Snowball to Snowstorm:

Rateless Reconciliation and Sampled Preference Voting
for Leaderless Pre-Block Set Agreement

Team Twerp

"The design supports letting users just be users."
— Satoshi Nakamoto, 29 July 2010

31 October 2026
Version 0.1

Abstract

Snowstorm lets cryptocurrency nodes reconcile and validate candidate transaction sets while miners search for the next block. It trades background bandwidth for confidence in those sets, aiming to improve node-hardware utilization and reduce processing and bandwidth bursts at block arrival. It adapts Snowball's repeated random-sampling approach from the Avalanche family [6] to transaction preferences and candidate-set roots. Nodes discover missing transactions and vote on preferences. Each root vote covers the whole candidate set, sharing its communication cost across the transactions represented. Each completed agreement cycle, called an epoch, freezes a nonempty batch; successive batches form a cumulative overlay. A miner builds from a frozen overlay; a receiver whose retained overlay reproduces the block commitment reuses its validation work. In the preferred order-independent Merklix path, a successful match leaves $O(1)$ announcement metadata and incremental post-arrival validation in the number of transactions, with header and coinbase sizes held fixed. Ordinary relay handles unmatched blocks, and block-validity rules are unchanged. Data-center simulations report preparation traffic, epoch turnover, receiver parking and optional miner template lag. Confidence accounting is conditional on a local response model; asynchronous security and matched cross-protocol cost comparisons remain open.

1 Preparing Blocks During the Block Interval

A block's discovery creates a deadline for work that could have begun earlier. Nodes already receive and validate transactions between blocks, but the arriving block still determines which transactions belong together. Receivers must identify that set, recover anything missing and complete the checks needed to accept the block. Snowstorm extends the preparation across the block interval: nodes reconcile candidate sets, validate compatible batches and retain their commitments as transactions arrive. A matching block can then identify a set whose transaction-set work is already complete.

The algorithm repeats four steps. Discover differences between peers' sets. Vote on individual transaction preferences. Freeze a batch when enough sampled peers support the same candidate-set root. Start the next batch immediately, preserving earlier work. Each cycle of forming and freezing a batch is an *epoch*; its boundary is a freeze decision, not a fixed elapsed time. Successive frozen batches form an overlay from which a miner can construct a block. Receivers retain earlier overlays too, so a miner need not choose their newest one for its block to match.

The resource tradeoff is deliberate: background communication buys repeated confidence observations and prepared transaction sets. The aim is better utilization of node hardware throughout the block interval, with less processing and bandwidth demand when a block arrives. Preparation proceeds without knowing when the next block will be found. Receivers reuse the prepared state on a match, check the header, coinbase and full block commitment, and use ordinary reconstruction and validation on a miss. These confidence signals could later be extended into an explicit finality mechanism, with additional protocol rules, participation assumptions and security analysis. The present design leaves block-validity rules unchanged.

The voting cost is shared across transactions. A root vote identifies a commitment to the whole candidate set rather than listing a separate vote for every transaction. At fixed sample size and polling cadence, this root-voting traffic does not grow proportionally with transactions per second (TPS). Larger batches amortize it across more transactions. Reconciliation and dissemination have separate costs: peers must communicate their set differences, exclusions and missing bodies. Total bandwidth can therefore increase with TPS even while the root-voting component remains approximately constant.

Our contributions are the connection between sampled preference voting and cumulative mining overlays, the distinction between item confidence and set-root confidence, and illustrative simulations of preparation and lookup policies. Snowstorm requires a reconciliation primitive with the interface in Section 4. We use Ribbit, whose coding construction is given in Appendix A; its field arithmetic and decoder are independent of the agreement algorithm. The detailed protocol and implementation contract remain available in the appendices.

2 Related Work

Compact blocks, specified by Bitcoin Improvement Proposal (BIP) 152, identify a mined block's transactions with short identifiers and fetch unresolved bodies [14]. Graphene reconciles the announced set using a Bloom filter and an invertible Bloom lookup table (IBLT) [16]. Erlay and BIP 330 reconcile transaction announcements between peers [17, 18]. Snowstorm adds agreement on prevalidated sets that a later block can identify.

Séchet's *Embrace the DAG* presentation frames pre-consensus as continuous reduction of differences between nodes and separates transaction membership from ordering [23]. Snowstorm develops that motivation through item and root votes, frozen batches and cumulative mining overlays.

Snowball supplies the repeated-sampling structure [6]; Bitcoin ABC's Avalanche post-consensus

polls on blocks after mining [19]. Snowstorm changes the decision rule and does not inherit Snow-family security guarantees [7, 8]. Ribbit builds on rateless IBLT reconciliation [9]. Appendix J discusses these relationships, other consensus settings, commitments and relay privacy.

3 State and Assumptions

The host supplies validated transaction bodies, conflict information, deterministic commitments, time, randomness and network delivery. Snowstorm's core is a deterministic state machine over those inputs. It targets geographically distributed data-center nodes; interface capacity and propagation delay are separate evaluation inputs.

This deployment model is consistent with Nakamoto's expectation of specialized operation at scale: "Those few nodes will be big server farms." [24]

Epoch boundaries. Each node starts epoch 1 on a new chain tip. Epoch k consists of repeated polling rounds to form the next nonempty batch. It ends when the node's candidate-set root reaches the confidence threshold and the body-admission conditions in Section 5.3 are satisfied. The node then seals the batch as E_k and immediately starts epoch $k+1$. The batch cannot change after sealing. Nodes can finish the same epoch at different times; there is no shared epoch clock or fixed number of rounds. A new chain tip ends the old context and restarts numbering at 1.

Admission and epoch assignment. The host mempool and Snowstorm track the same logical transaction admission. The host validates and admits an ordinary new transaction, then notifies Snowstorm through the serialized admission path. Snowstorm records it in an open, unassigned set U. This is neither recovery Pending nor assignment to the working epoch. At the next epoch cut, eligible open transactions are selected into the new epoch; ordinary arrivals after that cut continue into U without changing the working epoch or its root.

The epoch cut. Capture the complete eligible membership of U under the serialized mutation owner. Eligible transactions are admitted, nonarchived and compatible with the frozen parent; the selected set must be dependency-closed relative to that parent and conflict-free under the host's current selection policy. Transactions whose required ancestors are unavailable or excluded by a frozen conflict are not eligible. There is no age filter or arbitrary transaction-count prefix. Selected membership becomes the initial working candidate and leaves U; deferred membership stays open. An empty successor waits for eligible transactions and a separate cut event rather than freezing an empty batch. Ordinary admission itself never performs epoch assignment. Peers cutting at different times can select different initial candidates, which still require reconciliation and voting; the cut supplies no common wall-clock boundary.

Reconciliation recovery is different. A validated missing transaction may be associated with the revisable current epoch through that epoch's captured recovery context. If its body is already admitted, association changes its classification without admitting it again. Recovery Pending $\mathcal{P}$

tracks unadmitted recovery items awaiting bodies or admission; ordinary new admissions need not pass through it. A Pending item contributes no body to W. Epoch association must be checked independently of mempool admission, and any resulting candidate-root change resets root confidence.

Within the active epoch, A is its advertised identifier set, W is its compatible candidate body set, and $A \setminus \mathrm{IDs}(W)$ contains exclusions. The open set U is outside the epoch's advertised set and candidate root. Frozen epoch batches E_i form cumulative overlays

$$O_k = \bigcup_{i=1}^{k} E_i, \qquad O_0 = \varnothing.$$

The active root is $\rho(O_{k-1} \cup W)$ and the frozen-prefix root is $\rho(O_{k-1})$. Frozen membership is immutable until the next accepted tip. Appendix C specifies live stores, item lifetimes and host boundaries.

Figure 1 shows how frozen batches, their cumulative roots, the working epoch, open arrivals and recovery Pending fit together. Earlier batches stay in the commitment while new item voting concerns the active epoch.

Each round samples K distinct peers uniformly afresh from a compatible neighbourhood of size D, with $3 \leq K \leq D$. Adversaries may equivocate, fabricate identifiers, withhold replies or send malformed messages. The model assumes their fraction in each neighbourhood is at most $\beta_{\mathrm{adv}} < 1/3$ and assumes representative peer selection across the network. The core supplies no Sybil resistance. Sample selections can be fresh while peer preferences and responses remain correlated. The conditional confidence calculation below requires additional response assumptions, not just a bound on adversarial identities.

4 Set Reconciliation as an Interface

Let a Ribbit be an extensible encoding of a finite identifier set. Given a peer's encoding and a local reference set, a receiver attempts to recover their symmetric difference, including which side holds each recovered identifier. A short prefix can fail to decode; later polls can request more cells. Snowstorm uses the following interface properties.

1. Each encoding and its exclusions describe a fixed captured set. Differences reconstruct that advertised set against the corresponding saved reference, even if either node's live set subsequently changes.

2. A complete decode identifies the asserted difference. An incomplete decode leaves unrecovered items unknown; it must not turn them into absence votes. Any usable partial observations remain explicit.

3. Recovered identifiers do not authorize body admission. The host fetches and validates bodies, checks identifier bindings and enforces conflicts before including them in W.

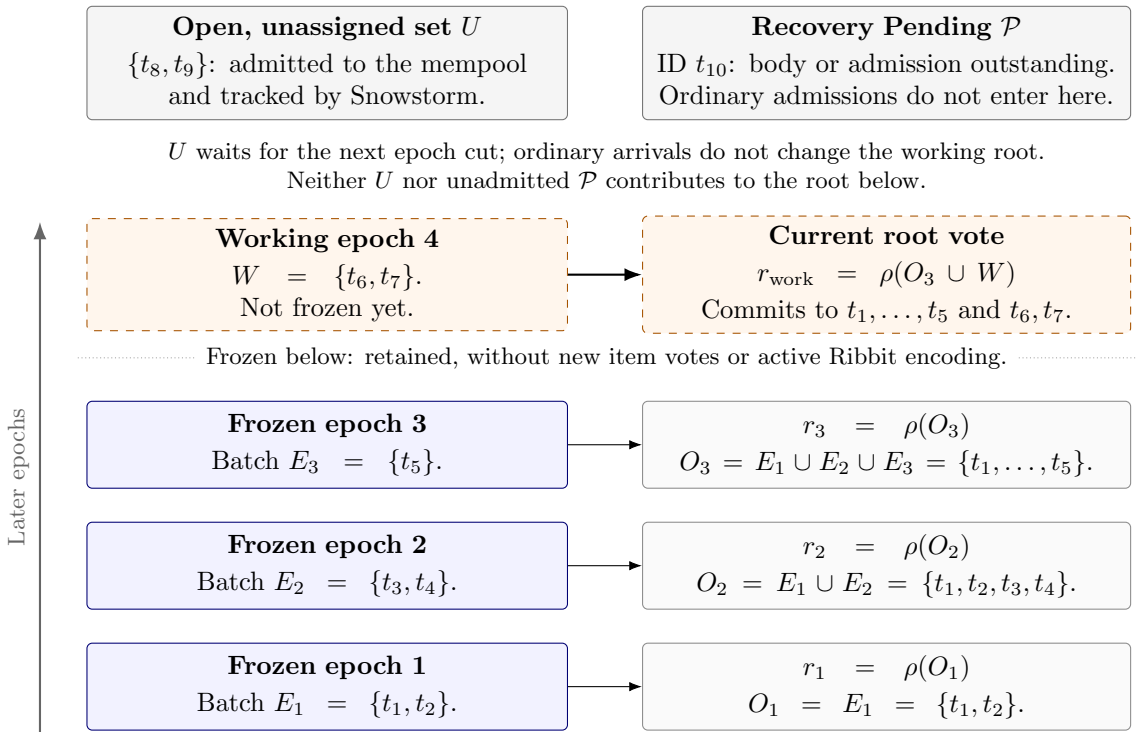


Figure 1: One node working on epoch 4. Ordinary newly admitted transactions enter the open set U, not the working epoch or recovery Pending $\mathcal{P}$. Each frozen root includes its row's batch and all earlier batches; the working root adds only W. Frozen transactions receive no new item votes. On freeze, W becomes E_4 and its root is retained as r_4; opening epoch 5 selects eligible open transactions into the new working set. Recovery of a missing current-epoch member is a separate, context-bound path. The layout shows commitment membership, not tree shape or transaction order; held exclusions are omitted.

These are the reconciliation semantics the agreement layer needs. They do not make an encoding a cryptographic proof of an honest peer's inventory. Ribbit uses finite syndromes and 64-bit identifiers, so collisions and malicious set assertions remain possible. Body validation and ordinary block fallback are necessary. Appendix A gives the construction, its limits and the within-reply peeling procedure.

A poll captures a tip, epoch, prefix, candidate root and reference state. A responder with the same prefix can return a compact root MATCH, a complete difference, or its own Ribbit when the poll's prefix cannot decode. A fallback Ribbit is already in that response and need not add a request/response cycle. An ahead responder can serve the requested historical epoch read-only; a behind responder abstains. Both endpoints fetch their own discovered bodies through the ordinary host path.

Only timely correlated responses from sampled peers supply votes. Inbound polls disseminate information but never vote. Complete replies identify the responder's asserted candidate set after exclusions. An incomplete reply supplies only explicitly recovered presence, absence or exclusion observations; other items abstain. Later decoding rescue can discover bodies but cannot add or upgrade the round's votes. Captures, Pending admission and reply validation are specified in Appendices A and K.

5 Preference, Confidence and Freeze

5.1 Moving a preference

Each nonfinal item has an IN or OUT preference and a cumulative counter for each side. If more than $K/2$ sampled peers vote for a side in a round, increment its counter once. Preference follows the larger counter and retains its previous value on a tie. A round with no strict majority changes neither counter. Abstentions supply no vote and do not reduce K. Preference is reversible until FINAL; confidence resets do not erase the majority counters.

Conflicts and body availability. The candidate W contains admitted bodies with effective preference IN. The host validates direct spent-input conflicts and performs any authorized replacement atomically. A FINAL-in or frozen winner constrains direct alternatives to effective OUT, without fabricating a statistical FINAL-out result. Frozen winners cannot be replaced within the same tip. A live FINAL-in Pending item vetoes freezing until its body is admitted or its lifetime ends. Preference is therefore not permission to bypass body validation. Appendix E gives the replacement and anomaly rules.

5.2 Qualifying a confidence run

Preference and confidence have different gates. For confidence, define

$$p_0 = \frac{1+\beta_{\text{adv}}}{2}, \qquad a_K = \min\{y : \Pr[\text{Binomial}(K, p_0) \geq y] < 1/2\}.$$

A side succeeds when its support y reaches a_K, still using the full sampled K. It contributes evidence

$$e_K(y) = -\ln \Pr[\text{Binomial}(K, p_0) \geq y].$$

Consecutive successes on the same side accumulate evidence. An opposite-side success starts a new run; a round without success clears the run. Changing K clears a nonterminal run. FINAL decisions are immutable for the item lifetime.

For independent binomial null draws Y_i, the length-specific threshold is the least attainable score satisfying

$$T_r(K; b) = \min \left\{ t : \Pr_{p_0} \left[\sum_{i=1}^{r} e_K(Y_i) \geq t,\ Y_i \geq a_K\ \forall i \right] \leq b\, 2^{-(r-2)} \right\}, \qquad r \geq 3.$$

If no score qualifies, reject every path at that length. Use $b = \varepsilon/2$ for each item side and $b = \varepsilon$ for the root. The distribution retains unconditional path probabilities, with inclusive tails and exact comparisons. A run qualifies when its evidence reaches the corresponding threshold; a single strong round cannot finalize an item or freeze a root. Appendix B gives the full rule, analytic counts and exact representation.

5.3 Freezing a set

Broad support for individual items does not imply that peers hold an identical set. A separate root vote tests $\rho(O_{k-1} \cup W)$. Only verified eligible MATCH replies to the current commitment contribute to its confidence run. Changing effective W membership or the candidate-body commitment resets that run immediately.

After round-end item updates and conflict processing, freeze when the root run qualifies, W is nonempty, and no FINAL-in item remains Pending. Atomically archive W as E_k, retain O_k and its root, then start epoch $k + 1$. Per-item FINAL is not a prerequisite for freezing. An empty set continues polling without emitting a batch. New epochs reset active tally state but preserve frozen bodies and ongoing host fetch obligations.

5.4 What the confidence accounting establishes

The null concerns support at most p_0 for an item side or root, not every disagreement between honest nodes. For a fixed prospective start, the geometric budgets sum to b. Under a valid conditional response model, a union bound over prospective starts gives error at most ε times the rounds polled, per item and separately for the root. This is a local conditional bound, not an established network convergence theorem.

Uniform sampling without replacement and live asynchronous replies must still be related to the binomial calibration. A bound on simultaneous supporters does not bound the union of peers that can support at their different response times. Representative peer selection and Snowball

metastability are also assumptions in the convergence argument [6]. The detailed accounting and open obligations are retained in Appendices B and G.8.

