CONCEPT PAPER / V0.2 / JULY 2026

Accountable wide-area pipeline inference

A proposed protocol for layer-routed inference, co-signed handoff evidence, and batched settlement.

IMPLEMENTED + PROPOSED

Read this first.

Shardwood is a proposed coordination, evidence, and settlement protocol for pipeline inference across independently operated GPUs. The current implementation is a concept website and analytical simulator. The GPU runtime, scheduler, Causal Handoff Receipts, contracts, token, and live network are not deployed. This paper makes that boundary visible on every page.

SYSTEM THESIS

The chain settles work.

It does not execute the model.

The VPS coordinates routes and evidence; contributor GPUs perform inference offchain.

IMPLEMENTED

Concept interface

Website, visual identity, ShardLab simulator, and honest simulated labels.

PROTOTYPE TARGET

Two-node runtime

Qwen3-0.6B correctness path, then a Qwen3-4B pooled-VRAM test.

PROPOSED

Evidence + settlement

Receipt schema, witness replay, fee accounting, and Merkle payout roots.

NOT DEPLOYED

Token + contracts

No live token, bond vault, escrow, payout, or dispute contract exists.

ESTABLISHED + PROPOSED

What is new - and what is not.

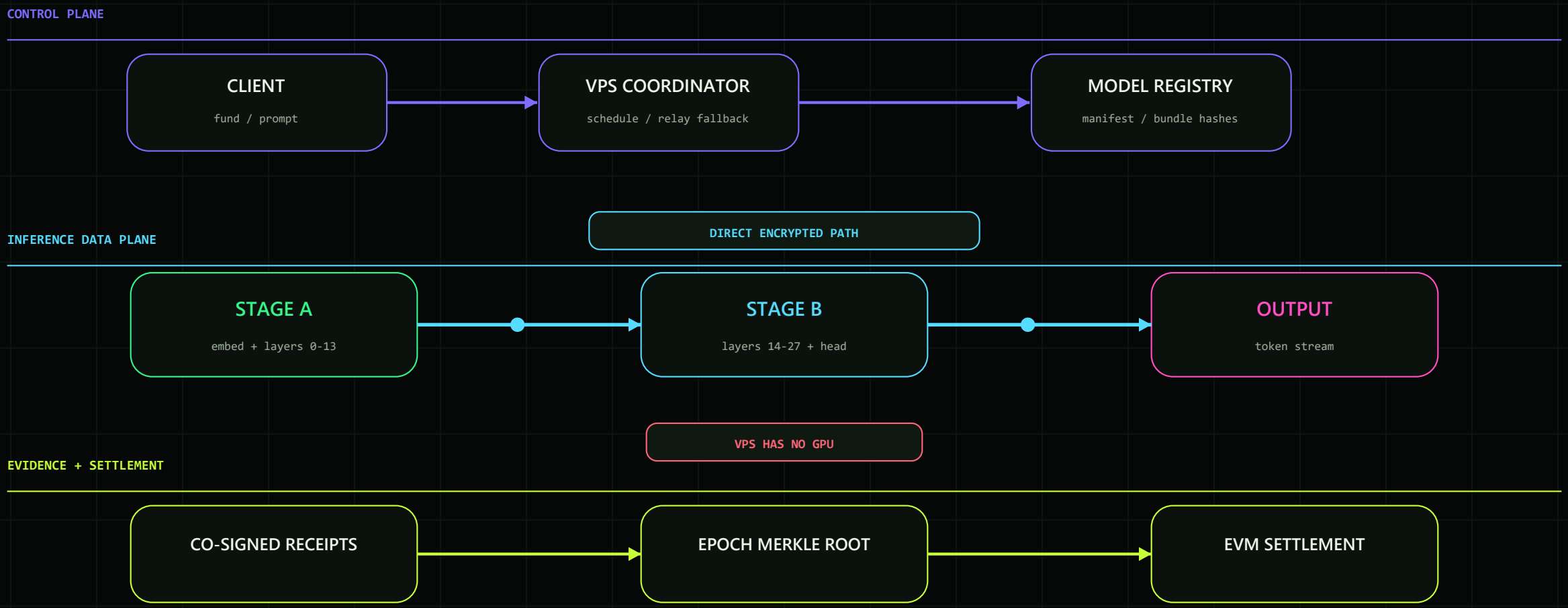
Layer partitioning, pipeline parallelism, volunteer inference, speculative decoding, signatures, and Merkle proofs have prior art. Shardwood's proposed contribution is the complete accountable path from an independently operated model stage to adjacent handoff evidence and batched EVM settlement.

