Why Ethereum Cannot Scale to a Literal "World Computer" Even With Rollups, DAS, and External DA: Information-Flow Data-Availability Lower Bounds and Global-Log Coupling vs. DSM

Brandon "Cryptskii" Ramsay

December 22, 2025

Abstract

Ethereum's scaling narrative is often summarized as "rollups make Ethereum a world computer" by moving execution off-chain while preserving trust-minimized settlement on L1. This paper formalizes why that narrative fails under a literal world-computer requirement: any trust-minimized rollup ecosystem must consume a conserved quantity—data availability (DA)—secured under decentralization constraints. Data Availability Sampling (DAS) can reduce *validator verification* bandwidth, but does not remove the *information-flow* requirement that some decentralization-eligible publisher upload, and some independent retriever set download and reconstruct, the DA payload every step. Hence aggregate trustless throughput remains bounded by a constant independent of adoption, so average per-user service vanishes as users grow. We contrast this with a Deterministic State Machine (DSM) architecture where verification is endpoint-local and unrelated relationships do not share a global ordering domain, yielding additive throughput across devices. We also formalize why DSM has no reorg class, sharply reduced MEV surface, and why online transfer does not require recipient signatures to be final at the sender.

1 Context, Goal, and What "World Computer" Means Here

The foundational promise of a "world computer" is not merely *some* global settlement; it is a substrate where arbitrary parties can perform trustless state transitions at planetary scale without per-user service collapsing simply because more humans exist.

The DSM report already identifies the structural culprit for blockchain-style ceilings: a single global ordering domain couples unrelated interactions (congestion externalities), creates reorg and MEV surfaces, and forces world-scale demand through one narrow global pipeline. DSM's alternative is relationship-scoped, forward-only determinism where unrelated relationships do not serialize behind each other.

This paper targets the specific claim: *"Ethereum can still be a world computer because rollup-based L2s scale it."* We show that rollups move computation, but they cannot move information if the system remains trust-minimized.

Sources for current Ethereum DA primitives. EIP-4844 introduces fixed-size blobs (131,072 bytes) as a DA resource for rollups.[1] Ethereum's roadmap frames danksharding as a DA-scaling system via data sampling across blobs.[2] Pectra/EIP-7691 increased the blob target/max from (3,6) to (6,9) per block.[3] These raise the constant DA budget, not the asymptotics.

[1] `https://eips.ethereum.org/EIPS/eip-4844`.

[2] `https://ethereum.org/roadmap/danksharding/`.

[3] `https://eips.ethereum.org/EIPS/eip-7691`.

2 Model: DA, Trust-Minimized L2, and Conserved Resources

Definition 1 (Protocol step)**.** *A* step *is a single protocol advancement unit (e.g. one L1 block/slot). All rates in this paper are expressed as per-step quantities. No wall-clock time is required.*

Definition 2 (Data Availability (DA) capacity)**.** *Let C be the maximum number of* raw *DA-bytes per step that the DA layer can make available under its decentralization constraint (i.e. without requiring only privileged hardware/network participants to remain eligible). We call C the DA capacity (bytes/step).*

Definition 3 (Trust-minimized L2)**.** *An L2 is* trust-minimized *if:*

- *(Verification) correctness can be checked without trusting a privileged operator; and*
- *(Exit/inclusion) a user can unilaterally force inclusion or exit using only the DA-layer-available information and the L1 (or DA layer) protocol rules.*

Definition 4 (Per-transition DA footprint)**.** *Let $s > 0$ be a lower bound (bytes/transition) on the DA-layer-published information required per L2 transition to preserve permissionless verification and unilateral exits. Validity proofs compress* compute*; they do not eliminate the need to make an exit-complete witness publicly reconstructable.*

3 Core Lower Bound: Trustless L2 Throughput is DA-Bounded

Lemma 1 (Exit-completeness implies published information)**.** *Consider any trust-minimized L2 that commits a batch of L2 state transitions to a DA layer via a per-step commitment. Let $\mathcal{I}$ be the minimum information required for an arbitrary verifier to reconstruct the batch's effect and compute exit claims. Then $\mathcal{I}$ (or an exit-equivalent witness) must be made available on the DA layer; otherwise verifiers cannot distinguish multiple batch histories consistent with the commitment.*