6 Overlays, Mining and Block Receipt

Successive epochs spread preparation across the block interval. Sealed batches retain completed validation work; ordinary later arrivals accumulate in U for the next epoch cut. Bodies are retained once across the live/archive union, while an index records cumulative overlay roots. A laggard can poll and fetch historical batches with a matching prefix. A genuine frozen-prefix split has no agreement repair rule; ordinary accepted-block handling resets the context.

Commitment compatibility. The miner and receiver must reproduce the same block transaction commitment from the frozen set and received coinbase. Merklix is preferred because it is order-independent and updates incrementally [2]; it is not Bitcoin's commitment. A chain-defined canonical ordering rule, or a participating miner's deterministic dependency-respecting order, provides a Merkle fallback. The order is derived from the cumulative set, not batch concatenation or arrival order. These fallbacks require ordering and root computation. Blocks using a different set or order use ordinary relay. Appendix D specifies construction and lookup.

The base GetBlockTemplate (GBT) path selects the latest frozen non-coinbase overlay. On block receipt, the host attempts snap lookup before clearing old-tip state, considering every retained overlay. A match selects prevalidated bodies only if the full block commitment reproduces correctly. A miss uses ordinary reconstruction and validation. After acceptance, block-included transactions are removed, old-tip rounds and fetch generations are cancelled, and surviving held items start a fresh context.

Once a matching overlay is identified, the preferred Merklix path uses $O(1)$ announcement metadata and incremental post-arrival validation in the number of transactions, with fixed header and coinbase sizes. This moves transaction-set work off the arrival path; it does not eliminate body storage or pre-discovery work. Lookup cost also depends on the number of retained overlays. Merkle fallbacks have additional ordering and commitment costs.

6.1 Receiver parking and miner template lag

A receiver can hold every block body but still be finishing the matching epoch. An optional host policy parks that block, continues parent-tip agreement and retries lookup when an overlay freezes. At the wait bound, ordinary reconstruction proceeds. A completed reconstruction is not retroactively counted as a delayed snap. The host must specify competing-block handling and actual tip transitions.

A miner can instead choose an older frozen overlay, trading freshness for immediate receiver coverage. This is optional template selection, not a validity rule. Omitted transactions may wait

until a later block; their confirmation delay is not merely the template's age. Neither policy changes voting. The evaluation distinguishes immediate lookup, delayed lookup and reconstruction.

7 Sizing and Scheduling

Voting traffic versus reconciliation traffic. The commitment and fixed root-vote fields have the same size whether a candidate batch contains one transaction or hundreds. Their interval cost follows the number of sampled poll exchanges, not the number of transactions represented by each root. With K peers and L rounds, those fixed fields cost $O(KL)$ bytes per poller, and their cost per represented transaction falls as the batch grows. Larger differences or harder agreement can change L and the polling cadence. Ribbit cells, exclusions, announcements and body transfers must be accounted for separately; this amortization is not a claim of constant total bandwidth or logarithmic scaling with transaction rate.

The base implementation adapts Ribbit prefixes from earlier responses and resets active sizing at epoch rollover. A round ends when all selected peers have answered or its captured timeout expires. The timeout uses four times the a_K-th sampled reply-time estimate, never shrinks on the same tip, and backs off when outstanding responses could have supplied root success. Appendix F gives the exact sizing, history and timer rules.

Experimental policies. The evaluated combined candidate excludes received, validated epoch-mismatch abstentions from the backoff's outstanding count, while keeping the voting denominator K unchanged. It also encodes exclusion identifiers losslessly, using bitmap or ranks against a captured reconstructed set on complete differences, and Elias–Fano encoding on fallback replies. Positional encoding requires coherent saved snapshots. These policies extend the base contract and remain distinct from the agreement algorithm. Small initial or minimum Ribbits and ending every fully terminal round immediately are not evaluated in the measurements reported here.

8 Evaluation

Models covered. The preserved continuous-arrival simulator allows ordinary admissions into the active epoch's tracked set. Tables 1 and 3 describe that earlier model, rather than the explicit open-set ownership rule in Section 3. Section 8.1 separately reports a preliminary sweep implementing open admissions. The detailed historical contract is retained in the appendices for reproducibility.

The deployment study simulates 500 geographically distributed nodes, degree 128, $K = 32$, announcement fanout eight and five seeds per configuration. Continuous Poisson arrivals average seven transactions per second over a ten-minute mining interval. Three initial divergence workloads differ by about 2.21%, 4.22% and 6.23%. Nodes share ingress and egress capacity; propagation spans five regions. Primary data-center runs have reliable delivery without injected loss. Transaction bodies are 250 bytes, and validation is modeled with a one-millisecond delay, not executed by a production chain implementation.

That fixed validation delay is not a CPU-work profile. The suite supplies no matched peak-versus-mean processing or bandwidth comparison and cannot establish a provisioning reduction. Such a comparison must include the baseline host's existing validation caches as well as block reconstruction and state updates.

The separate adversarial stress tests in Appendix G partly exceed the assumed local adversarial bound: at 20% global adversaries, 899 of 2000 honest node-height instances have more than 20% adversarial neighbours, reaching 31.25%. Their convergence results mix neighbourhoods inside and outside the model; they cannot verify its conditional security bound. The primary data-center measurements are honest-network performance experiments.

Earlier data-center runs of the combined deadline/exclusion candidate freeze the first epoch in about six rounds and later epochs in a median five rounds, with later median times of 1.55–1.63 seconds. Those runs contain a subsequently found cumulative round cap and are retained only as historical measurements in Appendix G.4. The corrected ten-minute suite applies the cap per epoch. Table 1 charges all its pre-mining polling, including unfinished epochs. Receiver observations share five trial histories; they are not thousands of independent experiments.

Initial difference	Epochs/node	Recon. MB/node	Total MB/node	Miner overlay txs
2.21%	359.27	44.209	49.975	5002.0
4.22%	366.17	44.035	49.830	4996.6
6.23%	356.82	44.427	50.252	4999.4

Table 1: Historical ten-minute pre-mining results after correction of the cumulative round cap, with five seeds and 2500 node-height instances per row. Epoch counts and communication are per-node means; transaction counts describe the selected miner's cumulative overlay. The model boundary is stated at the start of this section.

These counters measure the cost of preparing sets throughout the interval. The fixed-rate deployment table is distinct from the TPS sweep below. Total preparation traffic must not be identified with the fixed root-voting component described in Section 7. The computed comparisons normalize selected accounting terms; a matched implementation comparison must also reproduce dissemination, block selection and missing-body service.

8.1 Transaction rate and computed cost comparisons

A separate preliminary sweep implements open admissions U at 1–250 TPS, with the same 500-node, degree-128, $K = 32$, fanout-eight, 1 Gbps model. Table 2 reports 600-second pre-mining traffic, including unfinished epochs. These runs precede the minimum-prefix and fallback-sizing repairs. Enlarged-universe paths use estimated exclusion-encoding lengths rather than complete wire serialization; the figures are simulator cost-model results pending that implementation check.

Poll framing plus MATCH traffic rises from 22.22 to 28.00 MB between 1 and 100 TPS, rather than increasing in proportion to the hundredfold offered rate. From 3 to 250 TPS it remains near 27–30 MB. This illustrates the shared root-voting cost: increasing batch membership does not require

TPS	Seeds	Framing + MATCH	Reconciliation	Total
1	5	22.22	32.11	35.81
3	5	26.91	40.08	44.47
7	5	27.73	47.59	53.39
25	5	28.33	64.28	76.34
100	3	28.00	156.81	195.09
250	3	29.58	447.80	538.65

Table 2: Preliminary corrected-admission sweep: mean MB (10^6 bytes) sent plus received per node over 600 seconds. Framing + MATCH is $c_1 + c_3$, a fixed-message-cost indicator rather than an isolated measurement of every voting field; reconciliation is $c_1 + \cdots + c_6$. Bodies initially supplied to most nodes are outside these interval counters.

another vote entry per transaction. Other traffic grows with reconciliation and dissemination work. An unweighted fit to the five positive-rate means from 1 to 100 TPS is

$$B_{\text{total}}(r) \approx 38.41 + 1.5665r \quad \text{MB per node per 600 seconds}, \qquad R^2 = 0.9979.$$

This is a range-specific affine fit, not a proportional voting-cost law. At 250 TPS the measured model total is 538.65 MB, exceeding its 430.04 MB extrapolation. The fitted intercept is not an idle-load measurement: at zero continuous arrivals the simulator finishes its initial work and reports 9.19 MB.

An analytical full-degree announcement model plus compact-block relay gives a traffic crossover near 18 TPS after normalizing body distribution for the complete fixture, or near 27 TPS when auxiliary initial transactions are omitted from both models. Appendix G.5 gives the accounting and its assumptions. These are calculated intersections with an explicit relay model, not measured Bitcoin Core break-even points. A separate explicit-item polling budget with 80 peer responses per transaction gives approximately 11.52 KB/transaction of network-average sent-plus-received item fields before shared framing and signatures. This compares how voting is represented; it does not equate confidence guarantees.

8.2 Immediate and delayed lookup

The corrected audit finds no overlooked existing matching root or flagged commitment divergence in the tested suite. Some receivers have every block body but have not yet frozen the miner's newly selected overlay. Let d be a receiver's current round duration. The tested parked policy waits $d' = \max(250\,\text{ms}, d)$ or $3d'$ before fallback; the floor is an implementation choice. Table 3 reports operational acceptance counts.

Immediate coverage depends on when mining falls within the epoch cycle. In the 2.21% workload, three of the five miners select overlays frozen only 14, 100 and 179 ms before mining, with immediate coverage of 74.4%, 79.8% and 72.2%, respectively. The other two trials have full immediate coverage. Five block discoveries at a fixed mining time do not establish a monotonic relationship between initial divergence and snap coverage.

Initial difference	Immediate	By d'	By $3d'$	One-epoch lag	Deferred txs
2.21%	85.28%	99.84%	99.84%	100.00%	12.6
4.22%	95.40%	99.28%	99.96%	100.00%	10.4
6.23%	95.92%	99.88%	99.88%	100.00%	15.2

Table 3: Historical snap audit after correction of the cumulative round cap, with five trials and 2500 receiver instances per policy and workload. The d' and $3d'$ columns include immediate and delayed acceptance; one-epoch miner lag is a separate policy with immediate lookup. Deferred transactions are the mean difference from the miner's latest overlay.

For one-duration parking, pooled median delayed waits are 58, 89 and 60 ms across the three workloads. A one-epoch-older template matches immediately at every tested receiver while omitting about 0.21–0.30% of the roughly 5000 transactions in the latest overlay. Observed coverage is not a guarantee for other workloads or adversarial conditions.

The acceptance model stops new parent-tip polling on acceptance but can still serve old-tip events. It does not implement a full production tip transition; fallback completion uses geographic RTTs rather than a queued block-relay exchange. Consequently these are lookup and modeled acceptance measurements, not production validation or end-to-end propagation timings. Appendix G retains the methods, earlier stress tests, full tables and limitations.

9 Limitations and Remaining Work

The confidence accounting assumes a valid conditional response model and a bound on adversarial peers in each sampled neighbourhood. Extending that accounting to live asynchronous replies and network convergence remains open. Partitions or nonrepresentative peer selection can produce different frozen sets; ordinary block validity remains the authority.

Separating U prevents ordinary arrivals from changing the working root, but recovery can still revise W and restart confidence. Repeated disclosure or withholding of valid current-epoch members can therefore delay agreement. Fabricated identifiers cannot enter W without validated body admission and valid epoch association. With no unsolicited pushes, open arrivals become visible to epoch reconciliation only after a cut, so discovery can be delayed while an epoch remains unfinished.

Short-ID collisions, body availability and resource limits retain the consequences described in Appendix H. The principal measurement gap is a matched implementation comparison of total costs and block-arrival peaks against existing relay with validation caches. The preliminary open-admission sweep and analytical cost intersections do not establish a provisioning benefit. Additional implementation details and research directions are retained in the appendices.

10 Conclusion

Snowstorm uses the interval between blocks to prepare compatible transaction sets and retain their validation work. Sampled preferences align items, a root vote freezes batches, and cumulative overlays connect that work to mining and block receipt. A matching block can reuse the prepared set; a miss follows ordinary relay. Root votes share their cost across the candidate set, while reconciliation communicates differences between peers. The preliminary sweep and computed comparisons quantify this tradeoff under explicit workload and relay assumptions. Asynchronous security and production comparisons with compact-block reconstruction and cached validation remain open. Ribbit supplies one reconciliation implementation, but its decoder is separate from the agreement algorithm.

References

[1] J. Vermorel, A. Séchet, S. Chancellor, and T. van der Wansem, "Canonical Transaction Ordering for Bitcoin," June 12, 2018.

[2] J. Vermorel, A. Séchet, S. Chancellor, and J. Cox, "Merklix Tree for Bitcoin," July 2018.

[3] Y. Minsky, A. Trachtenberg, and R. Zippel, "Set Reconciliation with Nearly Optimal Communication Complexity," *IEEE Trans. Inf. Theory*, vol. 49, no. 9, pp. 2213–2218, Sept. 2003.

[4] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data," *SIAM J. Comput.*, vol. 38, no. 1, pp. 97–139, 2008 (preliminary version in *EUROCRYPT 2004*).

[5] M. T. Goodrich and M. Mitzenmacher, "Invertible Bloom Lookup Tables," in *Proc. 49th Annual Allerton Conf. on Communication, Control, and Computing*, pp. 792–799, Sept. 2011.

[6] Team Rocket, M. Yin, K. Sekniqi, R. van Renesse, and E. G. Sirer, "Scalable and Probabilistic Leaderless BFT Consensus through Metastability," 2019, revised 2020. https://arxiv.org/abs/1906.08936.

[7] I. Amores-Sesar, C. Cachin, and E. Tedeschi, "When Is Spring Coming? A Security Analysis of Avalanche Consensus," in *OPODIS 2022*, LIPIcs, vol. 253, pp. 10:1–10:22, 2023. https://doi.org/10.4230/LIPIcs.OPODIS.2022.10.

[8] A. Buchwald, S. Buttolph, A. Lewis-Pye, P. O'Grady, and K. Sekniqi, "Frosty: Bringing Strong Liveness Guarantees to the Snow Family of Consensus Protocols," 2024. https://arxiv.org/abs/2404.14250.

[9] L. Yang, Y. Gilad, and M. Alizadeh, "Practical Rateless Set Reconciliation," in *Proc. ACM SIGCOMM 2024*, pp. 595–612, Aug. 2024, doi:10.1145/3651890.3672219.

[10] R. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking, "Randomized Rumor Spreading," in *Proc. 41st Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, pp. 565–574, 2000.

[11] D. Eppstein, M. T. Goodrich, F. Uyeda, and G. Varghese, "What's the Difference? Efficient Set Reconciliation without Prior Context," in *Proc. ACM SIGCOMM 2011*, pp. 218–229, Aug. 2011.

[12] T. Itoh and S. Tsujii, "A Fast Algorithm for Computing Multiplicative Inverses in GF(2^m) Using Normal Bases," *Information and Computation*, vol. 78, no. 3, pp. 171–177, 1988.

[13] P. L. Montgomery, "Speeding the Pollard and Elliptic Curve Methods of Factorization," *Mathematics of Computation*, vol. 48, no. 177, pp. 243–264, 1987.

[14] M. Corallo, "BIP 152: Compact Block Relay," 2016. https://bips.dev/152/.

[15] Bitcoin ABC, "Avalanche: Network Messages," protocol documentation, accessed 31 October 2026. https://github.com/Bitcoin-ABC/bitcoin-abc/blob/master/doc/avalanche.md.

[16] A. P. Ozisik, G. Andresen, G. Bissias, A. Houmansadr, and B. N. Levine, "Graphene: A New Protocol for Block Propagation Using Set Reconciliation," technical report. https://people.cs.georgetown.edu/~clay/classes/fall2017/835/papers/Graphene_A_New_Protocol_for_Block_Propagation_Using_Set_Reconciliation.pdf.

[17] G. Naumenko, G. Maxwell, P. Wuille, A. Fedorova, and I. Beschastnikh, "Bandwidth-Efficient Transaction Relay for Bitcoin," 2019. https://arxiv.org/abs/1905.10518.

[18] G. Naumenko and P. Wuille, "BIP 330: Transaction Announcements Reconciliation," 2019. https://bips.dev/330/.

[19] Bitcoin ABC, "eCash Avalanche Development Update: The Road to Post Consensus," May 2, 2022. https://www.bitcoinabc.org/2022-05-02-avalanche-development-update/.

