



SIGCOMM 2026

Connex: Endpoint Mobility Primitives for Dynamic LLM Serving

How should an in-flight stream continue when its endpoint moves?

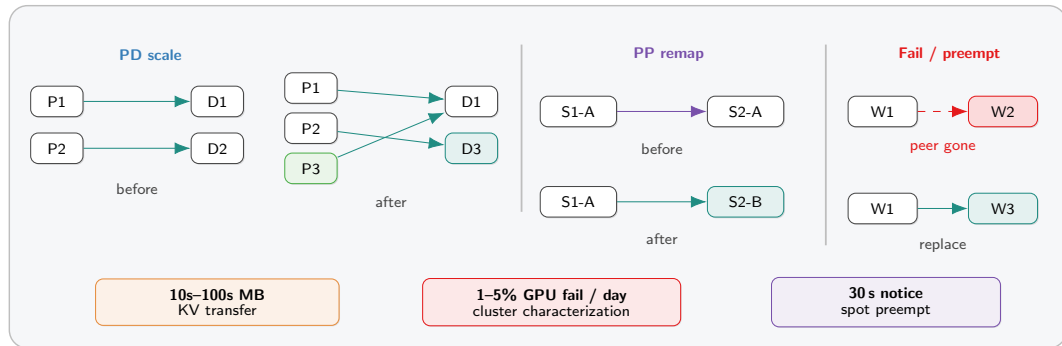
Speaker: Wuqiong Zhao

Yanying Lin · Vincent Liu · Tao Luo · Chengzhong Xu · Kejiang Ye

UCAS / SIAT · University of Pennsylvania · University of Macau

SIGCOMM 2026 · Denver

Serving elasticity ultimately changes who communicates with whom.



DistServe/Splitwise (PD); FlexPipe (EuroSys'26, PP); Jeon/Weng (GPU fail rate); SpotServe (30s notice).

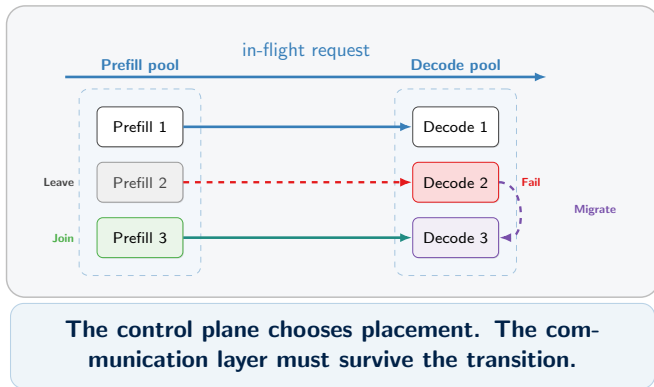
Scale, rebalance, preempt, and fail change edges, not only the worker set.

SCALE Prefill and decode pools grow independently.

REBALANCE Next PP stage or KV peer is remapped.

PREEMPT Spot capacity can disappear in 30 s.

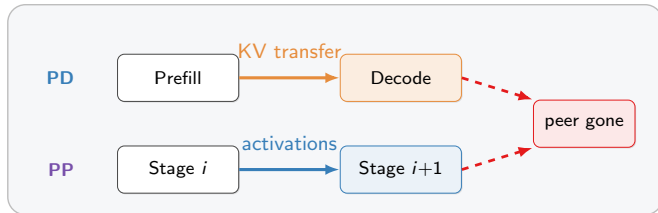
FAIL A departed peer breaks live edges.



An admitted request makes topology change stateful

Future routing is easy. Hard case: the changing edge already carries live request state.

- PD edges carry large KV transfers.
- PP edges carry ordered activations.
- Streams also carry token-level progress.



Architecture-dependent: KV,
activations, and/or stream progress.

Dynamic topology is not hard by itself. Dynamic topology + in-flight progress is.

DistServe (OSDI'24); Splitwise (ISCA'24); Mooncake (FAST'25)

High-performance APIs still bind correctness and performance to a relatively stable group.