Proof. Model the commitment as a function g from the space of possible batch histories $\mathcal{H}$ to published transcripts $\mathcal{T}$ that are DA-available in that step. If the DA transcript omits an exit-complete witness, then there exist at least two distinct histories $H_1 \neq H_2$ in $\mathcal{H}$ that induce identical DA transcripts, i.e. $g(H_1) = g(H_2)$, while producing different user exit claims. (This is the indistinguishability condition: the verifier's view is identical but the correct outcome differs.) A verifier observing only the DA transcript cannot determine which history occurred and therefore cannot compute correct exits without trusting an operator. This violates trust-minimization. Hence an exit-complete witness (or equivalent) must be DA-available. □

Theorem 1 (DA bottleneck theorem: aggregate trust-minimized L2 throughput is DA-bounded)**.** *Let C be DA capacity (bytes/step) and $s > 0$ be a per-transition DA lower bound. Then the aggregate throughput of* all *trust-minimized L2s settling to that DA layer, T_{L2} (transitions/step), satisfies*

$$T_{\mathrm{L2}} \leq \frac{C}{s}.$$

Proof. Each trust-minimized L2 transition requires at least s DA-bytes, otherwise Lemma 1 fails. Only C bytes can be made available per step. Hence $T_{\mathrm{L2}} \cdot s \leq C$, so $T_{\mathrm{L2}} \leq C/s$. □

Immediate consequence. "More rollups" does not create more trustless throughput; it partitions a shared DA budget.

4 DAS: Correct Security Math and Why It Does *Not* Remove the DA Ceiling

A common mistake is to treat DAS as if it makes DA capacity scale like "number of validators $\times$ validator bandwidth." That is not the relevant bound. DAS reduces *verification* cost per validator. The DA ceiling is governed by *information flow*: the bytes must still be uploaded and downloaded by some decentralization-eligible publisher and some independent retriever set.

4.1 Erasure-coded DA and sampling security

Definition 5 (Erasure-coded payload and shares)**.** *Let C be raw DA-bytes per step and let $k \geq 1$ be an erasure-coding expansion factor, so the coded payload is kC bytes/step. Partition the coded payload into n equal-sized shares of s_{sh} bytes:*

$$n = \frac{kC}{s_{\text{sh}}}.$$

Assume a standard threshold property: any $(1-\beta)n$ shares suffice to reconstruct, for some $\beta \in (0,1)$.

Definition 6 (Sampling model)**.** *Let h be the number of honest samplers (validators, light clients, or DA verifiers). Each honest sampler draws q shares uniformly at random (without replacement) from the n shares. Define $\alpha := q/n$. An adversary withholds a fraction β of shares.*

Theorem 2 (DAS detection probability)**.** *Under the above model, the probability that* no honest sampler *hits any withheld share is bounded by*

$$\Pr[\text{miss}] \leq (1-\beta)^{hq} \leq e^{-\beta hq}.$$

Hence

$$\Pr[\text{detect}] \geq 1 - e^{-\beta hq}.$$

Proof. Each draw avoids withheld shares with probability at most $(1-\beta)$. Over hq draws, the miss probability is at most $(1-\beta)^{hq}$. Using $1 - x \leq e^{-x}$ gives the exponential bound. □

Corollary 1 (Samples required for $(1-\delta)$ detection)**.** *To ensure* $\Pr[\text{detect}] \geq 1-\delta$*, it suffices that*

$$q \geq \frac{\ln(1/\delta)}{\beta h}.$$

Proof. Set $e^{-\beta hq} \leq \delta$ and solve for q. □

Key correction. DAS security depends on hq; as honest sampling participation increases, each sampler can sample fewer shares while maintaining fixed detection security. This is about verification cost, not about making C scale with validator count.