	ONE OPERATOR	MANY OPERATORS
COMPLETE MODEL	**LOCAL / HOSTED** ESTABLISHED One owner runs the complete model.	**WORKER MARKETPLACE** ESTABLISHED Many operators each serve complete-model jobs.
PARTITIONED MODEL	**DATACENTER PIPELINE** ESTABLISHED One owner partitions layers across trusted hardware.	**SHARDWOOD GROVE** PROPOSED Independent operators form a short-lived accountable route.

NO CLAIM OF INVENTING MODEL SHARDING / THE ACCOUNTABLE END-TO-END PATH IS THE EXPERIMENT

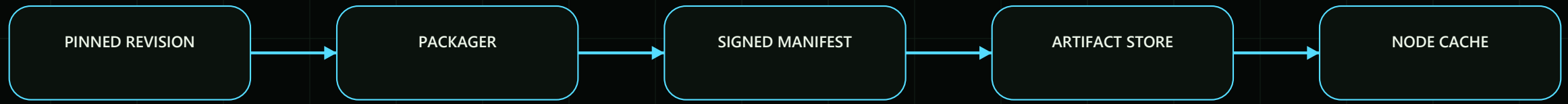
PROPOSED

Three planes. One accountable route.

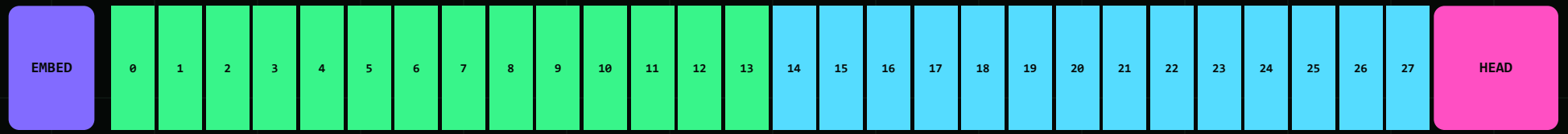


PROTOTYPE TARGET

A node downloads its assignment - not every layer.



QWEN3-0.6B / 28-LAYER TARGET LAYOUT



NODE A CACHE

Embedding + layers 0-13 + runtime metadata + exact manifest hashes.

NODE B CACHE

Layers 14-27 + final norm/output head + runtime metadata + exact manifest hashes.

EDGE MODULES AND TIED WEIGHTS CAN MAKE STAGES ASYMMETRIC / BUNDLES ARE VERIFIED AND CACHED ON SSD

ESTABLISHED + TARGET

The split is a constrained memory problem.

CONTIGUOUS FORWARD PASS

```
0 = b_0 < b_1 < ... < b_S = L
z_(s+1) = G_s(z_s; theta_s, KV_s)
G_s = F_[b_s, b_(s+1))
```

STAGE-LOCAL KV CACHE

For GQA layers, a useful planning estimate is:

```
M_KV = 2 * B * T * bytes * SUM(n_KV * d_head)
```

VRAM FEASIBILITY

```
M_s = M_weights + M_KV + M_buffers
      + M_workspace + M_runtime
M_s <= eta * V_free,s    where eta < 1
```

QWEN3-0.6B WORKED TARGET

28 layers / hidden 1024 / 8 KV heads / head dim 128. A 14-layer BF16 stage at batch 1 and 4096-token context has about 224 MiB of KV cache. A single decode boundary vector is 2048 bytes before framing. [1]

POOLED VRAM ALONE DOES NOT PROVE A FEASIBLE ROUTE; EVERY STAGE WORKSPACE MUST FIT.