[20] K. Babel, A. Chursin, G. Danezis, A. Kichidis, L. Kokoris-Kogias, A. Koshy, A. Sonnino, and M. Tian, "Mysticeti: Reaching the Latency Limits with Uncertified DAGs," *NDSS*, 2025. https://doi.org/10.14722/ndss.2025.240929.

[21] G. Fanti et al., "Dandelion++: Lightweight Cryptocurrency Networking with Formal Anonymity Guarantees," 2018. https://arxiv.org/abs/1805.11060.

[22] M. D. Penrose, "The Strong Giant in a Random Digraph," *Journal of Applied Probability*, vol. 53, no. 1, 2016. https://arxiv.org/abs/1409.4371.

[23] A. Séchet, "Embrace the DAG," presentation at Satoshi's Vision Instant Transactions Workshop, Gargnano, Italy, October 18, 2018. https://www.youtube.com/watch?v=9PygO-B1o6w.

[24] S. Nakamoto, "Re: Scalability and transaction rate," BitcoinTalk, July 29, 2010. https://satoshi.nakamotoinstitute.org/posts/bitcointalk/287/.

Technical Appendices

The construction, detailed rules and host contract below preserve the design used by the reported simulations. Their admission lifecycle predates the explicit open, unassigned set in Section 3: an ordinary admission could join active epoch tracking. These historical admission rules must be reconciled with the open-set ownership requirement before they can serve as its implementation contract. The reported measurements have not been rerun under that separation.

A Ribbit Construction and Reconciliation Details

A.1 Ribbit coding layer

We call this BCH-cell rateless invertible Bloom lookup table a Ribbit (BCH-RIBLT).

A.1.1 Field and short identifiers

Let $\mathbb{F} = \mathrm{GF}(2^{64}) \cong \mathbb{F}_2[z]/(z^{64} + z^4 + z^3 + z + 1)$, with addition written $\oplus$. Here, GF denotes a Galois field. Multiplication uses carry-less multiply instructions where available and a table-based fallback otherwise; inversion uses the Itoh–Tsujii method [12], and many inversions are batched with Montgomery's trick [13], which replaces k inversions by one inversion and $3(k-1)$ multiplications.

Each transaction has a short identifier $x = \text{SipHash-2-4}_{k_0,k_1}(tx_hash) \in \mathbb{F}^\times$. The host supplies the key (k_0, k_1) for each chain tip; *tx_hash* is the 32-byte hash of the full serialized transaction, and 0 is mapped to 1. Different transactions can collide. The core keys its set by short identifier and retains every admitted body under that identifier. Identifier agreement therefore leaves collision-body differences for host validation and ordinary block relay.

A.1.2 Cells

For a finite set $A \subset \mathbb{F}^\times$ define the odd power sums

$$s_m(A) = \bigoplus_{x \in A} x^m, \qquad m \in \{1, 3, 5\}.$$

A *cell* is the tuple (c, s_1, s_3, s_5) of a signed 32-bit count and three syndromes, serialized little-endian in 28 bytes. Counts must remain within $\pm(2^{31} - 1)$; overflow is an error and never wraps. Adding item x with sign $\sigma \in \{-1, +1\}$ sets $c \leftarrow c + \sigma$ and $s_m \leftarrow s_m \oplus x^m$ for $m \in \{1, 3, 5\}$. In characteristic two, $(u \oplus v)^2 = u^2 \oplus v^2$, so the even power sums follow from the odd ones: $s_2 = s_1^2$, $s_4 = s_1^4$, $s_6 = s_3^2$. A cell therefore determines the six consecutive syndromes $s_1, \ldots, s_6$ of its item set.

A.1.3 Coding schedule

The outer code is the rateless IBLT of Yang et al. [9] in an integer formulation. The Ribbit schedule provides a fountain of deterministically indexed cells, allowing the requested prefix to grow within

the protocol's symbol and message limits without selecting a fixed reconciliation capacity. Each item x is included in symbol 0, so symbol 0 contains every item and its count equals the set size. Subsequent inclusions are generated from a SplitMix64 stream seeded with x: one output u (its low 32 bits) gives $s = 2^{32} - u$, and from an included index i the next included index is the least integer $j > i$ with

$$s\,(j+1)(j+2) \;\geq\; 2^{32}\,(i+1)(i+2),$$

computed exactly with 128-bit integer arithmetic and no floating point. In the continuous idealization the probability of skipping past j is $(i+1)(i+2)/((j+1)(j+2))$, which gives marginal inclusion probability

$$p_j = \frac{2}{j+2}$$

at index j, the schedule of Yang et al. Because the schedule depends only on x, a sender and a receiver place a common item in the same symbols. The receiver uses its local reference set *Ref* as the *reconciliation baseline*: it forms the *residual* at index i as the received cell minus the encoding of *Ref* at index i; common items cancel, and the residual is the encoding, with signs, of the symmetric difference $U = U^+ \cup U^-$, where U^+ holds items the sender has and the receiver lacks ($\sigma = +1$) and U^- the reverse ($\sigma = -1$). A sender updates its encoding for every gain or loss in advertised A; body-only bucket changes do not alter the Ribbit encoding.

A.1.4 Cell decoding

A residual cell at index i is classified as follows.

- **Zero.** The cell is empty if $c = 0$ and $s_1 = s_3 = s_5 = 0$.
- **One.** The cell decodes to the single item (x, σ) if $|c| = 1$, $s_1 \neq 0$, $s_3 = s_1^3$, $s_5 = s_1^5$, the schedule of $x = s_1$ includes index i, and $\sigma = c$ agrees with membership: $\sigma = -1$ if and only if $x \in \mathit{Ref}$.
- **Two.** The cell decodes to the two items (x_1, σ_1) and (x_2, σ_2) if $c \in \{-2, 0, +2\}$, $s_1 \neq 0$ and the following succeeds. For distinct x_1, x_2 with $x_1 \oplus x_2 = s_1$, the identity $x_1^3 \oplus x_2^3 = s_1(s_1^2 \oplus x_1 x_2)$ gives

  $$x_1 x_2 = \frac{s_3}{s_1} \oplus s_1^2,$$

  so x_1, x_2 are the roots of $X^2 \oplus s_1 X \oplus x_1 x_2$. Substituting $X = s_1 u$ gives

  $$u^2 \oplus u = \delta, \qquad \delta = 1 \oplus \frac{s_3}{s_1^3}.$$

  This equation has a solution in $\mathbb{F}$ if and only if the absolute trace $\mathrm{Tr}(\delta) = 0$. Then $x_1 = s_1 u$ and $x_2 = x_1 \oplus s_1$. The decoder requires two distinct non-zero roots, $s_5 = x_1^5 \oplus x_2^5$, schedule inclusion of both at index i, and signs assigned by membership in *Ref* that sum to c.
- **Undecodable.** In every other case the cell is undecodable.

Solving the quadratic. The familiar half-trace $\sum_{k=0}^{(m-1)/2} \delta^{2^{2k}}$ solves $u^2 \oplus u = \delta$ only when the extension degree m is odd; here $m = 64$ is even, the half-trace is not a solution formula, and using it is an error. Snowstorm uses the standard construction for even degree, which requires a fixed element θ of trace one; it fixes $\theta = z^{61}$, evaluates the construction with byte-wise tables, and verifies $u^2 \oplus u = \delta$ before returning.

Finite syndromes are not an injective commitment to arbitrary large sets. A malicious sender can construct algebraically consistent cells for chosen identifiers; only fetched and host-validated bodies authorize admission.

A.1.5 Peeling

Within one peer's Ribbit, shared transaction identifiers link the cells. Recovering an identifier removes its contribution from every corresponding residual cell and can expose further decodable cells. Retained-reply rescue in Appendix K.1 can retry a reply after the shared reference changes, but cannot add or upgrade votes.

The receiver stores the residuals of one reply and peels them with a worklist. Each pass takes a snapshot of the cells changed since the previous pass, attempts singleton decodes before pair decodes, and computes the inverses s_1^{-3} needed by the pair candidates in one batch. Each decoded item is subtracted exactly once from every stored residual whose index its schedule includes, which may make further cells decodable; a Two result removes both items atomically. The receiver keeps a per-pass ledger of claimed items: if the same item is claimed twice while processing one reply, the reply is inconsistent and is discarded for the round without a report (Section K.8). Peeling runs until the worklist is empty, with no iteration cap; storage limits are separate typed errors. Decoding *succeeds* when every residual is zero. Inverting zero is an error; batch inversion rejects a batch containing zero atomically. The quadratic solver returns zero for a zero input and verifies every returned solution.

A.2 Polls and replies

A poll is a one-shot exchange scoped to a tip, epoch and frozen-prefix root. It carries the poller's overlay root and Ribbit cells for its advertised set. A prefix request supplies consecutive cells; an initial sparse probe samples the deterministic schedule. The responder keeps no reconciliation session. Figure 2 shows the three reply paths, and Appendix K.1 specifies their wire formats and sizes.

Epoch selection at the responder. A responder at the requested epoch selects its active state. An ahead responder selects that epoch's retained frozen batch as its raw and candidate set, with no exclusions, its cumulative overlay root as root vote and the preceding overlay root as prefix. A behind responder abstains. A POLL with a different frozen-prefix root from the selected state receives no reply, decode or fetch. Historical serving is read-only and creates no Pending or local tally changes.

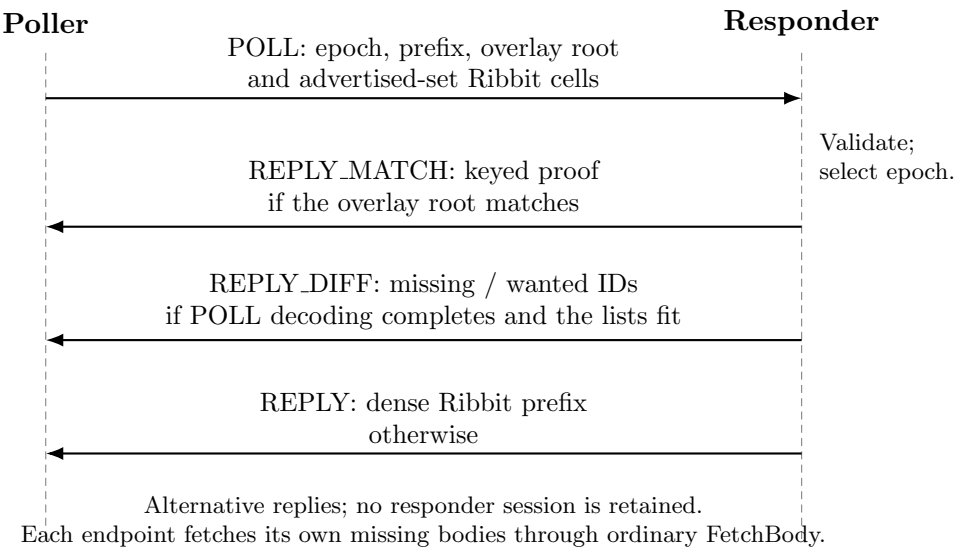


Figure 2: A one-shot Ribbit reconciliation poll and its alternative replies under a comparable epoch prefix. Historical service is read-only; active service can discover bodies in both directions. Inbound polls never vote, and decoding never admits a body.

Matching and differing sets. After structural validation, a matching overlay root under the same prefix yields REPLY_MATCH without peeling. Its nonce-keyed proof covers candidate identifiers; it is algebraic set evidence, not authentication of bodies or identities. Otherwise the responder peels the POLL against its selected raw set, never Pending. A complete decode yields REPLY_DIFF with responder-only `missing` IDs and poller-only `wanted` IDs. If decoding is incomplete or the complete lists exceed the message limit, the responder sends REPLY with its own dense Ribbit prefix. The prefix can grow on later polls, subject to the cap.

For DIFF, with captured raw set A_{cap} and peer exclusions X_j, the poller reconstructs the asserted candidate IDs as

$$\widehat{W}_j = \big((A_{\text{cap}} \setminus \mathit{wanted}) \cup \mathit{missing}\big) \setminus X_j.$$

Starting from raw captured IDs includes locally excluded items that the peer prefers IN. The poller checks the lists against that capture before any vote or fetch effect; validation details are in Appendix K.1. A captured Pending ID outside A_{cap} is present at the peer exactly when listed in `missing`. Post-capture items receive no observation.

Both endpoints use ordinary FetchBody for their positive discoveries. The active responder fetches from the POLL's sender, and the poller fetches `missing` IDs from the response sender. The `wanted` list is evidence of absence, not a body request or body-send action. Ordinary incomplete inbound positives can also start bounded fetches; an explicit contradiction discards all effects of that attempt before fallback. Historical replies never initiate a fetch. Inbound polls supply dissemination only: the receiver did not randomly select their senders, so they provide no votes or sizing samples and their cells are neither retained nor rescued.

Captured votes and fallback decoding. A round captures one shared state for its polls. MATCH verifies against its stored expected proof even if current membership changes; eligible captured item votes survive, but an old commitment cannot score a different current root. Each fallback attempt captures a fixed current *Ref* and peels received Ribbit cells minus its encoding. A complete decode reconstructs

$$\widehat{W}_j = \big((\mathit{Ref} \cup U_j^+) \setminus U_j^-\big) \setminus X_j.$$

Arrival observations determine that round's votes. Existing within-round retained-reply maintenance and rescue can discover donors and bodies but never create or upgrade observations. The residual invariant, capture lifetimes and release rules are retained in Appendix K.1. Peeling operates within a single peer's Ribbit.

A.3 Bodies, Pending and admission

Decoding discovers an identifier. Admission requires the host to fetch its body, check the short-ID binding and validate the transaction. Pending tracks the unadmitted item while fetching, awaiting admission, waiting on policy or retaining a FINAL-in decision. Until admission, it contributes nothing to the node's candidate root, proof or advertised cells.

Fresh Pending starts OUT. Matching admission transfers its whole preference, confidence and FINAL state without forcing IN, so body latency need not erase eligible remote observations. Fresh admission without a live Pending state starts OUT; removal and readmission starts a new incarnation. Donor failures remove unsupported speculative membership, while consensus-invalid matching bodies are rejected for the tip lifetime. Policy refusal is not consensus invalidity: a policy-refused nonconflict item reaching FINAL-in can request admission subject only to consensus validity, and a validated conflict alternative can do so on effective preference IN (Appendix E).

A live FINAL-in Pending item vetoes freeze until actual admission, consensus-invalid disposition or a lifetime-ending reset. Other Pending items do not veto freeze, and rounds continue while bodies are transferred and validated. Preference or confidence alone never cancels a fetch or creates a rejection mark. Appendix K.2 retains the exact donor, token, authorization and callback contract.

B Full Voting Rules and Calibration

B.1 Votes are preferences

An item held but excluded by a peer receives NO from that peer. Pending and policy-refused unadmitted bodies are never in that node's W, proof, own cells, set size or exclusions. Remote admitted holders may nevertheless vote YES for a poller's captured Pending lifetime.

Only timely, correlated replies in the captured tip, epoch and frozen-prefix context supply observations. A structurally valid complete reply under a different prefix counts for timing but supplies no peel, fetch, sizing, item or root effects; a prefix mismatch is not peer misbehaviour.

Same-prefix replies with different active sets retain normal item voting. Observations apply at most once per selected peer and captured live item lifetime:

- A verified REPLY_MATCH votes YES for captured candidate IDs and NO for all other eligible captured admitted or Pending IDs.
- A valid REPLY_DIFF or fallback REPLY fully decoded on arrival votes according to its reconstructed peer candidates, including exclusions.
- A partially decoded arrival votes YES only for decoded-present nonexcluded items, NO for decoded-absent or excluded items, and abstains on all other items.
- Silent, late, invalid and undecoded observations abstain, and discoveries made only by rescue abstain for that round.

Items first appearing after capture receive no votes in that round. Matching Pending admission preserves the captured lifetime; removal/readmission or loss of support does not. Late replies are discarded entirely, with no fetch or sizing effect. Inbound polls never vote.

B.2 Majority preference

Each nonfinal item maintains $p_x \in \{\text{IN}, \text{OUT}\}$ and two cumulative counters $d_x[\text{IN}]$, $d_x[\text{OUT}]$. At round end, if $y_{x,s}$ of the K sampled peers vote for side s and $y_{x,s} > K/2$, increment $d_x[s]$ once. Preference follows the larger counter and keeps its previous value on a tie. If neither side has a strict majority, neither counter changes. Preference updates do not require confidence success, and confidence resets do not reset these counters.

Until FINAL, preference is reversible. The candidate set is exactly the held nonarchived bodies whose effective preference is IN; the direct conflict constraints in Appendix E can make an item effectively OUT. Fresh admissions start OUT unless they transfer a live Pending state. Items held at epoch start start IN. There is no evidence-mean entry gate.