Typical communication abstraction

Elastic serving reality

Fixed / explicit membership

→

Membership changes at runtime

Route / address tied to peer

→

Placement changes under a stable name

Connection lifetime $\approx$ group lifetime

→

Endpoint can be replaced mid-stream

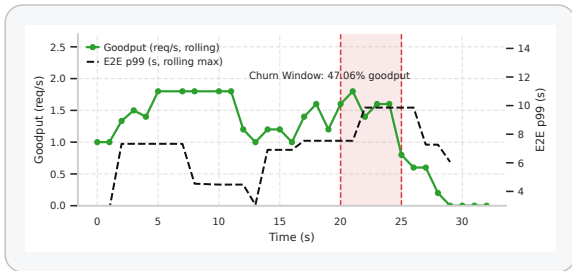
Stationary resource bottleneck

→

Recovery shifts congestion

For NCCL/ProcessGroup, a logical-topology change becomes a membership/rank change.

Finding a replacement is not enough. Communication state must converge across the transition.

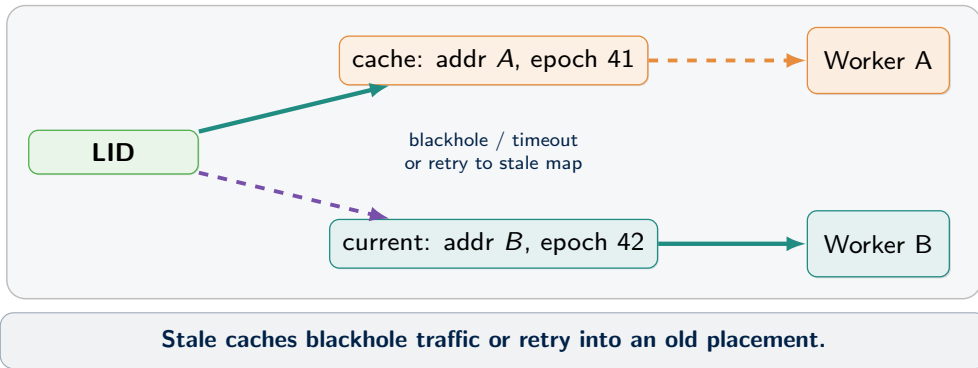


- 1 **47% goodput** in the churn window (Fig. 2).
- 2 **~2 s** detect/spawn vs **~8 s** elevated P99.
- 3 **Process-group reinit: 5–15 s.**

Disruption outlasts failure detection. The issue is state across the topology transition.

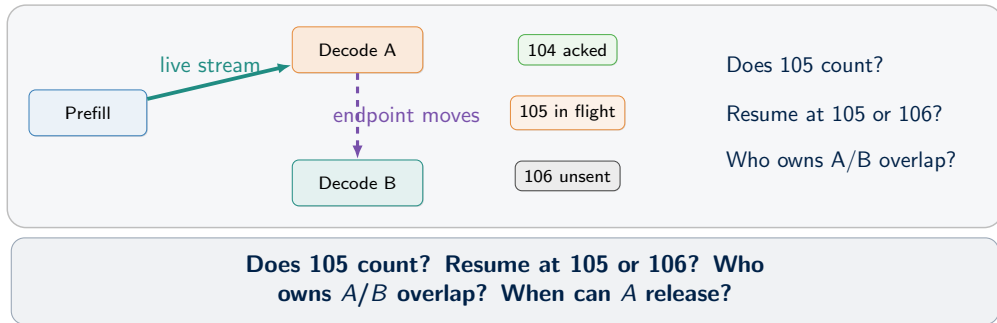
Challenge 1: Where is the logical peer now?

How do we bound stale routing without synchronizing the whole communication group?



Challenge 2: Where should an in-flight stream resume?