4.2 The real ceiling: information must still move

Definition 7 (Decentralization-eligible publisher and independent retrievers)**.** *Let $\mathcal{P}$ be the set of decentralization-eligible publishers (proposers/builders) who must be able to publish the coded payload in a step. Let each $p \in \mathcal{P}$ have per-step uplink capacity U_p (bytes/step), and define*

$$U_* := \inf_{p \in \mathcal{P}} U_p.$$

Let $\mathcal{R}$ be a set of independent retrievers such that the system's trust-minimization requires that no single operator is trusted to serve data. Let each $r \in \mathcal{R}$ have per-step download capacity D_r and define

$$D_* := \sum_{r \in \mathcal{R}} D_r.$$

Theorem 3 (Information-flow DA bound (DAS does not remove it))**.** *For any DA design (with or without DAS), if (i) a decentralization-eligible publisher must upload the coded payload each step and (ii) an independent retriever set must be able to reconstruct the payload each step, then*

$$kC \le U_*, \qquad kC \le D_*,$$

hence

$$C \le \min\left(\frac{U_*}{k}, \frac{D_*}{k}\right).$$

Proof. Publishing kC coded bytes in a step is impossible unless an eligible publisher can upload kC bytes in that step, so $kC \le U_*$. Reconstruction by independent retrievers requires that enough coded bytes can be downloaded in aggregate; a necessary condition is $kC \le D_*$. Combining yields the bound. □

Theorem 4 (Aggregate trust-minimized L2 ceiling under DAS)**.** *For any ecosystem of trust-minimized L2s settling to this DA layer,*

$$T_{\mathrm{L2}} \le \frac{C}{s} \le \frac{1}{s}\min\left(\frac{U_*}{k}, \frac{D_*}{k}\right).$$

Proof. Apply Theorem 1 and substitute the bound from Theorem 3. □

5 World-Computer Impossibility for Ethereum-Class L1+L2

Definition 8 (World-computer scalability requirement)**.** *A system satisfies the world-computer scalability requirement if there exists $\epsilon > 0$ such that for arbitrarily large user populations N, the average per-user service rate is at least ϵ under the system's decentralization constraints.*

Theorem 5 (World-computer impossibility under conserved DA)**.** *Assume:*

- *(Trust-minimized L2) L2s require DA-available exit-complete information (so $s > 0$).*
- *(Stable decentralization constraint) eligible publishers and independent retrievers are commodity-grade, so U_*, D_*, k, s do not scale with N.*

Then Ethereum-class architectures (global DA-coupled L1 with trust-minimized rollups) fail the world-computer scalability requirement.

Proof. By Theorem 4, T_{L2} is bounded by a constant independent of N. Therefore

$$\frac{T_{\mathrm{L2}}}{N} \to 0 \quad \text{as } N \to \infty,$$

which contradicts Definition 8. □

6 Why "External DA" Does Not Save the Claim

External DA (e.g. committee DA, restaked DA, modular DA chains) changes the location of C, but it does not remove the form of the bound.

Proposition 1 (Relocating DA relocates, but does not delete, the ceiling)**.** *If an L2's trust-minimization requires exits and verification from a DA layer whose security assumptions are comparable to the L1 assumptions (no privileged server, independent reconstruction), then replacing L1 DA with an external DA layer replaces C by C_{ext} but preserves the ceiling $T_{\text{L2}} \leq C_{\text{ext}}/s$ and preserves the information-flow bound $C_{\text{ext}} \leq \min(U_*^{\text{ext}}/k, D_*^{\text{ext}}/k)$.*

Proof. The DA bottleneck theorem (Theorem 1) and information-flow DA bound (Theorem 3) depend only on (i) exit-complete public information availability and (ii) information-flow constraints under decentralization. Those conditions still apply to the external DA layer by assumption. □

7 Global-Log Coupling: Congestion, Reorg Class, MEV, and Malware Surfaces

7.1 Congestion externalities are structural

Definition 9 (Non-interference indicator)**.** *For two transactions T_1, T_2, define*

$$\iota(T_1, T_2) = \begin{cases} 1 & \text{if the system forces } T_1, T_2 \text{ to serialize behind the same global resource,} \\ 0 & \text{if } T_1, T_2 \text{ can proceed independently.} \end{cases}$$

Definition 10 (Interference score for a step)**.** *Given N transactions $\{T_i\}_{i=1}^N$ active in a step, define the interference score*

$$\mathcal{I} := \sum_{1 \leq i < j \leq N} \iota(T_i, T_j).$$

Proposition 2 (Global DA implies systemic interference)**.** *In a global DA-coupled system, unrelated applications contend for the same DA budget C. In the worst case, $\mathcal{I} = \Theta(N^2)$, because all transactions share the same scarce global publication channel.*