B.3 Binomial evidence and the strict upper-median gate

Set

$$p_0 = \frac{1 + \beta_{\text{adv}}}{2}, \qquad X \sim \text{Binomial}(K, p_0).$$

Define the exact upper median and success threshold by

$$m_K = \min\{j : \Pr[X \le j] > 1/2\}, \qquad a_K = m_K + 1 = \min\{y : \Pr[X \ge y] < 1/2\}.$$

A confidence success for side s requires $y_{x,s} \ge a_K$, using the complete sampled K as denominator. The gate is a strict upper-median test, computed by exact rational comparisons. In particular, at $\beta_{\text{adv}} = 0$ with odd K, a null tail of exactly $1/2$ does not qualify. The gate is separate from the strict sample majority that moves preference.

A successful round contributes finite, nonnegative evidence

$$e_K(y) = -\ln H_K(y), \qquad H_K(y) = \Pr[\text{Binomial}(K, p_0) \geq y].$$

Retain one confidence side, summed evidence and consecutive-success count per item. A success on the same side extends the run. A success on the opposite side discards the old run and starts the new side at count 1 with its own evidence. A round without confidence success clears the side, evidence and count. Changing K clears a nonterminal confidence run before scoring the new round; majority counters and immutable FINAL decisions survive.

Exact null probabilities and score comparisons determine qualification; decimal logarithms are explanatory only. Appendix K.4 gives the finite exact representation and input interpretation.

B.4 Exact run-length thresholds

Allocate $b = \varepsilon/2$ to each item side and $b = \varepsilon$ to the root. For each K and $r \geq 3$, construct the exact null sub-probability distribution of

$$S_r = \sum_{i=1}^{r} e_K(Y_i), \qquad Y_i \overset{\text{iid}}{\sim} \text{Binomial}(K, p_0), \qquad Y_i \geq a_K \text{ for every } i.$$

Retain the original probability of each all-success path; do not rescale the successful paths to sum to one. Combine equal-score atoms. Among attainable scores choose the least t satisfying

$$T_r(K; b) = \min\left\{ t : \Pr_{p_0}[S_r \geq t,\ Y_i \geq a_K\ \forall i] \leq b\, 2^{-(r-2)} \right\}.$$

If none qualifies, use a reject-all sentinel at that length. Inclusive upper tails and exact probability-product comparisons are required. Figure 3 illustrates the run structure without substituting a numerical barrier. Compute the item and root tables once per K at startup. A run qualifies at the first $r \geq 3$ whose score reaches the corresponding T_r; a single strong round cannot finalize an item or freeze a root.

The strict gate gives null success probability below one half, so every sufficiently long all-success run eventually qualifies. Appendix K.4 gives the exact finite tail representation, including root runs whose emission is vetoed.

An item becomes FINAL on the reached confidence side at its first qualifying length, using $b = \varepsilon/2$. FINAL sets the terminal preference, is immutable for that lifetime and accepts no later tally updates. FINAL-out leaves W and the advertised A, sketches and exclusions. FINAL-in cannot leave through a tally change; an opposing FINAL or frozen conflict constraint is handled without rewriting the terminal decision. Host removal, epoch rollover or tip reset ends the active tally lifetime; frozen archive membership remains immutable until tip reset.

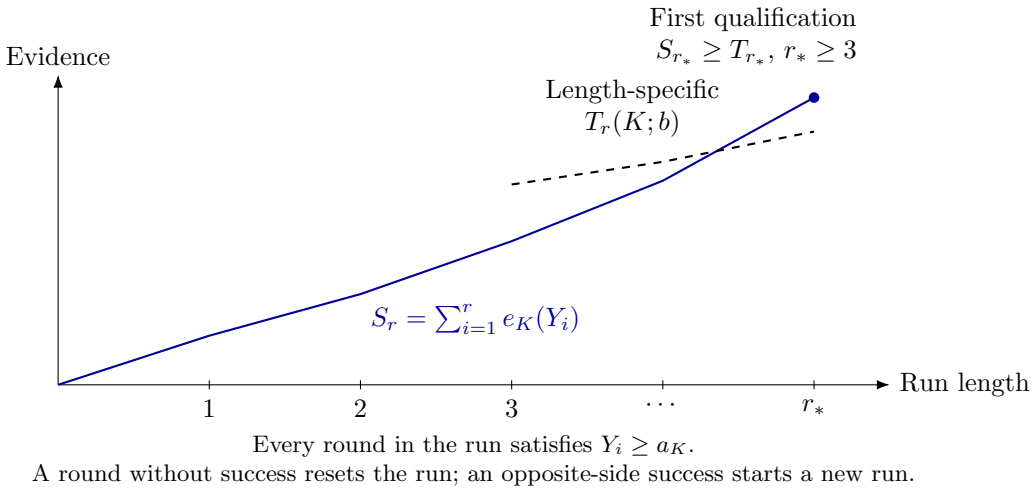


Figure 3: Schematic evidence accumulation, with uncalibrated vertical heights. The threshold depends on run length and is not a fixed horizontal barrier; some lengths may reject every attainable score. Exact qualification uses the null distribution, not this drawing.

B.5 Analytic reference counts

Table 4 gives first-qualification counts for exact $\beta_{\text{adv}} = 1/5$, $p_0 = 3/5$ and $\varepsilon = 10^{-9}$. Every round has the same fully eligible count $Y = \lceil qK \rceil$ for agreement fraction q, with item-side budget $\varepsilon/2$, no failed round and no change of K. The calculation depends on K; neither D nor network size enters the binomial formula. The requirement $K \leq D$ constrains sample selection but does not select a production crossover.

Every earlier eligible length is rejected by an exact rational lower bound above its spent budget, and the first accepted length is certified by an exact rational upper bound at or below that budget. Equal-score mass is included, and integer-power bounds certify score ordering. Thus the calculation certifies the discrete $T_r(K;\varepsilon/2)$ decisions without using rounded logarithms or simulation. The arithmetic method and its scope are described in Appendix K.5. Root qualification has a separate budget and additional freeze gates; these item counts do not predict a network's time to freeze.

B.6 Root vote and freeze

The root vote concerns $\rho(O_{k-1} \cup W)$. Only verified, eligible REPLY_MATCH responses to the current commitment count as root matches. The round is a root success when their count is at least a_K, and contributes the same $e_K(y)$ to a consecutive root run with thresholds $T_r(K;\varepsilon)$. A round without a root success clears the root run. Any change to effective W membership or its candidate-body commitment resets root evidence immediately. A changed captured commitment supplies no root confidence even though its eligible captured item observations remain usable.

After round-end item updates and conflict processing, freeze exactly when the root threshold predicate qualifies, $W \neq \varnothing$, and no FINAL-in item remains Pending. Per-item FINAL is not a

Table 4: Analytic consecutive rounds to item FINAL, not simulated waiting times or freeze-latency predictions. Values depend only on K, $p_0 = 0.6$ and $\varepsilon = 10^{-9}$; they are independent of D and of network size. Agreement percentages are rounded up to the integer observations shown; at small K this can make different percentages identical.

K	a_K	Y at 70/80/100%	70%	80%	100%
4	3	3 / 4 / 4	394	15	15
8	6	6 / 7 / 8	44	26	6
16	11	12 / 13 / 16	45	22	3
32	20	23 / 26 / 32	72	9	3
64	39	45 / 52 / 64	33	4	3
125	76	88 / 100 / 125	10	3	3
128	78	90 / 103 / 128	10	3	3
500	301	350 / 400 / 500	3	3	3

At $K = 4$, 70% rounds up to three of four votes, yielding 394 consecutive rounds to FINAL. The row $K = 125 = \lceil 500/4 \rceil$ illustrates a possible sample choice at $D = 500$, not a fixed sampling rule.

freeze gate. Atomically archive W as epoch batch E_k, store $O_k = O_{k-1} \cup E_k$ and $\rho(O_k)$, emit

$$\text{Frozen}\{tipContext, k, \rho(O_k), \text{sorted unique transaction IDs of } O_k\},$$

and start epoch $k + 1$. Each nonempty batch emits once. An empty active set continues ordinary root polling but emits no Frozen event, appends no batch and does not advance the epoch.

B.7 Rationale and error accounting

The calibration tests a local null: support for the item side or candidate root is at most p_0 throughout the run. A false qualification means accepting under that null, rather than any disagreement between honest nodes. Fresh uniform samples select K of the node's D peers. Selections are independent conditional on the neighbourhood; observed votes need not be. To apply the independent Binomial(K, p_0) calibration to those votes, one must bound the gated acceptance event under sampling without replacement and a conditional model of asynchronous responses. Appendix G.8 records the open proof obligations. Votes reflect local tallies and can spread neighbourhood information through Snowball metastability [6].

For a fixed prospective start round, summing $b\,2^{-(r-2)}$ over $r \geq 3$ gives b. The two item-side allocations sum to ε, and the root allocation is ε. Every polled round is a potential run start, including starts whose first round fails; calibration is unconditional and does not remove first-success mass. Under that calibrated conditional null model, for a node whose sampled neighbourhood satisfies the β_{adv} bound, per item and separately for the root, the probability of a false FINAL or freeze in an epoch is at most ε times the number of rounds polled in that epoch. This accounting is local; the network-wide false-freeze statement in Appendix H holds only under the assumption that every honest node's sampled neighbourhood satisfies the β_{adv} bound.

The null $p_0 = \beta_{\text{adv}} + (1 - \beta_{\text{adv}})/2$ accounts for all adversarial votes plus half the honest neighbourhood. Consecutive runs reflect recent preferences; one round is only one projection of a correlated

neighbourhood, motivating resets and the three-round floor. Majority counters let contested preferences move before confidence qualifies. Unanimous rounds supply more evidence than marginal successes, but exact thresholds determine qualification. Non-answers count for neither side, so silence alone cannot produce a success or move a preference; it can delay decisions.

Per-item agreement does not imply set agreement: many items can each have broad support while few peers hold an identical set. The cumulative-overlay root vote addresses that distinction through the existing REPLY_MATCH message. Candidate proofs and roots remain peer assertions, not proofs of body possession, honest intent or independent identities.

C Full State and Host Model

C.1 Host boundary and deployment

Snowstorm targets geographically distributed data-center nodes. Geographic latency and aggregate interface bandwidth are separate inputs to the evaluation. The protocol core is a deterministic state machine: the host serializes transaction admissions and removals, messages, timers, peer changes and chain-tip changes. It supplies time and a random seed, then executes the returned actions. Those actions send messages, fetch bodies, report frozen overlays and identify faults or anomalies. The core has no sockets, wall clock or remote ground truth; its host-supplied root function is pure and deterministic.

Definition 1 (Pseudo-consensus)**.** *Snowstorm coordinates transaction sets over an unchanged block-validity rule. Participation does not affect block validity. A block differing from every locally frozen overlay is relayed and validated through the existing path without a chain split.*

The host validates each transaction body and supplies immutable handles containing its transaction ID, transaction hash and sorted unique spent-input identities. These input identities support local, body-specific conflict checks. Fetch results distinguish consensus invalidity from host policy, resource and temporary-context refusals. Votes never authorize a consensus-invalid transaction.

C.2 Sets, commitments and epochs

We distinguish the live body store, the active epoch and the frozen archive.

- The compatible connected peer population has size D, and each round samples K distinct peers uniformly afresh, with $3 \leq K \leq D$.
- The live store R holds host-admitted body buckets keyed by distinct 64-bit short identifier; immutable frozen body handles are retained separately until the next tip.
- The active advertised set A contains identifiers of held nonarchived bodies whose active state is not FINAL-out; the active Ribbit encodes A, and $setSize = |A|$.

- The active candidate body set W contains held nonarchived bodies with effective preference IN; exclusions are $A \setminus \mathrm{IDs}(W)$.
- The Pending pool $\mathcal{P}$ holds reserved unadmitted item lifetimes supported by fetch, awaiting-admission, policy-waiting or FINAL-in-wait state; the outbound decode reference is $\mathit{Ref} = (A \cup \mathcal{P}) \setminus \{\text{active FINAL-out IDs}\}$.
- The cumulative frozen overlay is $O_k = \bigcup_{i=1}^{k} E_i$, where E_i is the immutable nonempty batch frozen in epoch i, and $O_0 = \varnothing$.
- The active root in epoch k is $\rho(O_{k-1} \cup W)$, and the frozen-prefix root is $\rho(O_{k-1})$.
- The configuration supplies the adversarial-fraction bound $0 \le \beta_{\mathrm{adv}} < 1/3$, error parameter $0 < \varepsilon < 1$, maximum set size M and root function ρ.

The host's commitment adapter computes the 32-byte non-coinbase overlay commitment ρ. With Merklix this is order-independent; a Merkle variant instead commits to the transactions in the canonical or deterministic order described in Appendix D. Tip and epoch correlation is carried separately; neither is added to the transaction commitment, and there is no ordered epoch manifest or separate prefix hash. Frozen bodies contribute to the overlay root but are not encoded again in the active Ribbit.

A tip starts at epoch 1. Held nonarchived items start each epoch IN with zero counters, absent confidence and no FINAL status; unadmitted Pending starts OUT. A fresh mid-epoch admission with no live Pending state starts OUT. Matching Pending admission preserves its entire live state. Archived body membership never reactivates during the same tip; a distinct body colliding with an archived identifier remains subject to the short-ID limitations.

C.3 Sampling and adversary

Adversaries may equivocate, fabricate identifiers, send malformed messages or withhold replies. The configuration assumes that their fraction in each node's neighbourhood is at most β_{adv}; the core supplies no Sybil resistance. A sampled peer abstains on an item when it supplies no eligible vote for either side. Peers are assumed to be selected randomly across the network. Fresh sample selections are independent conditional on the neighbourhood, although preferences and responses evolve. Appendix B.7 gives the conditional local accounting. Network convergence also relies on representative peer selection and Snowball metastability [6].

At small D the node polls all D peers. The production point at which K drops below D remains open for the specified (f, K) experiment, where f is host relay fanout. No production crossover or fixed formula for K is selected. With $D < 3$ the node opens no Snowstorm round or timer and uses ordinary block relay.

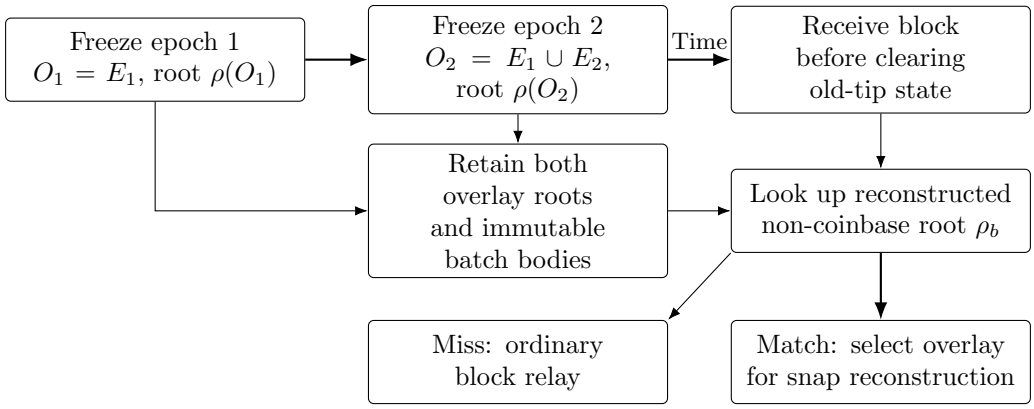


Figure 4: Successive epochs bind cumulative overlays rather than separate manifests. A header-plus-coinbase snap candidate reproduces the chain-specific block commitment from the frozen set and received coinbase, then compares with the header root. Merkle fallbacks use the required canonical or dependency-respecting deterministic order.

D Epoch and Commitment Details

Commitment compatibility. Merklix is the preferred commitment in all cases because it is order-independent, requires no ordering convention between miners and nodes, and updates incrementally as each epoch freezes. Canonical-order and deterministic-order Merkle variants are fallbacks for chains without Merklix. Merklix is not part of Bitcoin.

Snap lookup requires the block's transaction commitment to be reproducible from the frozen set. A chain can provide this through an order-independent commitment or a canonical ordering rule, such as CTOR with a Merkle root. On a chain without such a rule, such as Bitcoin, participating miners instead build from the Snowstorm overlay in a deterministic order computed from the cumulative set that respects topological dependencies. A topological sort with ties broken by the smallest transaction ID is one such convention. Receivers recompute the same order and Merkle root. The order is derived from the cumulative set, not from transaction arrival or concatenation of epoch batches. This participating-miner convention changes neither consensus validity nor the block commitment rule; blocks with a different set or order, including those from non-participating miners, miss the lookup and use ordinary relay.