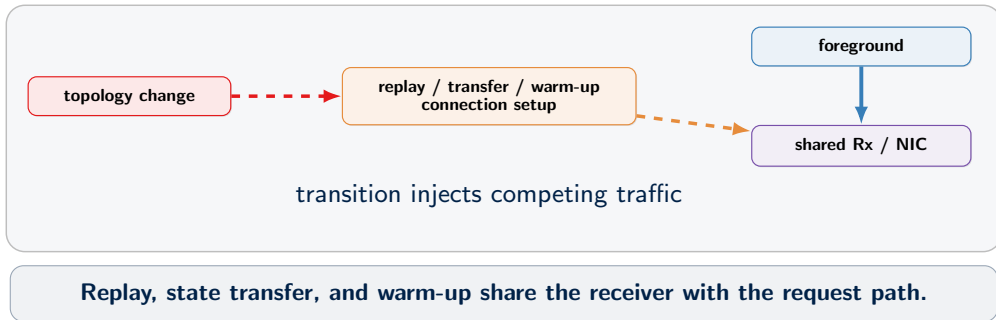
Knowing *B* is not enough. Progress across the move must have a named boundary.



Challenge 3: The transition itself creates a new bottleneck



Topology change injects competing traffic. Isolation is not ordinary congestion control.



Existing systems solve different pieces of this transition

Data movers, orchestrators, and Shrink each help. Live topology change still lacks one contract.

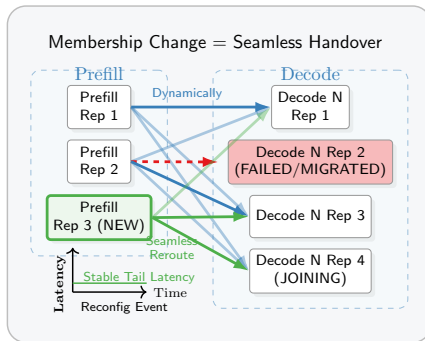
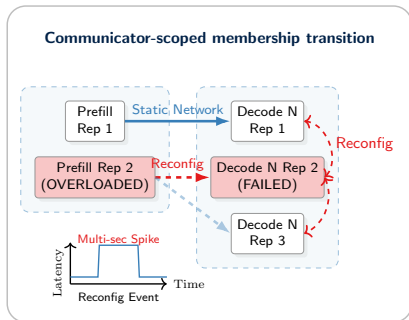
system	already provides	boundary that remains
NIXL	Fast async P2P + dynamic agent metadata	No successor for an already-posted transfer
Mooncake	KV-cache transfer / high-speed RDMA	No ordered stream continuation across replacement
NCCL Shrink	Communicator membership change	No LID or per-stream progress across newcomm
Dynamo	Worker discovery, routing, P/D handoff	Drain on removal; no mid-request switch
Retry / restart	Restores liveness after failure	Discards partial progress; amplifies tail

What is missing is a contract for communication across a live topology transition.

Shrink, Dynamo, and NIXL solve pieces of reconfiguration, not in-flight continuation.

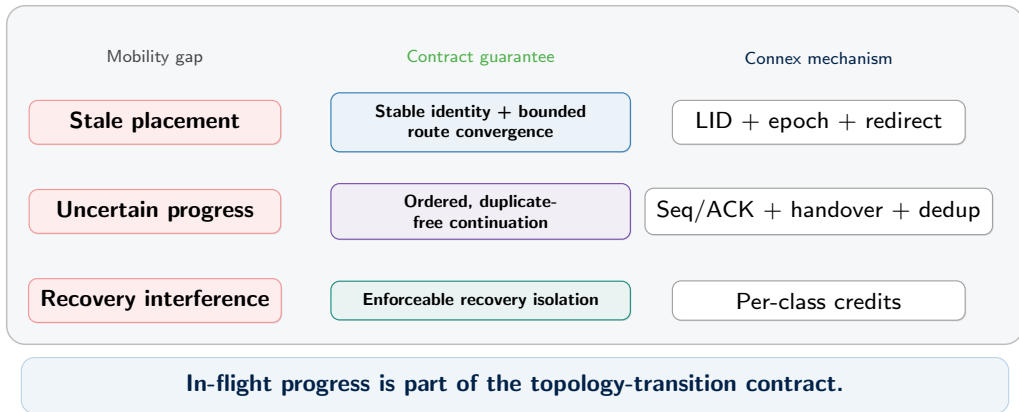
Communicator reconfiguration is not stream handover

Shrink overlaps on topology reconfiguration. Connex adds application-visible in-flight progress to the transition contract.