Proof. Since all transactions consume the same global DA resource, an adversarial or high-load regime can force contention between an $\Omega(N)$-sized subset and another $\Omega(N)$-sized subset, making $\iota(T_i, T_j) = 1$ for $\Omega(N^2)$ pairs. Therefore $\mathcal{I} = \Theta(N^2)$ in the worst case. □

7.2 MEV is a byproduct of privileged ordering

Proposition 3 (MEV requires a privileged ordering domain)**.** *MEV (reordering, insertion, and censorship extraction by third parties) requires a privileged actor that can influence ordering within a shared ordering domain. If ordering is determined only at endpoints without a common mempool and without a privileged sequencer, systemic MEV collapses to local bilateral negotiation.*

Proof. MEV is defined by third-party control over ordering relative to honest users' intent. Remove the third-party ordering role and the shared mempool, and there is no systemic place to insert or reorder transactions across unrelated parties. Any residual advantage is local to participants, not globally extractable. □

7.3 Smart-contract malware vectors are a shared-execution artifact

Ethereum's world-computer framing implies a shared execution substrate where arbitrary code runs against shared state. That expands the adversarial surface: hidden or obfuscated behavior, dependency attacks, and composability exploitation. DSM's approach restricts shared logic to deterministic, locally verifiable commitments and policy anchors rather than arbitrary shared execution.

8 DSM Contrast: Relationship-Scoped Determinism Scales Additively

DSM deletes the single shared log. It replaces "everyone writes to one channel" with "each relationship is its own forward-only channel." This changes the scaling law: unrelated interactions do not contend.

Definition 11 (Relationship-scoped state)**.** *Each device maintains forward-only state for each relationship. A valid transition is accepted only if it advances the relationship's straight hash chain and matches the expected sparse Merkle root (where applicable). There is no global total order across all relationships.*

Definition 12 (Per-device verification capacity)**.** *Let device d have verification capacity C_d accepted transitions per step.*

Theorem 6 (Additive throughput under endpoint-local verification)**.** *Assume each accepted DSM transition is verified and applied only by its endpoints, and unrelated relationships are disjoint state components. Then DSM system throughput satisfies*

$$T_{\mathrm{DSM}} \leq \frac{1}{2} \sum_{d} C_d,$$

which scales linearly with the number of devices under bounded average device capacity.

Proof. Each transition consumes verification budget at exactly two endpoints. The total verification budget across devices is $\sum_d C_d$. Dividing by 2 yields the bound. □

Corollary 2 (Asymptotic separation)**.** *Let N be the number of devices/users. Under bounded average device capacity $\bar{C}$, DSM has $T_{\mathrm{DSM}} = \Theta(N)$, while global DA-coupled systems have $T_{\mathrm{L2}} = O(1)$ under stable decentralization constraints. Therefore*

$$\lim_{N \to \infty} \frac{T_{\mathrm{DSM}}}{T_{\mathrm{L2}}} = \infty.$$

Proof. Immediate from growth rates. □

9 Online vs Offline DSM Transfers and Why Recipient Signatures Are Not Required to Send Online

DSM supports both offline bilateral exchange and online "mailbox" transfer via storage nodes (B0x). The guarantees are the same at the cryptographic layer: forward-only commitments and endpoint verification. The difference is when the receiver learns the commitment and performs acceptance.

9.1 Core separation: why signature checks need not live in the core

Definition 13 (Core transition function)**.** *Let* Apply *be the deterministic state transition function that maps* $(\text{state}, \text{op})$ *to a new state or reject. This function is purely deterministic and rejects any transition that does not advance the forward-only state as required.*

Proposition 4 (Authorization as a precondition is sufficient)**.** *If endpoints enforce a precondition* $\mathsf{Auth}(\text{op}, \text{sender\_pk}) = \mathit{true}$ *before calling* Apply*, then the core need not embed signature verification to preserve authorization; the security property holds at the acceptance boundary.*

Proof. A transition affects state only if accepted by an endpoint. If the endpoint rejects any operation failing Auth, then no unauthorized operation reaches Apply. Embedding Auth inside Apply is an implementation choice, not a requirement for the security invariant; the invariant is enforced by the acceptance rule. □

9.2 Online send: sender finality without recipient signature

Definition 14 (Online transfer transcript)**.** *An online transfer from A to B consists of:*

1. *A constructs a canonical operation* op *containing a nonce n and all transfer semantics.*
2. *A signs* op*, producing σ_A.*
3. *A commits the transition locally (advancing its forward-only relationship state) and submits* (op, σ_A) *to B0x.*
4. *B later retrieves* (op, σ_A)*, verifies σ_A, checks nonce-replay constraints, and either accepts (advancing state) or rejects (no state advance).*

Theorem 7 (Sender finality in online transfer does not require recipient signature)**.** *If A commits a forward-only state transition that spends funds to B (i.e. advances A's relationship state in a way that makes the spend irreversible under the protocol), then recipient signature is not required for the statement: "A* cannot revert the send once committed."