Figure 4 connects the successive freezes to block reconstruction. Freezing seals an immutable batch and immediately opens the next epoch within the same tip. All held nonarchived items start the new epoch IN with fresh preference counters, confidence and FINAL state, including items that were previously FINAL-out. Unadmitted Pending resets to OUT with fresh epoch-local tally state while retaining live fetch tokens, donor retries, reservations and host callback obligations. The completed round's capture, observations, retained cells, active sizing and root state are released. The next round uses a fresh capture; old-epoch replies have no effect.

Frozen body handles and membership survive host removal from live R until tip reset. A

body is retained once across the live/archive union; cumulative overlays are represented by batch membership plus a bounded root index, rather than copying every prefix's body set. Historical views stream the union of the first k batches. The host's ordinary body-serving lookup includes retained archive handles, so a laggard can fetch bodies for a historical epoch without readmitting them at the serving node. A laggard polls its own epoch and can catch up one frozen batch at a time when its prefix matches.

Constant-cost snap path. For a successful match to a prevalidated frozen overlay, the preferred Merklix path requires $O(1)$ announcement and propagation metadata and $O(1)$ incremental validation after block arrival, in the number of transactions, with header and coinbase sizes held fixed. The transaction set and its validation results are already retained. Arrival checks the received header and coinbase against that prevalidated state and its commitment, rather than repeating transaction-set validation. This separation is the purpose of freezing sets before blocks: pre-discovery work prepares the block's contents, leaving constant incremental work on the propagation path. The incremental validation bound applies once the matching overlay is identified. For a header-plus-coinbase candidate, the described search may test each retained overlay; its cost also depends on the number of retained overlays.

GBT. The Snowstorm-backed GetBlockTemplate path returns the latest frozen non-coinbase overlay. With Merklix, template construction uses its order-independent commitment. A Merkle fallback orders the cumulative overlay by the chain's canonical rule or the participating miners' dependency-respecting deterministic convention, then constructs the coinbase, full transaction Merkle root and header under the chain's validity rules. Before the first nonempty freeze the path reports no frozen overlay available. It does not expose active W or fabricate an empty frozen epoch.

Snap lookup. Before clearing old-tip state on block receipt, the host computes the reconstructed block's non-coinbase commitment and looks it up among the overlay roots frozen since the preceding block. Merklix lookup uses the order-independent set commitment; Merkle fallback lookup preserves the received transaction order, so a differing order does not masquerade as a match to the stored ordered overlay. A match selects the retained overlay for snap-block reconstruction only when the chain's full block transaction commitment also reproduces correctly.

For a header-plus-coinbase candidate, receivers reconstruct the full block transaction commitment from each candidate frozen set and the received coinbase, then compare it with the header root. A Merklix chain can use its coinbase/subtree composition; a Merkle chain recomputes the full Merkle root in the chain-defined canonical order or the participating miners' deterministic topological order. The overlay cannot commit to a miner's not-yet-created coinbase. These paths add no block or header consensus field. The deterministic-order Merkle fallback requires no consensus change on Bitcoin. A miss uses ordinary relay. Both paths enforce ordinary consensus rules; the snap path reuses the frozen overlay's prevalidation instead of repeating transaction-set work after arrival.

D.1 Bounded parked lookup and optional template lag

A receiver can hold every block transaction while still finishing the epoch selected by the miner. We evaluate a bounded host policy that parks the arriving block, continues parent-tip agreement and retries lookup whenever a relevant overlay freezes. An exact commitment match selects the snap path, subject to ordinary header, coinbase, body and consensus checks. At the wait limit the host reconstructs the block through ordinary relay. A match arriving after reconstruction completes does not count as delayed-snap acceptance.

The host retains the old-tip archive while a block is parked. It must define the wait bound, competing-block handling and transition to the accepted tip. Parking changes reconstruction policy, not voting or block validity. Appendix G.6 describes the simulation and the implementation checks still needed.

A miner may choose an older retained overlay to give receivers time to catch up. The base GBT interface still selects the latest frozen overlay; lag is optional and introduces no consensus field or ordering convention. It trades transaction freshness for immediate lookup coverage. Omitted transactions wait for a later block, so overlay age must not be confused with their eventual confirmation delay. Receiver parking also supports miners that choose the latest overlay.

New tip and reorganization. After snap lookup, discard the old archive and context, cancel old fetch generations and rounds, rekey surviving host handles and start epoch 1 with surviving held items IN. The host removes block-included transactions first; omitted frozen transactions that remain held return to the remainder. A block disconnection uses the same surviving-handle reset. Compatible connections, measured peer times and monotonic token generators survive, but old asynchronous results cannot resurrect old-tip state. The deadline floor resets on a new tip, while designated first-round sizing history follows Appendix F.1.

E Conflict Replacement and Host Policy Details

Two validated bodies conflict exactly when their immutable spent-input identity sets intersect. This is a direct pairwise relation: a body sharing different inputs with two others does not make those others conflict. The host initially keeps at most one held spend per input under first-seen policy. At epoch start that held item starts IN; its unheld alternative receives NO locally. Fresh mid-epoch admissions, including conflicts, start OUT unless they transfer a live Pending state. Votes continue to follow effective W with no new wire field or conflict message.

Majority-selected replacement. A validated, policy-refused direct conflict alternative can request consensus-only admission when its effective preference becomes IN, without waiting for FINAL. Nonconflict policy override still requires FINAL-in. A mempool first-seen conflict is policy, not consensus invalidity; replacement never bypasses consensus validation. Existing fetch and Pending lifetimes survive replacement, as specified in Appendix K.3.

The host rechecks current authorization immediately before an atomic, consensus-valid replacement; a preference change or opposing terminal constraint can withdraw permission while fetching. Failed validity, resource or context checks preserve the previously held body. Appendix K.3 specifies the query, token binding and notification ordering.

Terminal constraints and anomalies. Round-end item transitions use conflict constraints present before that scoring batch. Stage qualified terminal outcomes, publish them, and detect direct double-FINAL-in conflicts before applying new effective-OUT constraints or evaluating admission authorization and freezing. Item traversal order cannot choose a winner or erase another qualified terminal decision.

A live FINAL-in decision makes each known directly conflicting alternative effectively OUT while the winner's lifetime remains live. This derived constraint does not fabricate statistical FINAL-out or overwrite a terminal decision. Blocked alternatives cannot gain IN confidence or enter W; activating the constraint resets their nonterminal confidence run while preserving majority counters and fetch reservations. A held loser can remain held until replacement completes but votes NO. An unadmitted winner never votes YES.

If directly conflicting bodies both have live FINAL-in decisions, including when a delayed descriptor reveals the relation, emit a `CONFLICT_DOUBLE_FINAL_IN` anomaly before suppression, preserve both terminal states and the FINAL-in Pending freeze veto, and grant no authorization to admit both. Bounded live-item markers suppress repeated identical anomaly reports. The existing per-item, per-prospective-start accounting applies; it supplies no additional pair-wide guarantee.

Any body conflicting with an already frozen overlay transaction is effectively OUT until the next block or tip reset. It cannot enter a later active batch or authorize replacement of that frozen winner. A delayed FINAL-in descriptor revealing such a conflict produces a `FINAL_IN_FROZEN_CONFLICT` anomaly while preserving terminal diagnostic state and excluding the later body. Archive membership, roots, historical service, GBT and snap lookup remain immutable.

In a balanced observed conflict fixture, candidate roots differ and eligible root matches are insufficient. Majority preference and replacement must align the sets before it freezes. The core has no global 50/50 detector or explicit exclusion filter. Random samples need not be balanced, and network agreement remains subject to the sampling assumptions.

F Adaptive Sizing and Timing Details

F.1 Adaptive request sizing

After underestimating a difference, a poller can request a longer Ribbit prefix. It selects n by minimizing $n/\widehat{P}(n)$ over candidates from an empirical decoding table, where $\widehat{P}(n)$ estimates decode probability. The table measures coding behaviour; the exact confidence calibration in Appendix B.4 serves a different purpose.

A verified MATCH supplies a zero-difference sizing sample, without claiming raw-set equality. A

DIFF supplies exact counts of its asserted lists. A fallback REPLY uses an approximate maximum-likelihood estimate from residual emptiness before peeling, treating schedule correlations and mixed-sign cancellation approximately. Neither those samples nor the table authenticate a peer's set. Inbound polls and later rescue supply no sizing samples.

When a sample is censored because no received residual cell is empty, its recorded lower bound includes the smallest difference for which the table's cumulative distribution function (CDF) is strictly below one half at that prefix length, rather than just the set-size gap. A censored upper-median sample forces prefix growth. So does a round with cell replies but no arrival/rescue decode and no MATCH or DIFF success. With a prior prefix, either trigger at least doubles the next request subject to the cap. Otherwise usable samples can shrink the request, including after an all-MATCH round; no usable samples retain it.

An initial sparse probe is used only without a prior prefix or usable first-round history. Once a prefix exists the epoch never returns to probe mode. Same-tip epoch rollover clears active sizing and starts with a fresh probe; only designated epoch-1-round-1 history is reused at the next tip, scaled to the new live set. Appendix K.6 retains the exact estimator, conservative CDF, generator, bounds and history rules.

F.2 Round timing

A round ends when all selected peers have answered or its captured fallback duration expires, whichever occurs first. The normal duration is four times the a_K-th smallest request-to-complete-reply estimate among the K sampled peers, with a minimum duration of one microsecond. Each estimate is that peer's latest measured POLL-to-complete-correlated-reply time, including transfer, without averaging. Unmeasured sampled peers use the upper median of measured sampled peers; if none is measured, use the captured host round-trip time (RTT) fallback. Imputation precedes the order statistic.

Initialize the duration at tip start from this formula. Thereafter it never shrinks within that tip, including across epoch freezes, peer changes, candidate changes or temporary inability to poll. At round end let Y be eligible current-commitment MATCH count and U unanswered selected slots. Double the next duration exactly when there is no root success and $Y + U \geq a_K$; otherwise use the maximum of the normal formula and current duration. Thus backoff applies when unanswered peers could have provided root success. Only a new tip resets the floor.

Detailed reply/timer ordering and slot timing eligibility are retained in Appendix K.7. Freeze is evaluated only after round-end scoring.

The implementation bounds message size, body/archive storage, Pending claims and retained replies. Attributable structural faults are distinguished from reference-dependent peel failures; a stalled peel or nonanswer is not itself misbehaviour. Appendix K.8 preserves the exact resource envelope and reporting policy.

F.3 Evaluated implementation refinements

A received response can resolve a slot without supplying a vote. The deadline-abstention candidate removes correlated, validated epoch-mismatch abstentions from U in the backoff predicate. They cast no votes and leave the denominator K unchanged. This is an experimental extension to the base contract, in which a behind-epoch responder is silent. Ending a round once every slot is terminal is a further proposal, with no isolated performance result included here; it must preserve legitimate incomplete decoding or continuation work.

The exclusion-encoding candidate transmits the same IDs more compactly. A DIFF uses either a bitmap or bounded ranks in the reconstructed, sorted responder set; a fallback can use monotone Elias–Fano encoding of sorted 64-bit IDs. Raw IDs are used when shorter. Positional formats require a coherent reply-time snapshot: the DIFF and exclusions must describe the same set, reconstructed against the receiver's saved poll capture. Implementations must check this requirement and malformed-input bounds. The formats extend the base grammar in Appendix K.1; no comparison with a general-purpose compressor is available.

The selected experimental configuration keeps $K = 32$ and the standard adaptive Ribbit sizing. Modest initial-prefix increases and a small minimum prefix remain candidates for measurement; the failed larger-prefix experiment does not rule them out. The approximate maximum-likelihood estimator already has a bounded unimodal integer objective, so a faster equivalent search is an implementation optimization, provided it preserves integer maximization, tie-breaking and censoring.

G Extended Evaluation and Audit Record

The primary deployment study uses geographically distributed data-center nodes. We report complete ten-minute simulations and a subsequent snap-lookup audit, alongside historical constrained-link convergence and adversarial tests. Analytic confidence counts in Section B.5 remain separate from network measurements.

G.1 Method and reporting conventions

Configurations are written as $N \times D$, where N is node population and D peer degree; K is sample size and f host announcement fanout. The data-center study uses $N = 500$, $D = 128$, $K = 32$, $f = 8$, five seeds per configuration, seven continuous Poisson arrivals per second, a 600-second mining time and a separately accounted reception tail. The network spans US East, US West, Europe, Tokyo and Sydney with geographic propagation and per-message jitter. Full-duplex ingress and egress are shared per node at 100 Mbps, 1 Gbps or 10 Gbps; there is no separate low per-peer cap. Primary runs deliver messages reliably without injected loss; they do not simulate TCP retransmission.

Transaction bodies are 250 bytes and host validation delay is one millisecond. The simulator models validation rather than running a production chain implementation.

The three initial shared-set divergence targets omit each of 830 transactions independently at each node with probability $q = (1 - \sqrt{1-2p})/2$, for $p \in \{0.01, 0.03, 0.05\}$. Each node additionally starts with five unique transactions. Pairwise unique differences add ten identifiers, giving measured total initial differences of approximately 18.34, 34.99 and 51.69 transactions, or 2.21%, 4.22% and 6.23% of the 830-transaction reference universe. These are sensitivity workloads, not measurements of deployed cryptocurrency mempools. Initial node-unique transactions and continuous arrivals are distinct populations.

Historical stress runs instead serialize each message independently at 56 bytes/ms (448 kbps per peer stream), with no shared NIC queue. They permanently drop 1% of POLLs without transport recovery. Their elapsed times are not predictions for data-center deployment. Historical single-epoch tests stop opening polls after 120 rounds or 120 seconds. The later snap audit corrects an artificial 2000-round cap across an entire tip to a per-epoch cap; this changes no confidence threshold. Earlier data-center interval counts, traffic and snap fractions affected by the cumulative cap are explicitly separated from corrected results below.

The calibration is $p_0 = 0.6$, $\beta_{\text{adv}} = 0.2$ and $\varepsilon = 10^{-9}$. The base implementation and experimental deadline/exclusion candidate are distinct arms. Confidence uses the strict median gate; timing in the tested binary uses its rank $a_K = 20$ at $K = 32$, as does the current base contract in Appendix F.2. Earlier experimental descriptions instead referred to the historical 80% RTT rank; those descriptions must not be substituted for the actual tested scheduler.

Node-height observations share a topology and history; 2500 receivers per configuration are five trials, not 2500 independent experiments. Modal-root and split-root flags are agreement diagnostics, not independent tests of the statistical null or consensus validity. Observed freeze distributions are conditional on completion. Nonfinishers are reported separately, and an unreached all-node percentile is not replaced by the horizon. An immediate snap hit, an eventual delayed snap and body possession are different outcomes. Historical single-epoch snap percentages use averages over 1000 conditional block-time draws per trial; the later audit uses one miner-selected block per trial and actual receiver counts.

Reproducibility. Tables use raw logs and per-seed records. The snap suite uses five seeds: `0x20261025`, `0x20355268`, `0x204494ab`, `0x2053d6ee` and `0x20631931`. Appendix G.8 identifies further validation work and the claims it would support.

G.2 Benign convergence and the initial-skew fixture

Table 5 separates two initial-skew workloads at 200×32, with $f = 8$. The skew setting controls how many core transactions may be absent at epoch start. In the skew-75 control, every honest instance freezes. Polling all 32 peers reduces the mean round count relative to the experimental $K = 8$ fixture, but does not reduce the median elapsed time in this control; round durations are adaptive. These results do not select a production $D \to K$ crossover.

The skew-800 fixture places some core items near an initially even preference split. Body

Skew	K	Frozen / honest	Mean rounds	Time p50 / p99 (s)	Snap hit
75	8	1000 / 1000	11.27	4.082 / 12.988	99.20%
75	32	1000 / 1000	5.91	4.450 / 12.113	99.30%
800	8	399 / 1000	76.32	Unreached / unreached	38.70%
800	32	400 / 1000	31.25	Unreached / unreached	39.40%

Table 5: Benign 200×32 simulation results over five trials. Mean rounds are conditional on freezing; time percentiles include all honest instances. Skew-800 rows are fixture-limited. The $K = 8$ rows are experimental fixtures. Single-epoch snap hit uses the conditional block-time-draw estimator described in Appendix G.

possession and preference membership then diverge: an epoch-start item starts IN, while an item learned and admitted during the epoch starts OUT. Fetching every missing body therefore need not put every item in W. For a traced core item at five percent adversaries, all 190 honest nodes possess the body by 350 milliseconds, but only 86 prefer IN and 104 hold it OUT. In the ten-percent trace, one item reaches 179 of 180 honest nodes by 350 milliseconds and all 180 by one second. This distinction explains why rapid body relay alone does not establish rapid set agreement.

The adversarial fixture also changes the starting vote balance. In R3, adversaries transmit YES on the skewed core items, while equivocating on a designated split item. Their increasing population can push a near-even core-item preference distribution towards IN. Consequently, rising freeze rates as the adversarial fraction increases in this fixture are not evidence that adversaries improve the protocol. The severe-skew benign control itself freezes only about two fifths of instances. These workloads measure convergence from difficult initial preferences as well as adversarial behaviour.