PROTOTYPE TARGET

Qwen3-0.6B

1.52 GB package / 28 layers. Correctness test; sharding is not economically necessary.

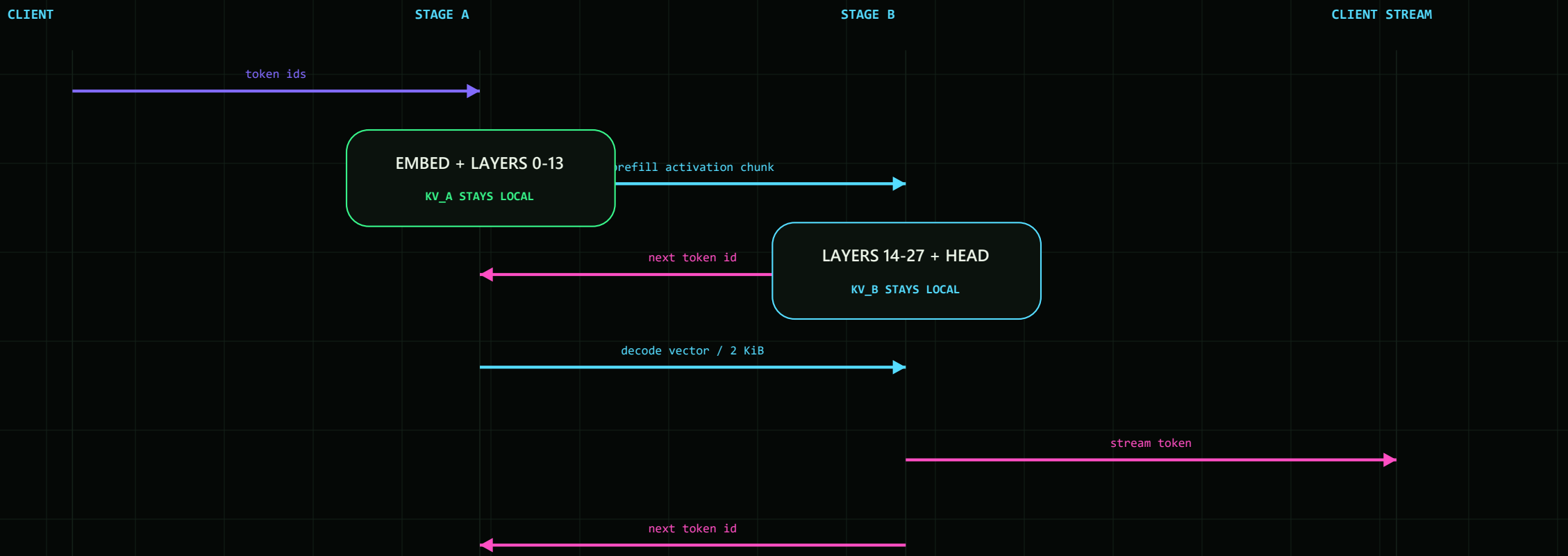
BETA TARGET

Qwen3-4B

8.06 GB package / 36 layers. A more credible pooled-VRAM demonstration. [2]

PROTOTYPE TARGET

One token through a Grove.



