The Amortization Envelope: Where a SHA-256d Mining Advantage Can and Cannot Live

Peter Hollows

June 22, 2026

Abstract

Bitcoin mining searches for a block header whose double-SHA-256 hash falls below a target, a weak-preimage problem solved in practice by brute force. We ask whether any classical algorithm beats brute force, and answer with a complete accounting of where an advantage could live. Decomposing SHA-256d into two layers with opposite properties (a shareable message schedule and a pseudorandom round function) splits the cost into a per-candidate factor and a candidate-count factor. We show the per-candidate factor is confined to three known, bounded regions (midstate caching, the ASICBoost message-expansion region, and a provably unshareable second hash), so it is near-fixed. We prove, in the random-oracle model, that the candidate count cannot fall below 2^D, a bound that survives preprocessing (Bitcoin is salted) and quantum search (Grover gives only a quadratic speedup), and we reduce the standard-model claim to a SHA-256 distinguisher. We close the remaining economic loophole (selecting which header prefixes to mine) by showing stems are exchangeable under the natural generator, as the round function's mixing predicts. The result localizes the entire residual risk to one object: a global algebraic shortcut through SHA-256, which is a distinguisher. *Mining cannot beat brute force unless SHA-256 is broken.* Whether that residual is empty is an empirical question about SHA-256's round function, which this paper isolates but does not resolve; the public cryptanalytic record is the present evidence.

1 The mining search problem

Bitcoin mining is a search problem over an iterated hash. A block header H is 80 bytes; the mining hash is the double application

$$\mathsf{M}(H) \;=\; \text{SHA-256}\big(\text{SHA-256}(H)\big),$$

read as a 256-bit integer. Mining succeeds when a miner finds an H (within its allowed degrees of freedom: the 32-bit nonce, the coinbase extranonce that varies the Merkle root, version bits, and the timestamp) such that $\mathsf{M}(H) < T$, where $T = 2^{256} \cdot 2^{-D}$ and D is the difficulty expressed in leading zero bits. Equivalently, H must land in a target set $S \subseteq \{0,1\}^{256}$ of density $|S|/2^{256} = 2^{-D}$.

The question is operational: *is there any classical algorithm that finds such an H at materially less expected cost than brute force?* Brute force tries candidates until one lands in S, at expected cost $\approx 2^D$ evaluations. An attack "beats" it only if it reduces the total work by more than a constant factor.

Cost model. The atomic unit is one SHA-256 compression-function call $h : \{0,1\}^{256} \times \{0,1\}^{512} \to \{0,1\}^{256}$. It applies the block cipher E (SHACAL-2, keyed by the message m) to the chaining value and feeds the input forward by *wordwise modular addition*, $h(C, m) = C \boxplus E_m(C)$, where $\boxplus$ adds each of the eight 32-bit words modulo 2^{32}. The feed-forward is thus itself a carry operation, which matters for the round function's resistance to prediction. One header evaluation decomposes into exactly three compression calls,

$$\underbrace{C_1 = h(\mathrm{IV}, B_0)}_{\text{block 0 (nonce-independent)}} \quad , \quad \underbrace{C_2 = h(C_1, B_1 \| \mathrm{pad})}_{\text{block 1 (nonce in word } W_3)} , \quad \underbrace{\mathrm{out} = h(\mathrm{IV}, C_2 \| \mathrm{pad})}_{\text{second hash}},$$

with B_0 the first 64 header bytes and B_1 the remaining 16 plus padding. Total mining cost is the number of candidate evaluations multiplied by the compression calls per candidate that sharing cannot remove. These two factors are governed by different structure, which the next section makes precise.

2 Two layers

The analysis rests on one structural fact: SHA-256d is built from two kinds of structure with opposite properties.

layer	**property**	**consequence**
message schedule	fixed recurrence (same per block)	computation is *shareable* across candidates
round / compression	pseudorandom	output is *unpredictable*; search is *irreducible*

The only avenues to beat brute force live in one layer or the other: either reduce the *per-candidate cost* by sharing schedule computation (Section 3), or reduce the *candidate count* by predicting or skipping evaluations (Section 4). We show the first avenue is bounded to known constant-factor techniques, and the second is impossible in the ideal model and equivalent to breaking SHA-256 in the standard model. Because the two avenues act on different layers, a gain in one neither requires nor enables a gain in the other, and the rest of the paper addresses them in turn. *Why* the round function is pseudorandom (the mechanism behind the second row) is beyond the scope of this paper; here we take it as the hypothesis to be localized and reduced.