G.3 Adversarial convergence and gate comparison

R3 denotes the equivocation scenario and R4 the withholding scenario. Table 6 reports the median-gate 500×128, $K = 32$, $f = 8$ runs. At five and ten percent adversaries, R3 reaches full modal agreement within the horizon. At twenty percent, R3 and R4 reach 98.55% and 98.95%, respectively. The missing instances are explicitly nonfreezing; their time percentiles are not silently replaced by 120 seconds.

Scenario	Frozen / honest	Nonfreezing	Mean rounds	Time p50 / p99 (s)	Snap hit
R3, 5%	2375 / 2375	0	17.74	24.650 / 44.213	98.40%
R4, 5%	2056 / 2375	319	23.84	28.734 / unreached	85.09%
R3, 10%	2250 / 2250	0	15.44	20.727 / 40.268	98.09%
R4, 10%	2250 / 2250	0	15.06	19.981 / 37.123	98.27%
R3, 20%	1971 / 2000	29	23.57	26.656 / unreached	93.60%
R4, 20%	1979 / 2000	21	20.15	22.819 / unreached	94.55%

Table 6: Adversarial 500×128 simulation results over five trials with $K = 32$. Mean rounds include freezing instances only; time percentiles include all honest instances. Zero simulator false-freeze and split-root flags are reported in every row. Single-epoch snap hit uses the conditional block-time-draw estimator described in Appendix G.

Snap hits without universal freeze. Freeze is a local decision, not a network-wide barrier. A miner can build from its latest frozen overlay while some other nodes are still polling. Each receiver independently obtains a snap hit if it has already frozen and retained the overlay selected by that block and can reproduce its commitment. Other nodes use ordinary relay. In the R4 five-percent row, for example, 86.57% of honest instances freeze within the horizon, while the conditional block-time-draw snap-hit estimator is 85.09%, despite 319 instances remaining nonfreezing. With successive epochs, a receiver may also match an earlier retained overlay selected by the miner while continuing to poll a later epoch. Universal freeze, eventual freeze within the experiment's horizon and possession of the selected overlay at block arrival are different conditions.

Benefit on a compact-block fallback. Reconciliation can also help when no snap lookup succeeds. A node may already possess transactions discovered through Ribbits and body fetching even while their preferences remain OUT or the root has not qualified. Where the host retains those bodies and makes them available to ordinary block reconstruction, greater overlap with the mined block can reduce the number of missing transactions that compact-block relay must fetch. Thus partial convergence can improve the fallback path as well as prepare successful snap blocks. This is a mechanism-level expectation, conditional on the block's contents and the host's body availability; the current experiments do not quantify the resulting reduction in compact-block requests, bytes or reconstruction latency.

A historical comparison replaces the median gate $a_{32} = 20$ with an 80% gate requiring 26 votes, using a separate exact threshold calibration for that gate. At twenty percent adversaries, its freeze fractions are 51.70% in R3 and 53.60% in R4, compared with 98.55% and 98.95% under the median gate. The median gate is not uniformly better: R4 at five percent adversaries freezes 86.57% under the median gate and 99.96% under the 80% gate. The former has 319 nonfreezing instances concentrated in one trial. Comparisons must retain this counterexample and the initial-skew qualification, rather than claiming a universal improvement.

Global adversary percentages do not establish the local security assumption. At twenty percent globally, 899 of 2000 honest node-height instances in these 500×128 runs have more than twenty percent adversarial neighbours, with a maximum of 31.25%. Even the ten-percent global configuration has one such instance out of 2250. The experiments therefore exercise neighbourhoods outside the stated bound as well as neighbourhoods within it; they are not an empirical verification of the conditional network-wide error bound.

G.4 Data-center latency and communication

The first epoch in the reliable data-center study takes approximately six rounds: median freeze time is 1.33–1.34 seconds at 100 Mbps and 1.17–1.21 seconds at 1–10 Gbps. In the combined deadline-abstention/exclusion candidate, later completed epochs have median five rounds and median freeze time 1.55–1.63 seconds. The corresponding constrained-link candidate takes 24.5–26.9 seconds. Bandwidth and loss assumptions change together in this comparison, so it does not isolate their

causal contributions.

Geographic delay remains: mean propagation RTT is approximately 111–114 ms, and nearly all terminal responses arrive within 250 ms. Gathering all selected replies takes about 200 ms. Received epoch abstentions still prevent the tested all-answered condition from firing. About one quarter of candidate rounds wait for the deadline despite having no outstanding reply; the resulting idle wait averages approximately 122–127 ms per round. Excluding those abstentions from backoff avoids exponential deadline inflation but does not itself implement terminal-slot early completion.

Initial difference	First p50 (s)	Later mean rounds	Later p50 (s)	Total MB/node
2.21%	1.177	5.68	1.622	48.161
4.22%	1.201	5.53	1.546	46.774
6.23%	1.207	5.70	1.617	47.779

Table 7: Combined candidate at 1 Gbps in the data-center network suite, five trials per row. Later-epoch median round count is five in each row. MB denotes 10^6 bytes; totals include polls, replies, exclusions, announcements, body requests and bodies over the interval and its tail. These runs retain the subsequently identified cumulative round-cap defect; their completed-epoch counts and snap fractions are not used as corrected estimates.

Complete interval accounting after the cap correction. The subsequent snap suite removes the cumulative cap while keeping the combined candidate and 1 Gbps model. Table 1 is calculated directly from its per-seed counters. It charges all polling during the ten-minute interval, including unfinished epochs, without extrapolating a first epoch or multiplying a median cadence. Reception and block-reconstruction costs are reported separately in Appendix G.6.

Where the bytes go. In constrained-link diagnostics, fallback Ribbits and repeated exclusion IDs account for about 85% of base reconciliation traffic. Much of the late-epoch difference is FINAL-out retirement lag: one peer has retired an ID while another still advertises it as excluded. MATCH samples shrink outgoing prefixes, but a nonmatching peer can then return a larger Ribbit. That fallback arrives in the response to the poll and need not add a request/response cycle. Enlarging every poll, however, charges all selected peers, including those that MATCH.

In that stress suite, exclusion encoding reduces exclusion bytes by 43–48% and full-interval total traffic by 1.9–10.3%, with mixed per-seed outcomes. An experimental exchange-cost policy instead expands late prefixes to hundreds of cells and more than doubles total communication. Its initialization also contains a fixed divergence heuristic; it is not adopted as a derived optimal policy. A capped small-K ramp yields mixed interval savings and slower freezes. None of these results establishes the best initial prefix, minimum prefix or production K.

Comparison boundary. The direct interval counters charge recurring agreement and unfinished epochs. The historical compact-block, Graphene and Erlay models use different workloads and block-selection assumptions. The explicit calculation below normalizes body distribution and initial

traffic, but does not constitute a matched implementation benchmark. Small snap messages alone do not imply low total bandwidth.

G.5 Computed relay costs and voting budgets

The preliminary TPS sweep in Section 8.1 uses five seeds at each of 0, 1, 3, 7 and 25 TPS and three at each of 100 and 250 TPS. Values are extracted from the per-seed `TPS_SCALING_SEED` records in the corresponding `corrected_rate` logs. All reported seeds have zero universe-cap drops. Fits give equal weight to each rate's mean, using 1, 3, 7, 25 and 100 TPS only; they do not weight the larger seed count at lower rates. The 250-TPS point is reported separately rather than forced onto that line. The reconciliation-only fit is $35.07 + 1.2172r$ MB per node per 600 seconds ($R^2 = 0.9965$). Its slope is approximately 2029 bytes per additional transaction, counting sent plus received traffic. This is a marginal fitted cost, distinct from total traffic divided by committed transactions.

Body distribution and the workload boundary. Continuous bodies are initially supplied to most nodes: at the three-percent shared-divergence setting, each node misses an arrival with probability $(1 - \sqrt{0.94})/2 \approx 0.0152$, apart from the guaranteed origin. Interval body counters therefore cover residual fetching, not full dissemination. The fixture also injects five node-unique transactions at each of 500 nodes. These 2500 auxiliary transactions consume communication even though they are excluded from the eligible mined set. The calculations below explicitly charge their distribution on both sides.

Let $n(r) = 830 + 600r$ be the nominal number of eligible transactions in the interval and let $u = 2500$ be its auxiliary initial transactions. Actual simulated arrival counts fluctuate and are used when normalizing measured counters. Suppose each transaction starts at one node, every other node fetches it once, a request costs 36 bytes and a body costs 250 bytes. The network-average sent-plus-received body budget per node is

$$D_{\text{body}}(n + u) = 2(36 + 250)\frac{499}{500}(n + u).$$

Replace Snowstorm's measured request/body counters by this same analytical distribution budget. The resulting positive-rate fit is

$$B^*_{\text{S}}(r) \approx 38.872 + 1.9023r \quad \text{MB per node per 600 seconds.}$$

This is an accounting normalization, not a simulation of single-origin propagation: changing delivery timing can also change reconciliation costs.

Full-degree announcements plus compact blocks. The existing baseline assumes one 36-byte INV per transaction per undirected peer link, yielding $36D(n + u)$ sent-plus-received bytes per node. It also charges three compact-block transmissions and receptions, each approximated by

an 80-byte header plus six bytes per eligible transaction. At $D = 128$,

$$\begin{aligned} B_{\mathrm{CB}}(r) &= 36D(n+u) + D_{\mathrm{body}}(n+u) + 6(80+6n) \\ &\approx 17.276 + 3.1289r \quad \text{MB per node per 600 seconds.} \end{aligned}$$

The first line is in bytes; the second converts to decimal MB. This simplified model omits prefilled coinbase data, count/framing overhead, missing-block-transaction exchanges, duplicate transfers and transport recovery. Its degree and announcement assumptions are explicit parameters, not a claim that every deployed node uses degree 128.

Equating the two fitted expressions gives

$$r^* \approx \frac{38.872 - 17.276}{3.1289 - 1.9023} \approx 17.6 \text{ TPS.}$$

Omitting the auxiliary initial population from both analytical budgets instead gives intercepts 37.445 and 4.329 MB, with the same slopes and a calculated intersection near 27.0 TPS. This sensitivity explains the roughly 18–27 TPS range. Both intersections lie within the fitted 1–100 TPS range. They are model-dependent traffic comparisons, not measured production break-even rates, equal-security results or hardware-cost estimates. Lower announcement degree or a different relay policy changes the baseline slope; the 250-TPS departure also prevents indefinite extrapolation of the fitted Snowstorm slope.

Explicit per-transaction polling. ABC's documented Avalanche messages list 36-byte inventory entries in polls and 36-byte vote entries in responses; a response also carries a shared 64-byte signature [15]. For a hypothetical policy soliciting P peer responses per transaction, item fields alone consume $72P$ bytes at the initiating node. If all nodes poll at the same rate, serving other nodes' polls adds the same network-average load, giving

$$B_{\text{item fields}} \approx 144P \quad \text{sent-plus-received bytes per transaction per node.}$$

At an illustrative $P = 80$, this is 11,520 bytes per transaction before message framing and shared signatures. Eighty is a comparison parameter, not a verified deployment default. Batching item lists amortizes their signatures and headers but retains the item fields. Polling one block, a shared set root or a DAG ancestor can amortize observations differently; this expression is not a lower bound for all Avalanche-family designs. It is also not an equivalent-confidence comparison: Snowstorm's root qualification and per-item FINAL have different roles. Its purpose is to expose the communication saved by representing a whole candidate set with a root rather than a list of transaction votes.

Zero or limited INV pushes. Reconciliation supplies a discovery path in place of unsolicited announcements. With $f = 0$, nodes discover identifiers through polls and replies and fetch the

bodies. With $f = 8$, the host also pushes an announcement for each newly admitted transaction to at most eight eligible peers, even at degree 128. Table 8 compares these modes. Combining push and pull follows the randomized rumor-spreading framing [10].

Small fanout can nevertheless disseminate a transaction to most nodes because each recipient forwards it in turn. In an idealized large, lossless network where every recipient forwards once to f independently selected uniform destinations, random directed-graph reachability gives the asymptotic reached fraction z_f as the positive solution of

$$z_f = 1 - \exp(-f z_f), \qquad f > 1.$$

For fixed fanout, this follows from the random f-out model [22]. The resulting fractions are approximately 98.02% at $f = 4$ and 99.97% at $f = 8$. These are analytic illustrations, not measured coverage fractions for the simulator: forwarding over an existing peer graph, eligibility filtering, losses and adversarial nonforwarding change the model. They explain how limited pushes can provide broad initial dissemination while polling and reconciliation supply an additional discovery path.

Transaction relay mode	INV fanout f	Mean rounds	Mean freeze (s)	Snap hit
Polling and body fetching	0	6.38	2.593	99.68%
Limited pushes plus polling	8	6.32	2.509	99.76%

Table 8: Snowstorm with unsolicited INV pushes disabled or limited to at most eight peers per transaction, at $N = 500$, $D = 128$, $K = 32$, with no unique-per-node arrivals. Each mode reaches full modal agreement over 2500 honest node-height instances in five trials. Eight is the tested limited fanout, not an established minimum or optimum. Single-epoch snap hit uses the conditional block-time-draw estimator described in Appendix G.

This historical single-epoch workload does not test continuous arrivals into the separate open set U. Under that lifecycle, an ordinary arrival is not advertised by the current epoch's Ribbit; without pushes it becomes discoverable there only after assignment at an epoch cut. This table therefore cannot establish zero-push discovery latency or liveness for the corrected design.

Both relay modes converge in this workload, and the no-unique-arrivals suite reaches full modal agreement at every tested valid K. Reducing fanout shifts discovery work into reconciliation, whose recurring cost appears in the interval measurements above. These experiments do not time the matching post-discovery validation path or change the $O(1)$ claim in Appendix D.

The host could also explore a relay layer such as Dandelion++ [21] alongside reconciliation. This is a future deployment option, separate from the $f = 0$ and $f = 8$ experiments: neither implements Dandelion++, and their convergence results establish no origin-privacy guarantee for the combined system.

G.6 Immediate and delayed snaps, and template lag

The snap audit at 1 Gbps checks every retained cumulative overlay, rather than only the latest root or matching epoch number. It finds no overlooked existing root or flagged commitment divergence in the tested suite. The initial low coverage partly came from the cumulative 2000-round cap: approximately one fifth of nodes stopped opening rounds before mining. Correcting the cap removes that artificial halt. A further arrival-time miss can occur when a receiver already holds all block bodies but has not yet frozen the miner-selected overlay.

The experimental parked policy preserves parent-tip polling until acceptance or a bounded wait expires. Let d be the receiver's current captured round duration. The tested code uses $d' = \max(250\,\text{ms}, d)$ and waits either d' or $3d'$; the 250 ms floor is an implementation choice, not a derived protocol constant. A match must use the correct parent and block commitment. A miss at the wait bound invokes a simplified short-ID reconstruction model. Table 3 uses operational acceptance counters, not retrospective overlay availability counts.

Among delayed snaps with the one-duration wait, the reported pooled median waiting times are 58, 89 and 60 ms, respectively. All-node modeled mean acceptance delays from reception are 20.2, 14.7 and 5.8 ms; immediate matches contribute zero. These are not end-to-end block-propagation or production validation timings. A one-epoch-older template achieves immediate lookup in every tested receiver, while omitting approximately 0.21–0.30% of the roughly 5000 transactions in the latest overlay. Its overlay age is roughly 1.4–3.5 seconds. The omission may defer confirmation until a later block; the experiment does not measure that eventual delay.

Miner lag improves immediate coverage by selecting an older set; parking preserves the latest template and waits for receivers to catch up. On the measured misses, receivers already hold every block body. Ordinary reconstruction can therefore avoid body transfers, although it still needs metadata. The experiment models a request and short-ID list, followed by requests and transfers for any absent bodies. It does not implement a new overlay-ID protocol or the full BIP 152 exchange.

Acceptance model. The simulator marks acceptance and stops starting parked parent-tip polls, but does not execute a production new-tip and validation pipeline at each receiver. Accepted nodes can still process parent-tip service events, which may help laggards. Fallback completion time is assigned from geographic RTTs rather than a queued block-relay exchange. Table 3 uses operational acceptance counts; retrospective freeze-time counts differ slightly at event boundaries. Production acceptance latency and catch-up after actual tip changes still need measurement.

G.7 Historical laggard recovery

A separate constrained-link, five-trial 500×128, $K = 32$, $f = 8$ experiment silences 25 nodes per trial until 20 seconds. All 125 laggard instances catch up through historical epochs. Median catch-up is 2.638 seconds and p99 is 8.388 seconds; the run serves 243289 historical polls and 4773 historical bodies with no flagged genuine prefix split. Its mining times and workload differ from the ten-minute data-center suite, and completed-epoch counts include a five-second event tail.