AUTOREGRESSIVE STEP

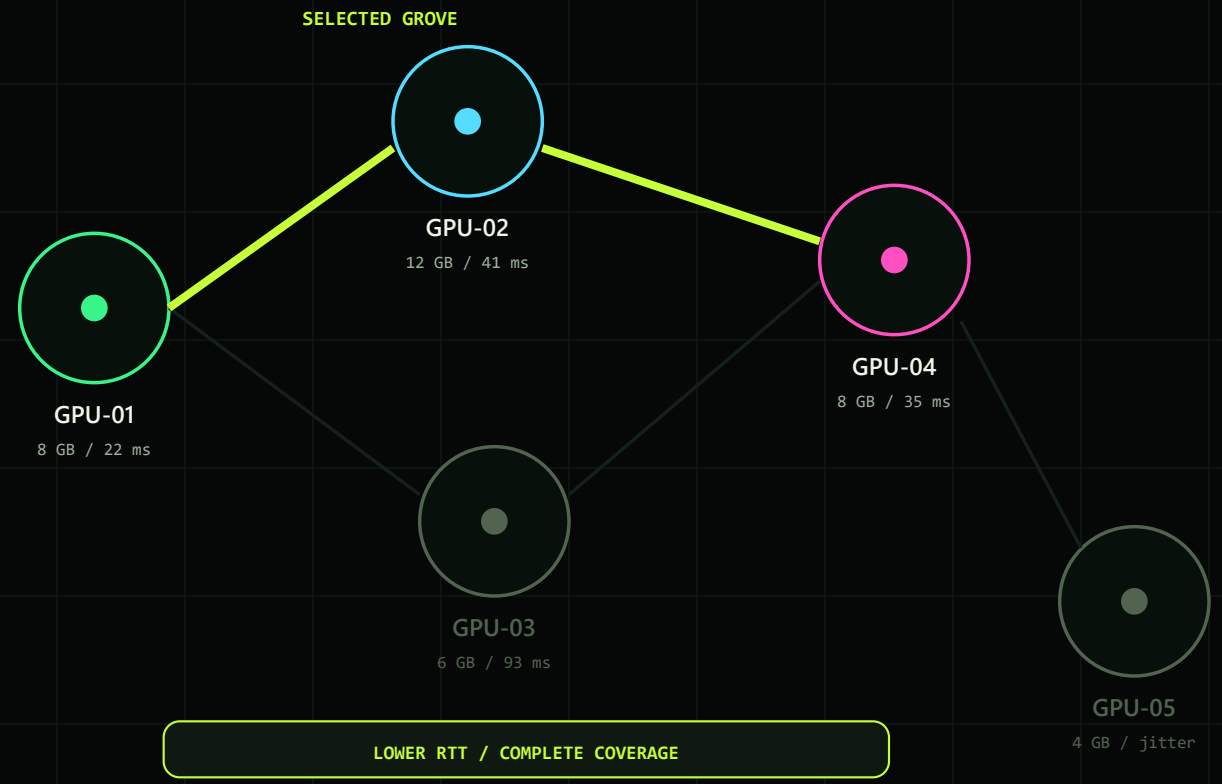
```
z_0 = Embed(x)
z_14 = StageA(z_0; KV_A)
logits = Head(StageB(z_14; KV_B))
```

KNOWN LIMIT

Every ordinary decoded token requires the complete stage path plus a token return. WAN latency therefore enters the critical path.

PROPOSED

Elastic routing is constrained optimization.



DISCLOSED ROUTE OBJECTIVE

```
min J(G) = wL*T_p95(G) + wC*Cost(G)
           - wR*log(R(G)) + wM*Move(G)
subject to:
  complete ordered layer coverage
  per-stage VRAM + runtime compatibility
  link bandwidth / RTT / reliability gates
```