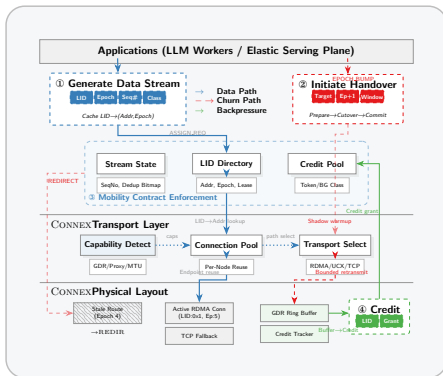


NCCL 2.27: `ncclCommShrink`. Membership change $\neq$ stream continuation.

The problem is continuity across a live topology transition. The triad enforces it.



Applications name a stable LID; placement and path selection stay inside Connex.



Three layers

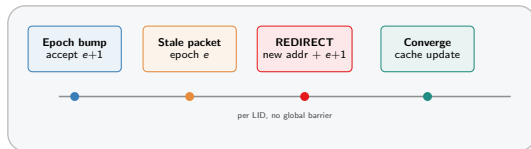
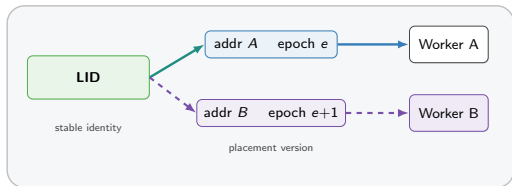
NAME Application sends to a logical endpoint.

RESOLVE Directory versions placement with an epoch.

SEND Payloads use pooled RDMA, UCX, or TCP.

Connex does not replace NIXL/Mooncake/UCX. It defines mobility semantics above them.

A stale epoch cannot supersede the current epoch. No global barrier.



LID is the stable name. Packets carry epoch. Stale packets get REDIRECT.

Epoch moves the route; it does not move stream progress.

Finding B is not enough. Decide from where B continues.

Route is current

Stale epoch cannot supersede.
Epoch advances monotonically per LID.

Progress is known

Highest contiguous Seq is committed.
Replay never re-delivers committed bytes.

Overlap is covered

New path covers the unacked window.
Then the old path may be released.

Routing decides where. Sequence state decides from where. Commit decides when.

Epoch moves the route. Handover moves the stream. Commit transfers ownership.

1 Prepare

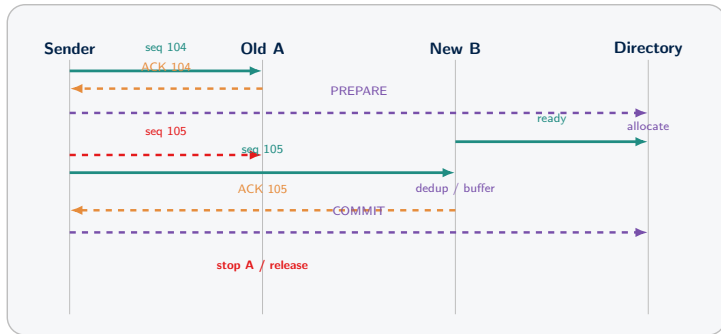
Install handover state;
establish the shadow path.

2 Cutover

Dual-path seq 105; *B* dedups.