3 The amortization envelope

Written as a product, the cost is

$$\text{cost} \;\geq\; \underbrace{(\text{candidate evaluations})}_{\text{locked at } 2^D,\ §4} \;\times\; \underbrace{(\text{compressions per candidate after maximal sharing})}_{\text{this section}}.$$

The left factor is locked at 2^D by the search lower bound. Hence any economically meaningful win against brute force must come from the right factor: per-candidate computation sharing. We bound it by classifying which of the three compressions can be shared across *distinct* candidates.

- **Block 0 (midstate).** Nonce-independent, and shared across the entire nonce sweep of a fixed header prefix, so its amortized cost per candidate falls to zero. This is midstate caching, standard practice in deployed miners.

- **Second hash.** Effectively unshareable. Two candidates share the second-hash compression only if they share a first-hash digest, which is a SHA-256 collision. Finding one costs about 2^{128}, far more than mining the block itself (cost 2^D with $D < 128$), so the second hash cannot be shared profitably and every candidate pays one full second-hash compression. This rests on the collision resistance of SHA-256, the mildest cryptographic assumption in the paper, and no input-side technique avoids it.
- **Block 1 expansion.** Block 1 of the first hash contains the nonce, and only its message-schedule expansion is shareable, across small batches of candidates engineered to collide in the expansion. ASICBoost [2] exploits exactly this region, covertly via the Merkle root or overtly via version rolling (standardized in the reserved `nVersion` bits [3]). It is a deployed technique worth about 20%, and the only region where a sharing win can live.

These three regions are the three compressions of the cost model, so they exhaust the per-candidate work: the sharing envelope is exactly midstate (at floor), the ASICBoost block-1 expansion (near-saturated), and the second hash (provably unshareable), with nothing left over. Sharing *between* the compressions is excluded by the same data dependency: each compression consumes the previous one's output, so two candidates share a later compression only if they collide in an earlier digest, which is again a SHA-256 collision at cost 2^{128}. The amortization avenue is therefore known, bounded, and already occupied, and the remainder of this paper concerns the candidate-count factor, where the open question lay.

4 The search lower bound

We first establish, unconditionally in the ideal model, that the candidate-count factor cannot be reduced below 2^D in expectation; then we transfer the statement to the standard model by a tight reduction.

4.1 Ideal model

Model the search map $H \mapsto \mathsf{M}(H)$ as a random oracle: with the compression function idealized, fresh inputs receive uniform, independent outputs (composition of ideal primitives is not automatic in general; the construction-level caveats are handled in the standard-model reduction below). This is the standard idealization under which proof-of-work hardness is proved [1].

Theorem 1 (ROM search optimality)**.** *Let $\mathcal{A}$ make at most q queries to the oracle and output a candidate H^*. Then*

$$\Pr[\mathsf{M}(H^*) \in S] \;\le\; (q+1)\,2^{-D},$$

over the oracle and $\mathcal{A}$'s coins. Consequently $\mathcal{A}$ needs $q \ge 2^{D-1} - 1$ queries to succeed with probability $\ge \frac{1}{2}$, and the expected number of queries to the first success is $\ge 2^D$.

Proof sketch. Without loss of generality the queries $x_1, \ldots, x_q$ are distinct. Each $\mathrm{RO}(x_i)$ is uniform on $\{0,1\}^{256}$, independent of all prior responses, so $\Pr[\mathrm{RO}(x_i) \in S] = 2^{-D}$. If H^* was queried, success is among these events; if not, $\mathrm{RO}(H^*)$ is uniform and independent, contributing a further 2^{-D}. A union bound over the q queries and the final output gives $(q+1)2^{-D}$. The query statement is immediate. For the expectation, random-oracle freshness makes each query succeed with probability at most 2^{-D} conditioned on all prior responses, so the first-success time Q stochastically dominates a Geometric(2^{-D}) variable: $\Pr[Q > q] \ge (1 - 2^{-D})^q$. Hence $\mathbb{E}[Q] = \sum_{q \ge 0} \Pr[Q > q] \ge \sum_{q \ge 0} (1 - 2^{-D})^q = 2^D$. □

Brute force achieves expected 2^D evaluations, so it is exactly query-optimal in expectation: no candidate ordering, feature, or filter reduces the count below 2^D in this model.