RELIABILITY APPROXIMATION

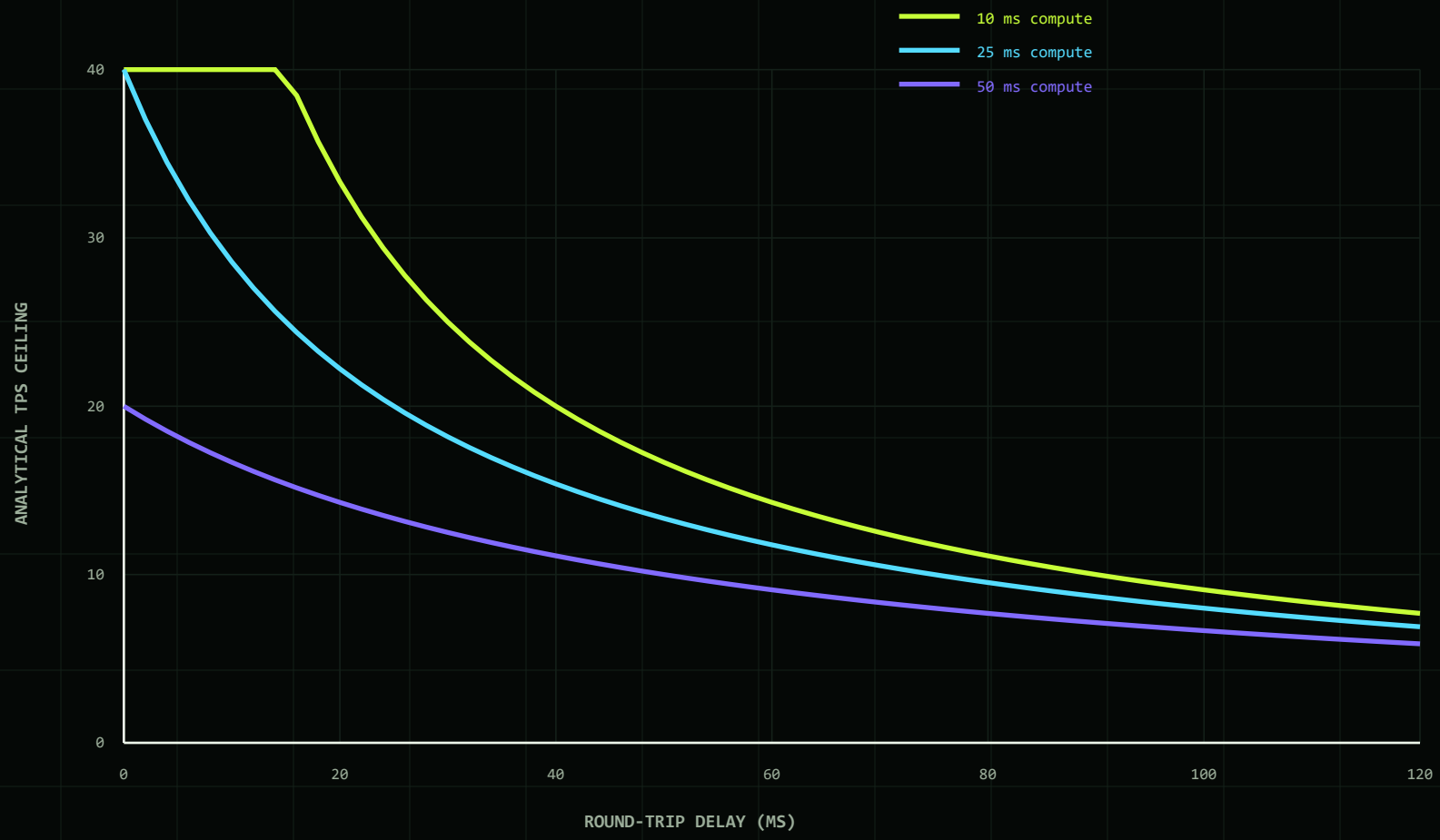
```
R_route ~= PRODUCT(p_stage) * PRODUCT(p_link)
p_stage(replicated) = 1 - (1 - p)^r
```

KNOWN LIMIT

The independence assumption is optimistic: ISP, region, relay, and coordinator failures are correlated. The production score must be calibrated from measured jobs.

ANALYTICAL / NOT A BENCHMARK

Latency is a design input, not a footnote.



PLAIN DECODE

```
T_step ~= SUM(C_stage)
       + SUM(RTT_boundary + bytes/BW)
       + protocol / sampling overhead
TPS_single ~= 1 / T_step
```

DIRECT PATH

Use the VPS for discovery and relay fallback; prefer an authenticated direct activation path once nodes connect.