These observations support same-prefix catch-up, not repair of a genuine frozen-prefix split or an adversarial data-center guarantee.

G.8 Open validation tasks

These simulations measure interval traffic under the stated network assumptions. This design draft leaves three questions open: costs in matched implementations, asynchronous confidence bounds and implementation fidelity. The computed relay comparisons above state their accounting assumptions. The older stress tests describe their own workloads and do not validate adversarial liveness of the newer host policies.

Calibration and asynchronous scope. The normative thresholds are exact. Historical simulator runs approximate longer confidence runs with score-binned dynamic programming; an audit found small budget excesses and added conservative rounding and exact enumeration. The numerical enclosure, runtime boundary comparisons and claims of unchanged decisions need independent checking. Static-snapshot checks do not prove a bound for changing asynchronous responses. In particular, a limit on simultaneous supporters does not limit the union of peers that can support during a round. The conditional accounting in Appendix B.7 still needs an appropriate response model; the neighbourhood adversarial-fraction bound alone does not supply it.

Implementation fidelity. Further implementation evaluation should check reply snapshots for positional exclusion encoding, parked-block tip transitions and historical serving, and matched relay accounting. The simulator models field operations, Merklix and host validation; it is distinct from an implementation of the public core API. The approximate-estimator search optimization and any global-state-dependent fixture decisions also need verification. Modest initial prefixes, minimum prefixes and terminal-slot completion are not evaluated in the corrected deployment measurements reported here.

H Detailed Limitations

Adversarial bound and liveness near the limit. The model requires $0 \le \beta_{\mathrm{adv}} < 1/3$ in each node's peer population. The core cannot establish that bound and supplies no stake weighting. Even if all honest peers agree and adversaries vote against them, the margin between honest support $1-\beta_{\mathrm{adv}}$ and the null $(1+\beta_{\mathrm{adv}})/2$ tends to zero as β_{adv} approaches $1/3$. Confidence runs can therefore take longer near the limit; no bounded freeze time is established. Snowstorm requires $3 \le K \le D$ and idles below that precondition.

Conditional error bounds. The terms false FINAL and false freeze retain the local-null meaning defined in Appendix B.7. Consider a node whose sampled neighbourhood satisfies the β_{adv} bound and the conditional-null sampling and response assumptions in Appendix G.8. Per item,

and separately for the root, the probability of a false FINAL or freeze in an epoch is at most ε times the number of rounds polled in that epoch. Let N be the number of nodes and R the round horizon per node. If every honest node satisfies those conditional-null assumptions and its sampled neighbourhood satisfies the β_{adv} bound, a union bound gives that the probability any honest node makes a false freeze within R rounds is at most $N \cdot R \cdot \varepsilon$. The union-bound step requires no independence between nodes or disjoint neighbourhoods; it does require valid local bounds. It supplies no guarantee where a neighbourhood exceeds β_{adv} or the required response assumptions fail.

Limits of the bound. No bound is claimed when a local adversarial fraction exceeds β_{adv}, as can occur under eclipse or a clique partition that violates this assumption. No bound fixed over unlimited heights is claimed, since the error bound grows linearly in rounds. The published Snowball/Avalanche security results [6, 7, 8] are not inherited: the decision rule changes through the strict median gate, evidence sums with exact T_r, a set-root vote, many items and adaptive timing.

Neighbourhood scope, eclipse and partitions. Random peer selection across the network is assumed. Host network-group diversity and outbound preference raise Sybil cost but do not establish a hard adversarial bound; stake-backed identity is required for that bound. Partitioned or clique-bridged topologies can freeze different roots. Such splits delay block reception under pseudo-consensus without changing consensus validity. A genuine frozen-prefix split supplies no cross-prefix item or root evidence and has no Snowstorm fork-repair rule; the next accepted block performs the ordinary reset.

Timing, decoding and equivocation. Abstentions leave thresholds based on captured K. Response selection, partial-decode conditioning, changing preferences and adaptive equivocation still require agreement and liveness analysis. A peer holding several candidate sets can supply valid MATCH proofs to different pollers. Those proofs establish neither body possession nor independent identities or honest intent; short-ID collision limitations also remain.

Arrivals and storage. Successive epochs retain sealed overlays while later arrivals affect only active state, but no throughput or liveness guarantee is established under an unbounded arrival stream. A FINAL-in Pending item can delay freeze while bodies or admission resources remain unavailable. Finite message and state limits bound storage; they do not guarantee successful decoding at the cap or bounded computation under adversarial input.

Short-ID collisions and consensus fallback. Sketches reconcile distinct IDs rather than all colliding bodies. A hidden colliding body can escape identifier reconciliation, and a matching invalid body can cause a valid colliding body to be conservatively rejected for the tip lifetime. Ordinary host validation and block fallback remain necessary. Snap lookup must precede archive reset and never bypasses consensus validation.

Commitment compatibility. Snap lookup is conditional on reproducing the block's transaction commitment from a frozen set. The preferred Merklix design in Appendix D is not Bitcoin's commitment. Chains without Merklix use the canonical-order or deterministic-order Merkle fallbacks; on Bitcoin, the latter is a participating-miner construction policy that respects topological dependencies and requires no consensus change. A block from a non-participating miner that uses a different set or order misses the lookup and uses ordinary relay. Set agreement alone does not make an arbitrary miner's Merkle root reproducible.

Scope of the complexity claim. The $O(1)$ claim applies after identifying a match to a prevalidated overlay in the preferred Merklix path, with fixed header and coinbase sizes. It covers announcement and propagation metadata and incremental post-discovery validation. Body storage, pre-discovery reconciliation and validation, lookup across retained overlays and ordinary relay on a miss have their own costs. Canonical- and deterministic-order Merkle fallbacks additionally require ordering and root computation. Network conditions still determine propagation latency.

Origin privacy. Origin privacy depends on the relay layer. Snowstorm does not supply an origin-hiding mechanism, and reconciliation exposes identifier-set information to peers. A deployment seeking origin privacy could consider a relay-layer design such as Dandelion++ [21]; its interaction with Snowstorm requires separate analysis.

I Additional Research Directions

I.1 Standalone block reconciliation with Ribbits

A Ribbit could reconcile a mined block against the receiver's mempool independently of Snowstorm voting. Coded prefixes would grow as needed, followed by fetching missing bodies. Such a design must reproduce the block's transaction order and commitment and perform ordinary consensus validation. It would reconcile after mining rather than identify a prevalidated overlay. Block "shredding," transport and recovery are open design work, with no standalone implementation or performance comparison here.

I.2 Initial Ribbit sizing and terminal-slot completion

A modestly larger first Ribbit may avoid expensive initial fallbacks, with later prefixes shrinking as differences disappear. A three- or four-cell minimum is another candidate. Moving from one to four 28-byte cells adds $28(4-1)K$ bytes per round before framing, or 2688 bytes at $K = 32$. The test is whether this extra poll traffic saves enough fallback bytes or delay. Neither minimum is selected, and neither has a matched result under the corrected data-center model.

The initial request is reset at epoch rollover; within an epoch the next round uses the preceding round's sizing observations. Alternative initial sizing and gradual downsizing should be evaluated independently. Terminal-slot completion could remove waiting after every selected peer has replied,

including abstentions, while preserving K and confidence rules. Starting a new round before every response arrives, or accepting past-round votes, would instead require a separate design for retained captures and confidence-run ordering; neither is part of this proposal.

I.3 Estimating network size to choose K

A future node-local policy could estimate the network population T from epoch-specific random identifiers. Each node would form its epoch ID as $\mathrm{hash}(\textit{pubkey} \,\|\, \textit{epoch salt})$, normalized to $[0, 1)$ and modeled as uniform, with independent salts across epochs. Nodes would gossip the smallest ID seen for that epoch, carrying one value per poll. For a stable population and completed minimum dissemination, the network-wide epoch minimum μ_e is approximately

$$\mu_e \approx \frac{1}{T}.$$

Across m independent epochs, the continuous German-tank approximation gives

$$\widehat{T} = \frac{m-1}{\sum_{e=1}^{m} \mu_e}, \qquad \text{relative error} \approx \frac{1}{\sqrt{m}}.$$

These expressions use the large-population approximation for uniform minima; they are not an exact finite-population error guarantee.

The node could use $D/\widehat{T}$ to choose K adaptively, polling all D peers when its neighbourhood is a large fraction of the network and sampling $K < D$ when $D \ll T$, as indicated by the estimate. This would replace the open fixed crossover with a network-size-aware local policy, while retaining the existing confidence calibration and reset rules for the chosen K.

Small fake IDs lower observed minima and inflate $\widehat{T}$, conservatively reducing the estimated fraction of the network covered by a neighbourhood. Adversaries can also deflate the total-population estimate by withholding their own IDs, even when all honest minima disseminate; an eclipse can hide further minima. Additional participation assumptions are therefore needed to estimate the total population. The proposal is an unimplemented, unmeasured node-local policy. It adds no consensus rule, supplies no Sybil resistance and does not strengthen the neighbourhood-to-network guarantee. Its extra poll value is outside the current wire contract.

Participation assumption. With H honest nodes and A adversarial nodes, complete dissemination of honest IDs can coexist with adversaries withholding their own IDs. The observed minimum then has mean $1/(H+1)$ although the stated target is $T = H + A$, without an eclipse. Choosing the target population and the participation assumption remains an author decision; the estimate must not be used as an established safety bound.

J Extended Related Work

Compact blocks and Graphene. BIP 152 represents a block with short transaction identifiers and prefilled transactions; receivers request missing bodies through the block-transaction path [14]. Graphene uses a Bloom filter and IBLT to recover the announced set [16]. Both exploit mempool overlap after the miner has chosen a block. Snowstorm prepares a cumulative overlay before mining. Appendix G.4 states what the available measurements can support about bandwidth.

Erlay and BIP 330. Erlay combines limited flooding with set reconciliation for transaction dissemination [17]. BIP 330 specifies the announcement-reconciliation exchange and its extension/fallback path [18]. Peers can converge on their inventories without freezing a candidate set. Snowstorm adds preference voting and a root decision; transaction announcements remain a host function.

Rateless reconciliation. IBLTs cancel shared identifiers and peel residual cells [5, 11]. Yang et al. make the encoding rateless so that a prefix can grow with the difference [9]. BCH power-sum syndromes provide Snowstorm's cell algebra [3, 4]. Its integer schedule is deterministic, while its two-item decoder resolves a pair within one cell. Peeling removes each recovered identifier from the other cells to which it contributes within that peer's Ribbit.

Avalanche and Mysticeti. Snowball supplies the repeated-sampling structure [6]. Bitcoin ABC distinguishes post-consensus on mined blocks from pre-consensus on transactions [19]; post-consensus alone does not prepare a pre-block overlay. Mysticeti studies consensus and a transaction fast path over an uncertified directed acyclic graph (DAG) [20]. Its setting differs from sampling connected neighbours. Further binomial-weighting optimizations remain possible node-local speed-ups; Snowstorm uses the null test and thresholds defined here.

Snow-family security. Amores-Sesar, Cachin and Tedeschi formalize Avalanche and identify a liveness vulnerability [7]. Frosty studies Snowman consistency and adds a liveness module for adversarial conditions [8]. These results concern their specified protocols and assumptions. Snowstorm changes the decision rule and does not inherit their guarantees; Appendix H states its security scope.

Relay privacy and commitments. Dandelion++ addresses transaction-source anonymity in the relay layer [21]. Snowstorm provides no such mechanism; any combined deployment requires a separate privacy analysis. Polls disseminate identifiers in both directions, motivating a push–pull framing [10], while the commitment must be reproducible from the frozen transaction set. Merklix is an order-independent commitment proposal, not part of Bitcoin [2]. A chain-defined canonical order, such as canonical transaction ordering (CTOR) with a Merkle root [1], can also make that root reproducible. On chains without either rule, participating miners and receivers can derive

the same dependency-respecting transaction order from the set and compute its Merkle root. The received coinbase is incorporated when reconstructing the full block commitment; it is not part of the frozen non-coinbase set.

K Implementation Contract

This appendix records the implementation contract for the main algorithm and the detailed mechanisms in the preceding appendices.

K.1 Wire grammar, proof and reconstruction

All four message kinds carry version 3, a kind byte, the 32-byte tip context, the epoch number and the 32-byte frozen-prefix root. Kinds 0, 1, 2 and 3 denote POLL, REPLY_MATCH, REPLY and REPLY_DIFF respectively. Each carries a nonzero poll ID that strictly increases without reuse at its originating node. Epoch numbers are at least 1; integers are 64-bit little-endian except the byte-sized version, kind and mode, lengths use canonical CompactSize, and trailing bytes are rejected. The message-specific fields are:

```
POLL:        pollID, nonce[16], root[32], setSize, mode, n,
             m, cells[m]
REPLY_MATCH: pollID, proof[16]
REPLY_DIFF:  pollID, rootVote[32], setSize,
             exclusionCount, exclusions[],
             missingCount, missing[], wantedCount, wanted[]
REPLY:       pollID, rootVote[32], setSize,
             exclusionCount, exclusions[], m, cells[m]
```

A prefix POLL requests and supplies Ribbit cells at $[0, n)$, with mode 0 and $n \geq 1$. A probe uses mode 1, $n = 0$ and indices $\{0\} \cup \{2^j - 1 \leq \mathit{setSize} : j \geq 1\}$, sorted, deduplicated and clipped to the cell cap by keeping the lowest indices. Its basis is the send-time set size. A fallback REPLY always supplies a dense Ribbit prefix $[0, m)$, with no mode or separate basis-size field. Cells, size, root and exclusions are computed from a coherent serialized sender state.

Epoch selection at the responder. A responder at the requested epoch selects its active state. An ahead responder selects that epoch's retained frozen batch as its raw and candidate set, with no exclusions, its cumulative overlay root as root vote and the preceding overlay root as prefix. A behind responder abstains. A POLL with a different frozen-prefix root from the selected state receives no reply, decode or fetch. Historical serving is read-only and retains no cell workspace or inbound state after the call. It creates no Pending state or local tally changes.

Responder processing. Every POLL undergoes structural checks, including its cell grammar and symbol-0 count, before any root-match shortcut. If its root equals the selected responder overlay root under the same prefix, the responder sends REPLY_MATCH without peeling. Otherwise it peels once against the selected raw set, using active A or the historical batch, never Pending. Successful decoding yields responder-only IDs in `missing` and poller-only IDs in `wanted`. These lists are strictly ascending, nonzero and disjoint. The responder sends the complete REPLY_DIFF if it fits the message limit; otherwise it sends fallback cells without truncating or fragmenting the lists.

An active responder fetches eligible positive decodes from the poller using ordinary `FetchBody`, including positives from an ordinary incomplete peel. A historical responder never starts a fetch. An explicit peel contradiction discards all effects of that attempt before fallback. Inbound polls provide no item vote, root evidence or sizing sample because the receiver did not select their senders. Inbound cells are neither retained nor rescued.

Fallback length. Write $S(d) = \mathit{RequiredSymbols}(d)$ and let $c_{\max}$ be the shared cell cap. For a prefix POLL the fallback length is

$$m = \min(c_{\max}, \max(2n, S(|\mathit{setSize}_r - \mathit{setSize}_q|))) .$$

For a probe, inspect the residual against the selected responder raw set before peeling. Let u be the smallest actually sent probe index with a zero residual cell, or the largest sent index if none is zero. The fallback length is $m = \min(c_{\max}, S(u))$, using the index itself. The fallback Ribbit prefix carries no partial ID lists, residuals or bitmaps. At the cap no further prefix growth is possible.

Candidate proof and round capture. The MATCH proof contains two little-endian 64-bit exclusive-or (XOR) reductions, one for each domain byte $b \in \{0, 1\}$:

$$\bigoplus_{x \in \mathrm{IDs}(W)} \text{SipHash-2-4}_{nonce}(b \,\|\, \text{LE64}(x)).$$

LE64 denotes little-endian encoding in a 64-bit word. The nonce's two eight-byte halves supply the little-endian SipHash key. Each distinct candidate ID is hashed once; the empty proof is zero. For historical service the candidate IDs are those of the selected frozen batch, excluding its prefix. This proof is algebraic set evidence and does not authenticate bodies or peer identities.

All polls of a round are sent from one captured state, tip, epoch and prefix. One shared capture holds advertised admitted IDs with incarnation and candidate bit, plus live Pending identities, at most $M + P$ records; it holds no body handles. Each slot stores its expected 16-byte proof and references that capture. A delayed poll may share it only if the captured admitted, candidate and Pending state is unchanged; otherwise it waits for the next round. MATCH is checked against the stored proof even after a current-root change. A valid old-root MATCH retains captured item-vote eligibility but cannot score a different current root.