4.2 Closing the realistic loopholes

Precomputation and time–memory trade-offs. A miner might precompute an enormous table offline. The auxiliary-input (non-uniform) random-oracle framework [4, 5, 6] models an adversary with S bits of arbitrary advice and q online queries and shows that *salting* generically defeats preprocessing. For the salted inversion problem the advantage is bounded by $ST/(K\alpha) + T/\alpha$, which for a salt as wide as the function's output ($K = N$) collapses to $\mathcal{O}(T/N)$ [7], i.e. the non-preprocessing bound up to constants. Bitcoin is salted by construction: each header commits to `prev_block_hash`, a fresh, unpredictable 256-bit value, and to the Merkle root of fresh transactions, so the per-block salt is effectively as wide as the function output. Classical time–memory trade-offs [8] and their rainbow-table refinement [9] require a *fixed* target function so that one precomputed table amortizes over all instances; a fresh per-block salt makes each block a distinct function, and offline computation cannot target a block whose predecessor is not yet known, so these trade-offs do not apply.

Quantum. Grover search gives a quadratic speedup, $\mathcal{O}(2^{D/2})$ queries, and this is optimal (the BBBV lower bound [10]): no exponential speedup exists even quantumly. This carries over to the mining setting specifically: the post-quantum analysis of the Bitcoin backbone proves that each quantum query is worth only $\mathcal{O}(p^{-1/2})$ classical queries, with a unified hybrid quantum/classical search bound interpolating the two regimes [11, 12]. The quadratic factor is economically irrelevant at current scale, since a coherent SHA-256d Grover oracle costs orders of magnitude more per query than a classical ASIC gate-pass.

Standard-model reduction. A concrete miner $\mathcal{M}$ against the true SHA-256 with expected cost $2^D/\alpha$ for super-constant α succeeds at a rate exceeding the ceiling that Theorem 1 imposes on every algorithm in the ROM. Running $\mathcal{M}$ against a real or an ideal primitive and checking whether it beats the bound is then a distinguishing experiment, formalized by indifferentiability [13]: if the mining map is indifferentiable from a random oracle, the ROM bound transfers to the standard model up to the distinguishing advantage, and standard-model proof-of-work primitives of exactly this lower-bound shape have been formalized [14]. One subtlety must be handled. Plain (strengthened) Merkle–Damgård is *not* indifferentiable from a random oracle, because of length extension: from SHA-256(x) one can predict SHA-256($x\|\text{pad}\|y$) without knowing x [15]. This is the structure that the *outer* application in SHA-256d = SHA-256(SHA-256($\cdot$)) is designed to remove (length extension specifically): re-hashing the full digest is intended to seal the inner chaining value behind a second, length-committed invocation, so that the length-extension distinguisher does not reach the mining map. The literature comes close to this composition from both sides without covering it. Coron et al. [15] prove plain Merkle–Damgård indifferentiable (with birthday bounds) on prefix-free inputs, of which fixed-length inputs are a special case (so a *single* SHA-256 restricted to 80-byte headers is covered), and separately prove indifferentiability of the double hash $H(H(0^\kappa\|M))$, which differs from SHA-256d only by the prepended zero block whose role is to domain-separate the outer single-compression call from adversarially controlled first message blocks; SHA-256d omits that block, so the theorem does not literally apply. Dodis et al. [16] analyze the unrestricted double hash $H^2(M) = H(H(M))$ and show it is indifferentiable only with weak bounds (any successful simulator must make $\Omega(\min\{q_1 q_2, 2^{n/2}\})$ queries, capping composition-derived security near $2^{n/4}$); their differentiating attack, however, requires feeding n-bit digests back

into the construction, which the mining map's fixed 80-byte domain forbids, and they prove no positive result for the domain-restricted composition. No published theorem covers SHA-256d on its fixed-length domain, so we state indifferentiability of the mining map as an explicit assumption, consistent with formal treatments of Bitcoin that model the mining function as a random oracle directly [1]. Indifferentiability composes for single-stage games; multi-stage settings can require more care [17], but the mining experiment is single-stage: one adversary, one search.