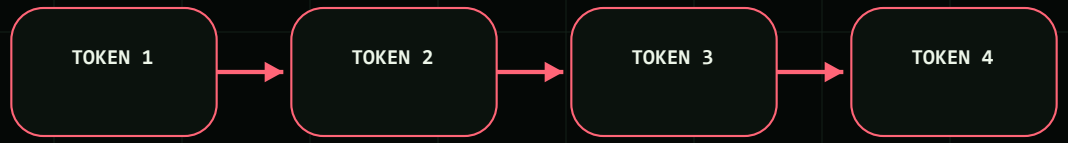
WHAT COMPRESSION CANNOT FIX

Lower-precision activations reduce bytes/BW. They do not remove propagation delay or correlated dropouts.

EXTERNAL METHOD / FUTURE TARGET

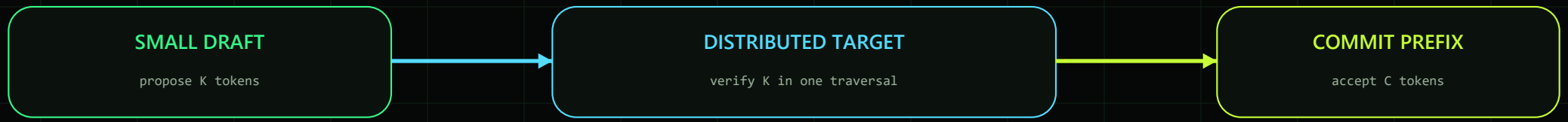
Trade one-token WAN loops for chunk verification.

PLAIN AUTOREGRESSIVE



Four serial target traversals. Network delay is paid each time.

SPECULATIVE VERIFY



EXPECTED COST

```
T_spec/token ~= (T_draft(K) + T_verify(K)
                + T_network + T_misc) / E[C]
WAN rounds per committed token ~= 1 / E[C]
```

TARGET USE

Qwen3-0.6B can become a future draft for the 4B target. Acceptance depends on prompts and sampling; no speedup is claimed until measured. [5][6]

PROPOSED FORMAT

Causal Handoff Receipts bind the path.

CANONICAL COMMITMENT

```
c_in  = H('SW/IN'  || job || step || salt || Canon(a_in))
c_out = H('SW/OUT' || job || step || salt || Canon(a_out))
r_k   = H('SW-CHR-v1' || domain || job || manifest
          || route || range || step || c_in || c_out
          || units || quote || prev || nonce)
sig_k = Sign(sk_k, r_k)
ack   = Sign(sk_(k+1), H('ACK' || r_k))
```

BINDS

Job, model revision, route, layer range, ordered input/output bytes, quote, meter, nonce, and previous receipt.

DOES NOT PROVE

Correct neural computation, unique physical GPU, privacy, non-collusion, elapsed energy, or output quality.

NODE SIGNATURES MAY BE VERIFIED OFFCHAIN; CHEAP ED25519 VERIFICATION IS NOT ASSUMED INSIDE THE EVM CONTRACT.

PROPOSED + KNOWN LIMIT

Sampled evidence is not full proof.

I

SIGNED ROUTE

Adjacent acknowledgement and ordered metering.

II

WITNESS REPLAY

Post-commit sample with deterministic/tolerant comparison.

III

ATTESTED / PROOF

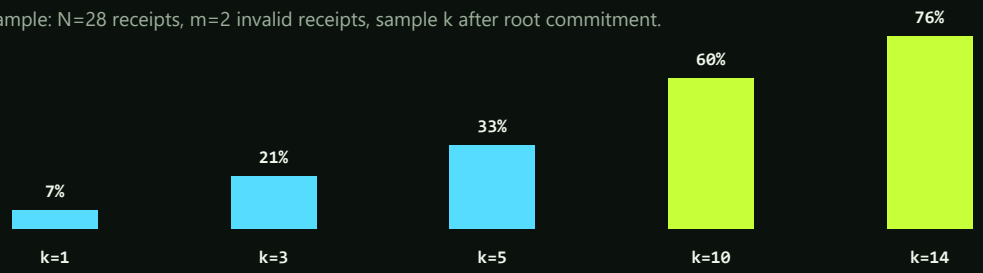
Future TEE or cryptographic evidence for selected operations.