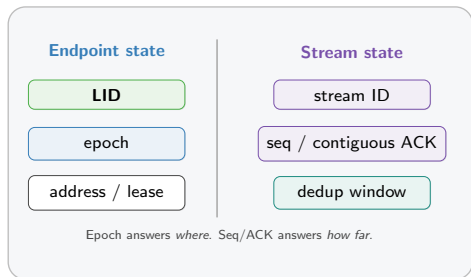
3 Commit

ACK 105, then release *A*.



A new worker does not imply a new stream

Epoch tells us where. Seq/ACK tells us how far.



Endpoint state and stream state stay separate

LID Stable name. A move does not rename the peer.

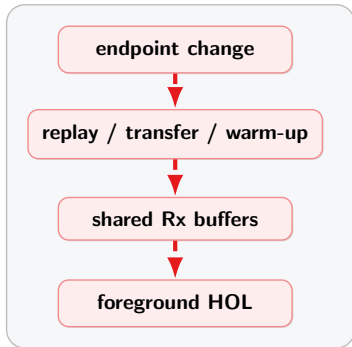
EPOCH Versions the mapping. Address and lease are only the binding.

STREAM Seq, contiguous ACK, and the dedup window stay with the stream.

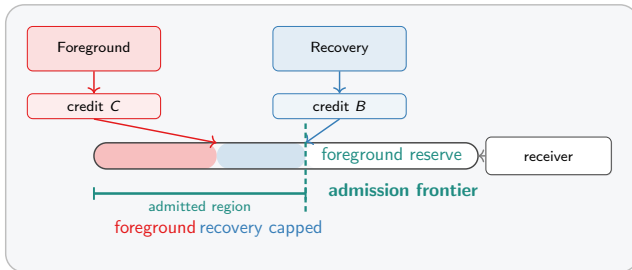
Progress resumes from the committed Seq, not from the new worker.

Design 3: Bound mobility-induced interference

Third guarantee: mobility must not wreck foreground performance.



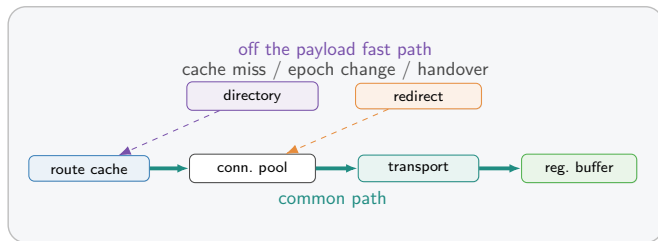
Replay and warm-up share the Rx.
Isolation keeps them from blocking
the token stream.



Per-class credits bound admission. Recovery is capped below receiver capacity, preserving headroom for latency-critical foreground traffic.

FlexPass (EuroSys'23); Credit-Based Transport (SIGCOMM'25).

Steady state reuses the route cache, connection pool, and registrations.

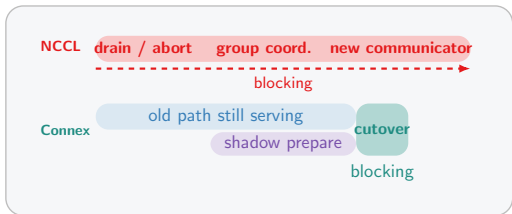


- 1 2000 endpoints; $\sim 1\text{k}$ updates/s.
- 2 Route-refresh p99 $< 10\text{ ms}$.
- 3 25 B header: $< 0.05\%$ on 64 KB.

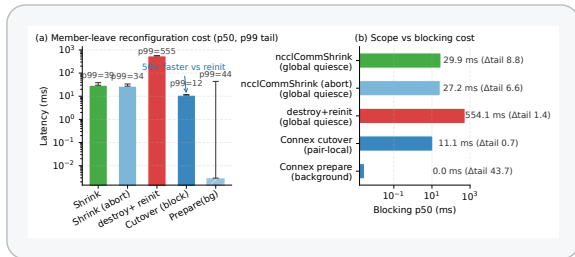
Directory and header tax stay off the payload fast path.

