu_U}(K \leq k) = 0.025.$$

Here the subscript specifies the mean used to evaluate the Poisson probability. These equations can be solved numerically using the Poisson distribution already developed in the paper.

An equivalent expression uses the relationship between Poisson tail probabilities and the chi-squared distribution:

$$\mu_L = \frac{1}{2} \chi_{2k, 0.025}^2, \quad \mu_U = \frac{1}{2} \chi_{2k+2, 0.975}^2,$$

where $\chi_{\nu, q}^2$ is the q -quantile of a chi-squared distribution with ν degrees of freedom. This is a way to express the Poisson endpoints; it introduces no additional assumption about the hashing process. Dividing by pt gives the interval for average hashrate:

$$\left[\frac{\chi_{2k, 0.025}^2}{2pt}, \frac{\chi_{2k+2, 0.975}^2}{2pt} \right].$$

For $k = 0$, the lower endpoint is defined as zero. The upper endpoint of this two-sided interval is $-\ln(0.025)/(pt)$; the one-sided 95% upper limit in Section 7.2 instead uses $-\ln(0.05)/(pt)$.

References

- [1] E. T. Jaynes, “Information Theory and Statistical Mechanics,” *Physical Review*, vol. 106, pp. 620–630, 1957. doi: 10.1103/PhysRev.106.620.

- [2] E. T. Jaynes, *Probability Theory: The Logic of Science*, Cambridge University Press, 2003.
- [3] C. E. Shannon, “A Mathematical Theory of Communication,” *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x.
- [4] L. Boltzmann, “Über die Beziehung zwischen dem zweiten Hauptsatze der mechanischen Wärmetheorie und der Wahrscheinlichkeitsrechnung respective den Sätzen über das Wärmegleichgewicht,” *Sitzungsberichte der Kaiserlichen Akademie der Wissenschaften*, vol. 76, pp. 373–435, 1877.
- [5] J. W. Gibbs, “On the Equilibrium of Heterogeneous Substances,” *Transactions of the Connecticut Academy of Arts and Sciences*, vol. 3, pp. 108–248, 1876; pp. 343–524, 1878.
- [6] J. W. Gibbs, *Elementary Principles in Statistical Mechanics: Developed with Especial Reference to the Rational Foundation of Thermodynamics*, Charles Scribner’s Sons, 1902.
- [7] L. Brillouin, *Science and Information Theory*, Academic Press, 1956.
- [8] J. M. Keynes, *A Treatise on Probability*, Macmillan and Co., 1921.
- [9] N. N. Taleb, *Statistical Consequences of Fat Tails: Real World Preasymptotics, Epistemology, and Applications*, STEM Academic Press, 2020.
- [10] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008. Available: <https://bitcoin.org/bitcoin.pdf>.
- [11] K. Kreder, S. Shastri, A. Tzinas, S. Vishwanath, and D. Zindros, “A Better Proof-of-Work Fork Choice Rule,” IACR Cryptology ePrint Archive, Report 2024/200, 2024. Available: <https://eprint.iacr.org/2024/200>.
- [12] Bitcoin Developer Reference, “Block Chain: Block Headers and Target nBits.” Available: https://developer.bitcoin.org/reference/block_chain.html.
- [13] NIST/SEMATECH, *e-Handbook of Statistical Methods*, Section 8.4.1.2, “Maximum Likelihood Estimation.” Available: <https://www.itl.nist.gov/div898/handbook/apr/section4/apr412.htm>.
- [14] R. Landauer, “Irreversibility and Heat Generation in the Computing Process,” *IBM Journal of Research and Development*, vol. 5, no. 3, pp. 183–191, 1961. doi: 10.1147/rd.53.0183.
- [15] C. H. Bennett, “Logical Reversibility of Computation,” *IBM Journal of Research and Development*, vol. 17, no. 6, pp. 525–532, 1973. doi: 10.1147/rd.176.0525.
- [16] T. Sagawa, “Thermodynamic and Logical Reversibilities Revisited,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2014, P03025, 2014. doi: 10.1088/1742-5468/2014/03/P03025.
- [17] D. Reeb and M. M. Wolf, “An Improved Landauer Principle with Finite-Size Corrections,” *New Journal of Physics*, vol. 16, 103011, 2014. doi: 10.1088/1367-2630/16/10/103011.
- [18] J. M. R. Parrondo, J. M. Horowitz, and T. Sagawa, “Thermodynamics of Information,” *Nature Physics*, vol. 11, pp. 131–139, 2015. doi: 10.1038/nphys3230.