Theorem 2 (conditional, informal)**.** *If the mining map* $\mathsf{M} = \text{SHA-256}(\text{SHA-256}(\cdot))$ *is indifferentiable from a random oracle (with its compression function modeled as ideal), then no mining algorithm achieves expected cost below* $2^D/\mathcal{O}(1)$ *candidate evaluations, up to the indifferentiability advantage; an algorithm that does yields a distinguisher against* SHA-256*.*

The provability wall. A fully unconditional, standard-model version of Theorem 2 is beyond current techniques. Mining for the fixed function SHA-256 is finite, so the statement is not literally about P versus NP; an unconditional proof would be an explicit circuit lower bound, which no method can establish for a function of this complexity. In the asymptotic framing, for a scaled family of SHA-256-like functions, proving that search requires $\sim 2^D$ work would exhibit an explicit one-way function and so separate P from NP. Either way, the conditional reduction above is the strongest conclusive statement available, and it localizes the entire risk to one assumption.

4.3 Localization

Corollary 1. *Every economically meaningful win against brute force is confined to the per-candidate sharing factor of Section 3. Output prediction and search-skipping are impossible in the ideal model (Theorem 1) and equivalent to a* SHA-256 *break in the standard model (Theorem 2).*

Whether the round-function layer admits such prediction is exactly the residual this localization isolates: an empirical question about SHA-256, not one the reduction settles. One economic loophole in the corollary remains to be closed directly, and we do so next.

5 Stem selection

Theorem 1 locks the candidate count for a *fixed* header prefix. A miner with freedom over the prefix (the extranonce, version bits, and timestamp) might still hope that some prefixes, which we call *stems*, yield more readily than others, and allocate effort to the best. This would not contradict the per-stem bound, but it would beat brute force in aggregate. We close the loophole: under the natural generator, stems are exchangeable, and any yield bias is bounded by the round function's mixing.

Define the yield of a stem as the extreme-value statistic of a full nonce sweep. Sweeping K nonces and recording the best (smallest) hash, set $Y = -\log_2(\text{min-hash}/2^{256})$, the number of leading zero bits of the best hash found. Under the random-oracle idealization the per-nonce hashes are i.i.d. uniform, so Y follows a Gumbel law with $\mathbb{E}[Y] \approx \log_2 K + 0.83$ and variance ≈ 3.41, *independent of the stem*. A stem carries exploitable structure only if its yield deviates from this law, equivalently if some cheap, nonce-independent partition of stems has a heavier tail than the null.

We tested this directly. Under a synthetic mining-template generator, sweeping on the order of 10^4 nonces over each of $\sim 3 \times 10^4$ stems, no partition drawn from a pre-registered feature catalogue beat its null tail-hit rate through a three-stage discovery / confirmation / fresh-holdout protocol with multiple-testing correction (no partition of several dozen survived across hundreds of buckets);

a positive control with an injected per-partition bias was recovered, and a separate family of synthetic-uniform negative controls produced no lift. Under the tested generator and catalogue, stems carry no exploitable yield. (Full pre-registration, generator, feature catalogue, and per-bucket results accompany this paper as code. The closure is for the synthetic generator; we did not obtain operational pool work-queue data, so the claim is the natural-model statement, not an audit of any specific deployed generator.)

The measurement is what one expects if the round function thoroughly mixes the stem bits into the output. A stem fixes some input bits, but their influence on any one output bit is diffused through the many carry-coupled modular additions between the header and the digest, so no cheap, nonce-independent feature of a stem survives to bias a nonce sweep. The empirical exchangeability is the evidence; this mixing picture only indicates why it is unsurprising.