Mobility changes the reconfiguration critical path

Member-leave. Measured NCCL 2.27.5 Shrink. Connex blocks only on cutover.



Shrink: collective / safe point ABORT: collective + abort Connex: pair-local cutover



Who participates: communicator vs affected pair.

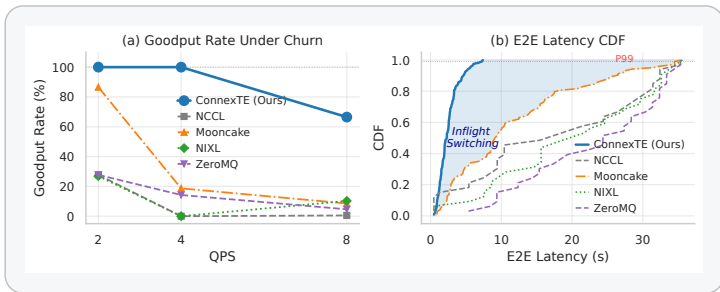
What blocks: newcomm vs final cutover.

554 → 30 → 11 ms

reinit / Shrink / Connex cutover

Does locality preserve serving under churn?

Pair-local recovery protects goodput and tail when an endpoint moves.

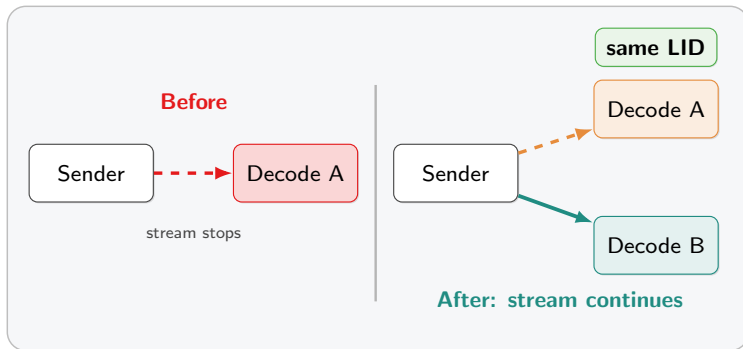


1 100% success @ QPS 4

Baselines 0–19%.

2 7.2 s vs 21–35 s P99
under churn.

Dynamic topology itself is not the hard part. Dynamic topology + in-flight progress is.



Connex makes in-flight progress part of the topology-transition contract.

**Keep the logical endpoint. Continue from known progress.
Bound the recovery impact.**

Dynamic topology + in-flight progress is the hard part.

Thank you.

How should an in-flight stream continue when its endpoint moves?

**Keep the logical endpoint. Continue from known progress.
Bound the recovery impact.**

Yanying Lin

yy.lin1@siat.ac.cn

<https://yanyinglin.github.io>

Speaker: Wuqiong Zhao

Yanying Lin · Vincent Liu · Tao Luo · Chengzhong Xu · Kejiang Ye

Backup: Why not just Shrink/Grow?

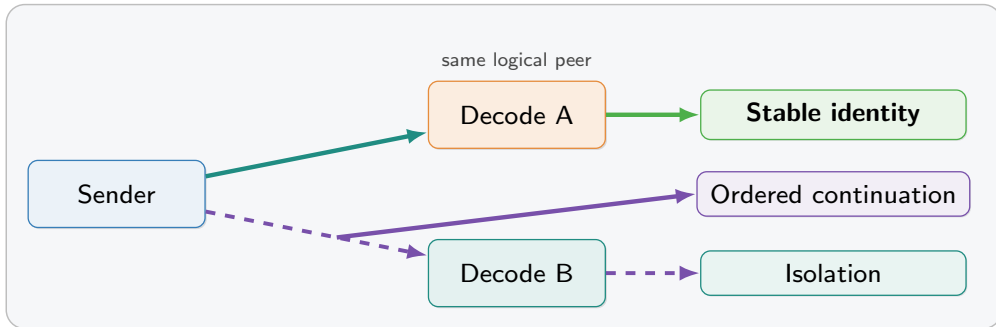
Shrink/Grow makes membership elastic. Connex makes an active stream mobile.