Proof. Recipient signature would be relevant only if A's protocol permitted reverting the local state transition absent recipient confirmation. But DSM sender finality is defined by forward-only state advancement at A: once A advances its chain tip and records the spend, there is no protocol-valid inverse transition that restores the prior tip (by the forward-only acceptance rule and preimage resistance of the hash commitment linking tips). Therefore A cannot revert regardless of whether B signs anything. □

Corollary 3 (Receiver protection)**.** *Receiver signature is unnecessary for sender finality, but receiver verification is mandatory for receiver safety: B must verify σ_A and apply nonce/replay rules before accepting, otherwise B could accept invalid or replayed transfers.*

Proof. Acceptance is endpoint-local. If B does not verify authenticity and replay constraints, B can be induced to accept invalid operations. Therefore verification belongs at the receiver boundary. □

10 DSM Atomic Composability Without Global Ordering (Clockless)

Definition 15 (Deterministic tick and expiry)**.** *Let $\theta \in \mathbb{N}$ be a monotone deterministic tick embedded in each relationship chain state (a step counter, not wall-clock time). A commitment may specify an expiry tick θ_{exp}; any party rejects the commitment once its local relationship tick reaches $\theta \geq \theta_{\text{exp}}$.*

Definition 16 (Hash-locked k-party commitment)**.** *A k-party transaction graph has canonical hash $\mathcal{T}_{\text{hash}}$. Participant i publishes commitment*

$$c_i = H(\mathcal{T}_{\text{hash}} \,\|\, \theta_{\text{exp}} \,\|\, r_i \,\|\, \Delta_i)\,,$$

where Δ_i is i's local deterministic state delta and r_i is a reveal secret.

Theorem 8 (DSM atomicity without global ordering)**.** *Assume all parties verify the same $\mathcal{T}_{\text{hash}}$ and θ_{exp}.*

(i) If all reveals r_i are provided and verified before each party reaches its expiry tick, then all parties commit their local deltas.

(ii) If any reveal is missing for some party when that party reaches $\theta \geq \theta_{\text{exp}}$, that party deterministically aborts, and no party can produce a complete valid commit proof set. Thus the protocol enforces all-or-nothing semantics without a global log.

Proof. Commitments bind each party to the same transaction graph and expiry. A reveal opens the commitment to the specified delta, making acceptance a local deterministic rule. If any reveal is missing past expiry, no valid opening exists in the transcript for that commitment; hence no complete commit proof set exists. Abort is deterministic from $(\theta, \theta_{\text{exp}})$ and the missing opening, so atomicity holds without a global ordering domain. □

11 Concrete Parameterization (Per-Step, Not Wall-Clock)

This section gives per-step DA magnitudes to plug into theorems. These are not required for the impossibility proof.

Definition 17 (Blob DA magnitudes (Ethereum))**.** *EIP-4844 blobs are fixed at 131,072 bytes.*[4] *Let b be blobs/step. Then the raw blob DA per step is*

$$C_{\text{blob}} = b \cdot 131{,}072 \text{ bytes/step.}$$

After EIP-7691, blob target/max is (6,9) per block.[5] *Future dankssharding discussions commonly reference $b \approx 64$ as an upper regime.*[6]

Remark 1 (Per-step throughput bound from blob DA). For any assumed per-transition DA footprint s, Theorem 1 implies

$$T_{\text{L2}} \leq \frac{C}{s}.$$

If one substitutes $C = C_{\text{blob}}$ (treating blobspace as the dominant DA budget), then for $b = 64$ blobs/step,

$$C_{\text{blob}} = 64 \cdot 131{,}072 = 8{,}388{,}608 \text{ bytes/step}, \qquad T_{\text{L2}} \leq \frac{8{,}388{,}608}{s} \text{ transitions/step.}$$

All L2s combined share this bound. Increasing b increases the constant but does not change the asymptotic result that per-user service vanishes as $N \to \infty$.