Difference validation and reconstruction. Let A_{cap} be the captured raw advertised set and X_j the peer's exclusions. The poller checks sorted nonzero lists, disjointness, $\mathit{wanted} \subseteq A_{\text{cap}}$, $\mathit{missing} \cap A_{\text{cap}} = \varnothing$, and the checked cardinality equation

$$\mathit{setSize} = |A_{\text{cap}}| + |\mathit{missing}| - |\mathit{wanted}|.$$

The reconstructed raw set must have that size and contain all exclusions. The peer's asserted candidate-ID set is

$$\widehat{W}_j = \big((A_{\text{cap}} \setminus \mathit{wanted}) \cup \mathit{missing}\big) \setminus X_j.$$

Reconstruction starts from raw A_{cap}, including locally excluded IDs. A captured Pending ID absent from A_{cap} is present at the peer exactly when it occurs in `missing`. Invalid assertions are rejected before fetching or recording votes. Valid lists remain unauthenticated peer assertions. The poller fetches eligible `missing` items; `wanted` only describes peer absence for votes and triggers no poller body-send action. Each side fetches its own positive discoveries through the existing host path.

Fallback decoding and rescue. At reply entry capture the current *Ref* and peel received cells minus its encoding without changing that input reference during the attempt. A fully decoded reply reconstructs

$$\widehat{W}_j = \big((\mathit{Ref} \cup U_j^+) \setminus U_j^-\big) \setminus X_j.$$

After peeling, eligible positives may start fetches and enter Pending. An ordinary incomplete reply retains indices, exclusions and one invariant residual array,

$$\mathit{residual}_j[i] = \mathit{raw}_j[i] - \text{Encode}_i(\mathit{Ref}),$$

with count subtraction and syndrome XOR. Net reference membership gains or losses update that invariant exactly once at scheduled indices. A Pending-to-A transfer preserving reference membership needs no delta. Each attempt peels one transient copy, discards private subtractions and leaves the invariant buffer controlled only by reference deltas.

Touched replies become dirty and are retried in poll order until none is dirty. Rescue can discover donors and fetch more bodies, but creates no new item observations and never upgrades arrival abstentions, even within the same application programming interface (API) call. Successful replies release cell workspace; incomplete replies survive only until round end. Admissions and root changes do not abandon eligible captured item evidence. No cells or captures survive epoch rollover or tip cancellation. There is no work cap or proportional-work claim; attempts per call are bounded by the retained-reply bound times one plus the number of reference changes in that call.

K.2 FetchBody, tokens and Pending lifetimes

Decoding never inserts a transaction into R. Ordinary `FetchBody` uses `STANDARD_POLICY`; the host parses and binds the returned hash to the requested short ID under the request's tip key, validates

the body and reports the result. The combined in-flight, awaiting-admission, policy-waiting and FINAL-in-wait union has one reservation per ID, bounded by $P = 2M + 32$. At most one in-flight or awaiting operation exists per ID. Up to $\min(D, 8)$ usable donors are kept in discovery order, then PeerID order; failed or disconnected donors leave usable slots, and known alternatives are retried immediately. Algebraically consistent positive discoveries identify donors irrespective of exclusions; a reconstructed peer candidate set can also establish alternate donors for live Pending IDs.

- A matched admitted result retains its token, expected hash and generation while awaiting the matching host admission callback; that callback transfers the complete Pending item state without forcing IN.
- A not-found result or unbound invalid body is a donor failure; the ID leaves speculative reference membership and enters probation, and subsequent donor claims cannot restore that membership before actual admission.
- A matching-ID body that fails consensus validation is rejected for the rest of the tip lifetime and its donor is reported; a live FINAL-in decision additionally produces a protocol anomaly and never authorizes admission.
- A matched POLICY refusal preserves the live item state and a validated body descriptor in policy-waiting state; the node votes NO locally because the item is unadmitted, while eligible remote votes can still move its preference and confidence.
- A RESOURCE or CONTEXT refusal is nonpunitive and does not establish consensus invalidity; reserved FINAL-in or validated conflict-wait state is retained for bounded retry under the specified lifecycle.

A policy-refused nonconflict item reaching FINAL-in triggers `CONSENSUS_ONLY` fetching and admission subject to consensus validity. A validated conflict alternative can trigger it earlier on effective preference IN (Appendix E). A live FINAL-in Pending item vetoes freezing until actual admission, consensus-invalid disposition or a lifetime-ending reset. Other Pending items do not veto freezing, and rounds continue during body transfer and validation.

Fresh Pending starts OUT with zero counters and no confidence or FINAL status. Matching admission preserves that lifetime and capture eligibility. Loss of ordinary support discards an unsupported nonfinal lifetime; later refetch starts fresh. Removal and readmission also creates a new admitted incarnation. Preference or confidence alone never cancels a fetch or creates an epoch rejection mark. Duplicate or stale tokens are ignored; disconnect cancels an in-flight donor token and retries an alternative, while awaiting-admission callbacks remain required unless a tip reset cancels their generation. The host owns fetch timeout completion. Same-tip epoch rollover preserves bindings, reservations and callback obligations while resetting epoch-local tally state.

K.3 Admission authorization and notification ordering

Majority-selected replacement. A matched POLICY-refused body with a validated direct conflict to a held body is a conflict alternative within its existing Pending lifetime. When its cumulative majority counters yield effective preference IN, issue or reuse `FetchBody` in `CONSENSUS_ONLY` mode without waiting for FINAL. If the descriptor arrives after the majority transition, reevaluate the trigger then. An ordinary fetch already in flight is not duplicated; its policy-refused completion can trigger the retry. Nonconflict policy override still requires FINAL-in. A mempool first-seen conflict is policy rather than consensus invalidity.

The host checks current admission authorization immediately before committing an override. Its pure serialized query, `GetAdmissionAuthorization`, takes the live fetch token and exact expected hash. ALLOW requires a currently authorized consensus-only operation with its live body/lifetime binding, a FINAL-in or validated-conflict effective-IN trigger, and no opposing FINAL or frozen-winner constraint. A changed preference, epoch or opponent decision can withdraw permission while the fetch is in flight. The query grants no consensus-validity bypass.

The host revalidates consensus and resources, then serializes the authorization check, atomic replacement and callbacks without an intervening core event. It removes actual held direct conflicts and ordinary host-invalidated dependants, then admits the selected valid body. Notifications are ordered: target admitted/matched result, body-specific loser removals, then target admission callback. Loser removal preserves the target's Pending lifetime, reservation, descriptor, binding and votes. Failed validity, resource or context checks leave the held body in place and do not fabricate admission. A withdrawn authorization returns CONTEXT; a merely policy-based refusal while authorization still holds is a host-contract error.

K.4 Exact confidence representation

Configuration floating-point inputs denote their exact encoded finite binary values. Convert them to rationals before forming p_0 or the error budgets. Null probabilities, score ordering and decisions use exact arithmetic. A score can be represented symbolically as a negative logarithm of a product of exact tail probabilities; displayed decimal logarithms are not decision arithmetic.

The representation is finite. Let $q = H_K(a_K)$, so $0 < q < 1/2$, and let R_b be the least integer $r \geq 3$ with

$$q^r \leq b\, 2^{-(r-2)}.$$

Store exact thresholds before R_b; every all-success path qualifies at every $r \geq R_b$, since $2q < 1$. Items have finalized by then. A root run can continue while empty W or FINAL-in Pending prevents emission: at R_b its protocol count saturates and an exact qualified-tail tag replaces further score products. Before R_b, qualification is the current length's predicate without a qualification latch. A round without success, current-commitment change, K change, epoch rollover or tip reset clears root qualification, including this tag. Calibration storage and exact score payloads count toward allocated protocol memory.

K.5 Analytic reference calculation

The reference method evaluates integer binomial probability mass function (PMF) numerators with denominator 5^K at $p_0 = 3/5$. Success support is restricted to $Y \geq a_K$ without renormalization. Exact polynomial powers bound the inclusive probability of scores at least the observed $re_K(Y)$, with score bins certified by integer powers rather than floating logarithms. A lower bound above $(\varepsilon/2)2^{-(r-2)}$ rejects that length; an upper bound at or below it accepts it. Refinement continues until the comparison is certified. A unanimous observation is the unique maximum-score atom with probability p_0^{Kr}. The same atom handles rounded percentages that yield $Y = K$. This is an analytic verification procedure for the reference decisions, not an empirical sizing-table generator or a materialized production confidence table.

K.6 Exact sizing and empirical table generation

A timely verified MATCH contributes one uncensored sizing sample with start and current difference both zero; this does not assert equality of raw advertised sets. A valid DIFF supplies exact list-count samples

$$\widehat{\Delta}_{\text{start}} = |\mathit{missing}| + |\mathit{wanted}|,$$
$$\widehat{\Delta}_{\text{now}} = |\mathit{missing} \setminus \mathit{Ref}| + |\mathit{wanted} \cap A|,$$

using current sets at reply entry before Pending effects. These count the asserted lists and do not authenticate holdings. Fallback REPLY estimates use pre-peel residual emptiness; inbound polls and later rescue supply no samples.

Fallback estimator. For indices $i \geq 1$, write $p_i = 2/(i+2)$. Let $\mathcal{E}$ and $\mathcal{N}$ be the pre-peel empty and nonempty nonzero-index cells, and let L be the larger of the number of distinct items decoded on the first attempt and the absolute symbol-0 count difference. Maximize

$$\sum_{i \in \mathcal{E}} \Delta \ln(1 - p_i) + \sum_{i \in \mathcal{N}} \ln\bigl(1 - (1 - p_i)^{\Delta}\bigr)$$

over integers $L \leq \Delta \leq 2M + P$, breaking ties toward smaller values. This is an approximate maximum-likelihood estimate: cells are correlated and mixed-sign cancellation is treated as negligible. It never authenticates set size. A maximizer above the domain is censored at L; no unbounded search is performed.

If all nonzero-index residuals are empty and symbol 0 is zero, the estimate is zero. If no received residual cell is empty, record a censored lower bound

$$\mathit{lowerSizingBound} = \max\{L, \mathit{InverseHalfCDF}(m)\},$$
$$\mathit{InverseHalfCDF}(m) = \min\{d \in \mathbb{Z}_{\geq 0} : F_d(m) < 1/2\}.$$

For a symbol-0-only probe, a zero residual gives zero and a nonzero residual uses this censored floor. The inverse uses the conservative step CDF, not inverse median interpolation, scanning grid intervals and then the scaled tail. A hostile floor can exceed $2M + P$; record it without an empty-domain search or peer penalty and clip requests to the cell cap.

Each fallback arrival records a current-reference estimate for the next request and a round-start estimate for reusable history. The latter undoes only reference additions during this round still present at processing time, using a list bounded by P. Moves preserving reference membership are omitted and removals drop the ID. If that list would overflow, use the current estimate as the start sample for the rest of the round. Add-back is solely for estimation and creates no body or evidence effects. Neither sample is rewritten by rescue.

Empirical decoding table. The sizing table, distinct from exact statistical confidence calibration, records Ribbit prefix-length quantiles for differences d on $\{0, \ldots, 64\} \cup \{\text{round}(64(11/10)^j)\}$ through 4096, with half-up rounding and a deduplicated endpoint. Each positive d uses 2000 trials of distinct uniform nonzero IDs with equiprobable signs and an independent reference peeler. SplitMix64 uses seed `0x534E4F5753544F52`; accepted IDs take their sign from the next output's low bit. The specified historical generator cap is $\lfloor (2^{21} - 100)/28 \rfloor = 74894$ cells, with cap-exhausted trials counted at the cap. Runtime replies use the current shared wire cap. The generator, seed, table and digest are to be frozen together after approval; no completed table generation or performance result is asserted here.

Stored quantiles are 5%, 10%, ..., 95% and 100%, using 1-based rank $\lceil qN \rceil$. The responder function $S(d)$ is the 50% quantile, rank 1000, interpolated linearly and rounded up, extended proportionally beyond 4096; $S(0) = 1$. The conservative $F_d(n)$ is the largest stored quantile whose length is at most n, zero below the first quantile and one at or above the stored maximum. Between grid points use the next larger d; beyond 4096 scale all quantile lengths by $d/4096$ and round up. The zero-difference row has every quantile at length 1.

Request selection and growth. For current samples define $\widehat{P}(n)$ as their mean $F_{\widehat{\Delta}}(n)$, taking censored samples at their recorded lower bounds. Minimize $n/\widehat{P}(n)$ over the union of stored quantile lengths and S for those samples, capped at $c_{\max}$, with ties to smaller n. Positive length divided by zero costs infinity; if all candidates have zero probability, choose the smallest before applying growth rules. At most $21D$ candidates exist before deduplication.

With a prior prefix n_{prev}, enforce

$$n_{\text{next}} = \min(c_{\max}, \max(n^*, 2n_{\text{prev}}))$$

if the upper-median sizing sample is censored, or if at least one fallback cell reply arrived and none decoded on arrival or rescue and no MATCH or DIFF succeeded. Order a censored sample above an uncensored sample of equal value when taking the upper median. Otherwise usable samples, including MATCH zeros, recompute the request and may shrink it. A round without usable samples

retains the previous prefix.

Probe only when the epoch has no prior prefix and no usable first-round history. After a probe with a censored or unusable upper-median sample, the first prefix is the largest actually sent probe index plus one, clipped to the cap; otherwise use the estimator. Once a prefix exists, never return to probe mode within that epoch.

Retain two banks of at most 1024 samples each: current-round samples and the tip's epoch-1-round-1 history. For epoch 1 after the next tip transition, individually scale that saved history by $|R|_{\text{now}}/|R|_{\text{then}}$ and round up before applying the same selection rule. Missing usable history or previous $|R| = 0$ permits a probe. Same-tip epoch rollover clears active sizing and prior-prefix state and starts with an initial probe; later epochs neither consult nor overwrite the reusable bank. No third bank or cross-round change log is retained.

K.7 Scheduling and timing eligibility

Correlated complete malformed or invalid-proof arrivals count for timing but not votes, and structurally valid incomparable-prefix replies also count for timing. Uncorrelatable bytes cannot satisfy a slot. Late or stale replies are discarded entirely. Captured fallback time does not move with later measurements. Equal-time replies precede timers, in peer-ID order; a late timer evaluates once at actual time without catch-up. Freeze evaluation occurs only after round-end scoring. All-answered early ending remains available even with a retained larger duration.

K.8 Resource envelope and attributable faults

The compatible connected peer population is at most 1024. Let $B = 2^{21}$ bytes, $1 \le M \le \lfloor (B - 256)/8 \rfloor$, $P = 2M + 32$ and $F = 8M + 128$. The union of live and archived bodies has at most M bodies and M distinct IDs, with one retained record per archived body and at most M nonempty frozen batches. Item state is bounded by $M + P$. Probation, rejection, local-refusal and unavailable-source stores each have bound F. Exhausting these protection stores suspends new unadmitted claims and fetches until tip reset while preserving awaiting-admission obligations and already reserved policy-waiting or FINAL-in decisions; ordinary valid host admissions remain allowed.

The maximum symbol index is $I_{\max} = \min(2^{39} - 1, 64M + 4096)$. Conservative fixed header bounds are 156 bytes for POLL, 140 for fallback REPLY and 149 for DIFF, using maximum CompactSize widths. The shared cap is

$$c_{\max} = \min\left(I_{\max} + 1, \left\lfloor \frac{B - 156}{28} \right\rfloor, \left\lfloor \frac{B - 140 - 8M}{28} \right\rfloor\right).$$

Actual serialization must also fit B. DIFF has at most M IDs in each list and a combined exclusion/missing/wanted envelope of $\lfloor (B - 149)/8 \rfloor$ IDs. Sparse probes allocate by their index count, not their largest index. Parsing checks grammar, bounds and arithmetic before allocating message-owned collections.

At most K replies and $Kc_{\max}$ retained cells survive within a round, with one residual array per reply and one shared transient working buffer. The shared round capture is bounded by $M+P$. Actual allocated capacities, indices, padding, shared handles, exact calibration and variable payloads count toward core memory; host body payloads are reported separately. Conflict checks scan bounded retained descriptors and do not store a pairwise edge graph, transitive groups or a historical loser table.

Report only individually attributable faults: malformed grammar, oversize declarations, overflow, raw symbol-0 count differing from set size, infeasible raw counts, invalid exclusion/list structure, candidate-proof mismatch, or body-fetch faults. Raw structural checks precede reference-dependent peeling. Ordinary stalled peels are retained without penalty; explicit peel contradictions are discarded without peer penalty because another source's fabricated Pending ID may have contaminated the reference. Nonanswers are never penalized. The core has no reputation or poll-rate counters; rate limiting and discouragement belong to the host.