6 Related work and conclusion

Proof-of-work security is conventionally proved in the random-oracle model [1], with standard-model [14] and post-quantum [11, 12] analyses extending it. As we make explicit, the impossibility of beating brute force is not provable unconditionally in the standard model without resolving open hardness questions. The contribution here is the *complete accounting*. A bounded sharing envelope, an unconditional ideal-model search floor, a tight standard-model reduction, and a closed stem-selection loophole together localize the entire residual to one object: a global algebraic shortcut through SHA-256 that resolves the target output without resolving the carries on the path, which by the reduction is a distinguisher against SHA-256. ASICBoost [2] is the canonical input-side win, and it occupies the only shareable region we identify. Auxiliary-input bounds [5, 6, 4, 7] and the BBBV quantum lower bound [10] close the preprocessing and quantum loopholes. The public cryptanalytic record is consistent with this ceiling: reduced-round collision attacks reach only 31 of 64 steps (made practical in [18], building on [19]) and 39 steps semi-free-start [18], and preimages reach 43 steps at cost $2^{254.9}$ [20], all far from the full function.

Whether that residual is empty is an empirical question about SHA-256's round function. The public cryptanalytic record is the present evidence: no shortcut of the required kind has been found, and the reduced-round results above remain far from the full function.

SHA-256d mining cannot beat brute force by any route we or the cryptanalytic record can find, and cannot at all unless SHA-256 is broken. The only standing economic wins are the known constant-factor sharing techniques, and they are bounded.

References

[1] J. Garay, A. Kiayias, N. Leonardos. *The Bitcoin Backbone Protocol: Analysis and Applications.* EUROCRYPT 2015.

[2] T. Hanke. *AsicBoost: A Speedup for Bitcoin Mining.* 2016, `arXiv:1604.00575`.

[3] *nVersion Bits for General Purpose Use (BIP 320).* Bitcoin Improvement Proposals, 2018.

[4] D. Unruh. *Random Oracles and Auxiliary Input.* CRYPTO 2007.

[5] S. Coretti, Y. Dodis, S. Guo, J. Steinberger. *Random Oracles and Non-Uniformity.* EUROCRYPT 2018.

[6] S. Coretti, Y. Dodis, S. Guo. *Non-Uniform Bounds in the Random-Permutation, Ideal-Cipher, and Generic-Group Models.* CRYPTO 2018.

[7] Y. Dodis, S. Guo, J. Katz. *Fixing Cracks in the Concrete: Random Oracles with Auxiliary Input, Revisited.* EUROCRYPT 2017.

[8] M. E. Hellman. *A Cryptanalytic Time-Memory Trade-Off.* IEEE Trans. Inf. Theory 26(4), 1980.

[9] P. Oechslin. *Making a Faster Cryptanalytic Time-Memory Trade-Off.* CRYPTO 2003.

[10] C. Bennett, E. Bernstein, G. Brassard, U. Vazirani. *Strengths and Weaknesses of Quantum Computing.* SIAM J. Comput. 26(5), 1997.

[11] A. Cojocaru, J. Garay, A. Kiayias, F. Song, P. Wallden. *Quantum Multi-Solution Bernoulli Search with Applications to Bitcoin's Post-Quantum Security.* Quantum 7:944, 2023.

[12] A. Cojocaru, J. Garay, F. Song. *Generalized Hybrid Search with Applications to Blockchains and Hash Function Security.* ASIACRYPT 2024.

[13] U. Maurer, R. Renner, C. Holenstein. *Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology.* TCC 2004.

[14] J. Garay, A. Kiayias, G. Panagiotakos. *Proofs of Work for Blockchain Protocols.* IACR Cryptology ePrint Archive, Report 2017/775, 2017.

[15] J.-S. Coron, Y. Dodis, C. Malinaud, P. Puniya. *Merkle-Damgård Revisited: How to Construct a Hash Function.* CRYPTO 2005.

[16] Y. Dodis, T. Ristenpart, J. Steinberger, S. Tessaro. *To Hash or Not to Hash Again? (In)differentiability Results for H^2 and HMAC.* CRYPTO 2012.

[17] T. Ristenpart, H. Shacham, T. Shrimpton. *Careful with Composition: Limitations of the Indifferentiability Framework.* EUROCRYPT 2011.

[18] Y. Li, F. Liu, G. Wang. *New Records in Collision Attacks on SHA-2.* EUROCRYPT 2024.

[19] F. Mendel, T. Nad, M. Schläffer. *Improving Local Collisions: New Attacks on Reduced SHA-256.* EUROCRYPT 2013.

[20] K. Aoki, J. Guo, K. Matusiewicz, Y. Sasaki, L. Wang. *Preimages for Step-Reduced SHA-2.* ASIACRYPT 2009.