	NCCL Shrink/Grow	Connex
Unit	communicator	logical endpoint / stream
Coordination	participating ranks	affected pair
Transition	construct new communicator	update mapping + handover
Active traffic	drain/abort before transition	continues during prepare
State reuse	communicator-dependent	connection/MR pools by design
Continuity	membership only	seq/ACK + dedup

The gain is the abstraction boundary, not a faster communicator rebuild.

Backup: a mobility contract must preserve three things

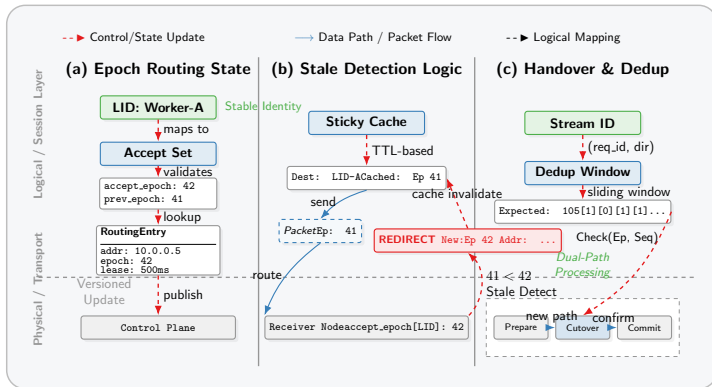
Identity, ordered continuation, and isolation must survive a placement change together.



This contract, not any one mechanism, is Connex's abstraction.

Backup: full protocol overview (Fig. 5)

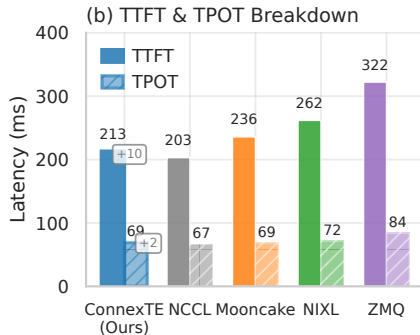
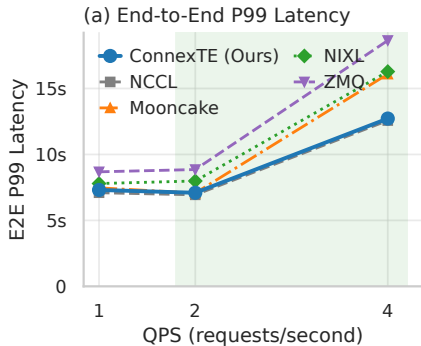
Epoch state converges the endpoint; sequence state preserves stream progress.



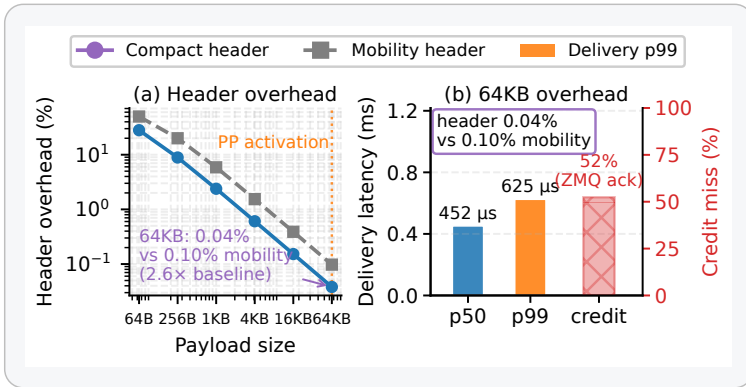
The LID stays. Progress resumes from the committed Seq. Recovery stays capped.



Identity stays stable. Progress advances once. Recovery remains bounded.



Compact header is about 0.04% of a 64 KB serving payload.



Backup: three causes over time (Fig. 3)