NOT AVAILABLE

UNIFORM SAMPLE DETECTION

```
P_detect = 1 - C(N-m, k) / C(N, k)
```

Example: N=28 receipts, m=2 invalid receipts, sample k after root commitment.



CONDITIONS + RESIDUAL RISK

- Sampling must be unpredictable and post-commit.
- Replay must cover the claimed runtime and model revision.
- Cross-hardware BF16 may need a declared tolerance, not byte equality.
- Slash only objective faults: invalid signature, equivocation, or disclosed challenge nonresponse.
- A malicious unsampled stage can still escape detection.

PROPOSED / TESTNET FIRST

Batch the claims, not the activations.

CONSERVATION BEFORE TOKENOMICS

```
F_job = SUM(node_payout_i) + F_witness + F_relay
        + F_protocol + F_growth + F_reserve
deposit = F_job + refund
alpha_exec + alpha_witness + alpha_relay
  + alpha_protocol + alpha_growth + alpha_reserve = 1
```

BATCHED PAYOUT COMMITMENT

```
leaf_i = H('SW/PAYOUT/v1' || chainId || epoch
          || asset || payee || amount || nonce
          || receiptPathRoot)
R_epoch = MerkleRoot(leaf_0 ... leaf_n)
```

OFFCHAIN

Prompts, outputs, activations, node receipts, arithmetic, and witness evidence.

ONCHAIN

Escrow, epoch root, capped total, claim bitmap/nonces, challenge state, and withdrawals.

ECONOMIC LIMIT

Stable-asset fees are distinct from any future $SHWD bond. Staking is security exposure, not guaranteed yield.

ROBINHOOD CHAIN TESTNET: EVM-COMPATIBLE ARBITRUM L2 / CHAIN ID 46630 / ETH GAS. VERIFY CURRENT DOCS BEFORE DEPLOYMENT. [8]

BUILD SEQUENCE

Proof before protocol.

NO PERFORMANCE CLAIM BECOMES 'MEASURED' WITHOUT HARDWARE, ROUTE, COMMIT, PROMPTS, AND REPRODUCIBLE OUTPUT ARTIFACTS.

REFERENCES

[1] Qwen. Qwen3-0.6B model card and config. huggingface.co/Qwen/Qwen3-0.6B

[2] Qwen. Qwen3-4B model card. huggingface.co/Qwen/Qwen3-4B

[3] Borzunov et al. Petals: Collaborative Inference. arXiv:2209.01188

[4] PyTorch. torch.distributed.pipelining documentation.

[5] Leviathan et al. Fast Inference from Transformers via Speculative Decoding. arXiv:2211.17192

[6] Song et al. Speculative Decoding in Decentralized LLM Inference. arXiv:2511.11733

[7] leyten/shard. Apache-2.0 research engine; reported WAN results are self-reported.

[8] Robinhood Chain official docs. docs.robinhood.com/chain/connecting/

[9] RFC 8032 (Ed25519); NIST FIPS 180-4 (SHA-2); RFC 6962 (Merkle audit paths).

DISCLOSURES

- Concept paper v0.2; not a deployed protocol specification.
- No $SHWD token, public sale, contract, yield, or live node network exists.
- Analytical charts are not benchmarks or financial projections.
- Volunteer nodes are not inherently private; participants process plaintext or decrypted activations.
- Shardwood is independent and is not affiliated with, endorsed by, or sponsored by Robinhood Markets, Inc.
- Project site: public deployment pending.
- Public repository: release pending.