[4] `https://eips.ethereum.org/EIPS/eip-4844`.

[5] `https://eips.ethereum.org/EIPS/eip-7691`.

[6] `https://notes.ethereum.org/@dankrad/new_sharding`.

12 Brand vs. Technical Role: "World Computer" Definition Laundering

Ethereum's public narrative still prominently uses the slogan "world computer" in official communications, even in recent years.[7] In parallel, core technical discourse increasingly frames Ethereum L1 as a "world ledger" / settlement base layer for L2s.[8]

This produces a definitional substitution:

Definition 18 (Literal "World Computer")**.** *A system is a literal world computer if it provides a trust-minimized execution substrate where average per-user service does not vanish under arbitrarily large adoption while preserving the system's decentralization constraints. Formally, there exists $\epsilon > 0$ such that for arbitrarily large N, the per-user service rate is at least ϵ (in per-step units), without introducing trusted operators.*

Definition 19 (Ecosystem "World Computer" (Marketing Definition))**.** *A system is an ecosystem world computer if the* ecosystem *(L1 + L2s + operators) can be described as "running the world's code", even if end-to-end trust-minimization and per-user service guarantees depend on shared bottlenecks or privileged operators.*

Remark 2 (Why the marketing definition is not a scalability claim)*.* The ecosystem definition is not a technical property: any federation of off-chain systems with a shared settlement layer can satisfy it by re-labeling execution as "part of the ecosystem." It therefore cannot be used to justify a literal world-computer scalability promise.

Theorem 9 (DA-impossibility persists under the ecosystem redefinition)**.** *If an ecosystem claims literal world-computer semantics (trust-minimized verification + unilateral exits, and non-vanishing per-user service), then the aggregate trust-minimized throughput of all L2s is bounded by the DA capacity C of the underlying DA layer via $T_{\mathrm{L2}} \leq C/s$, and under stable decentralization constraints $C = O(1)$ with respect to adoption. Therefore per-user service $T_{\mathrm{L2}}/N \to 0$ as $N \to \infty$, contradicting the literal world-computer requirement.*

Proof. The DA bottleneck is Theorem 1. Under DAS, the ceiling refines to Theorem 4 by substituting the information-flow DA bound (Theorem 3). Under stable decentralization constraints U_*, D_*, k, s do not scale with adoption, so T_{L2} is bounded independent of N. Dividing by N yields vanishing per-user service, contradicting the literal definition. □

Conclusion of this section. Ethereum can keep the brand slogan by expanding the meaning to "the ecosystem runs code somewhere", but the literal claim (planetary-scale trust-minimized compute with non-vanishing per-user service) fails for the same conserved-DA reason. DSM is designed to satisfy the literal requirement by removing the default global ordering/DA-coupled path: verification is endpoint-local, so unrelated interactions do not contend for one global publication channel.

13 Conclusion

Rollups and DAS do not make Ethereum a literal world computer. They change what is scarce: compute becomes cheap, but data availability becomes the conserved bottleneck. DAS reduces per-validator verification cost, but cannot eliminate the information-flow requirement

[7] For example, the Ethereum Foundation has described Ethereum as "humanity's shared world computer" (2025). `https://blog.ethereum.org/2025/04/28/ef-vision` The original launch framing also used "world computer" language (2015). `https://blog.ethereum.org/2015/07/30/ethereum-launches`

[8] See Vitalik Buterin, *Simplifying the L1* (2025): "Ethereum aims to be the world ledger". `https://vitalik.eth.limo/general/2025/05/03/simplel1.html`

that decentralization-eligible publishers upload and independent retrievers download the DA payload each step. Therefore aggregate trust-minimized throughput remains bounded by a constant under stable decentralization constraints, and average per-user service vanishes as adoption grows.

DSM delivers the original promise more completely: it deletes the single global ordering and DA bottleneck by making state relationship-scoped and forward-only, so unrelated interactions do not contend. The resulting scaling law is additive across devices, and the adversarial surfaces intrinsic to a global log (reorg class, systemic MEV, shared-execution malware vectors, and congestion externalities) are structurally sidestepped rather than merely mitigated.