LCMP: Distributed Long-Haul Cost-Aware Multi-Path Routing for Inter-Datacenter RDMA Networks

Anonymous Author(s)
Submission Id: 257

Abstract

RDMA-empowered cloud services are gradually deployed across datacenters (DCs) with multiple paths, which exhibit new properties of path asymmetry, delayed congestion signals, and simultaneous flow routing collisions, and further fail existing routing methods.

We present LCMP, a distributed long-haul cost-aware multi-path routing framework that aims to place RDMA flows on multiple inter-DC paths, achieving low-cost, low-latency, and congestion-responsive transmission. LCMP combines a control-plane path-quality score with compact on-switch congestion signals, where the former unifies quality assessment for asymmetric paths and the latter enables responsive reaction to path congestion. LCMP further resolves the simultaneous flow decision collision problem by filtering high-cost candidates, and performing a diversity-preserving hash inside the reduced set. On an 8-DC testbed, LCMP reduces median and tail FCT slowdown by up to 76% and 64%, respectively compared to state-of-the-art (SOTA) baselines. And large-scale NS-3 simulations under the 2000 km inter-DC scenario confirm similar improvements.

CCS Concepts: • **Networks** → **Routing protocols**; **Data center networks**.

Keywords: Data center networks, RDMA, Routing, Long haul, Multi-path routing

1 Introduction

Modern cloud services increasingly depend on geographically distributed deployments that span multiple datacenters (DCs) to provide geo-replicated storage[1, 2] and distributed machine learning training[2, 3], which impose stringent latency and throughput requirements while transferring large volumes of data across inter-DC links. To meet these demanding performance requirements, RDMA-empowered cloud services are being gradually deployed across DCs, leveraging RDMA's ability to offload the network stack to RNICs and bypass the kernel for ultra-low latency with minimal CPU overhead [4, 5]. However, as these RDMA flows traverse multiple inter-DC paths, they encounter new challenges including path asymmetry, outdated congestion signals, and simultaneous flow routing collisions that cause existing routing methods to fail[6–8].

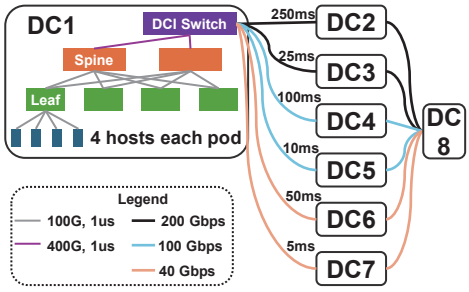
Many routing schemes were designed for the intra-DCs and rely on either feedback-driven reactivity[9–11] or randomized forwarding[6–8]. Both approaches suffer in inter-DC networks for two reasons. First, ***slow and outdated feedback signals***: congestion signals traverse long paths, so reactive decisions may act on outdated information. Second, ***path heterogeneity and asymmetry***: different from intra-DC links, topologically similar paths may differ greatly in propagation delay and link capacity in long-haul network while oblivious hashing or capacity-only metrics can misplace flows.

To illustrate, consider an inter-DC scenario (Fig. 1). From DC1 to DC8, there are six candidate routes (two high-, two medium-, two low-capacity), and each capacity class contains one low-delay and one high-delay path. When RDMA traffic is sent between DC1 and DC8, we observe two effects. First, a capacity-centric policy (UCMP) concentrates traffic on the high-capacity/high-delay paths (e.g., the DC1–DC2 link shows 17% utilization under UCMP vs. 6% under ECMP), leaving lower-delay capacity underused. Second, ECMP's random hashing can instead choose some low-delay links (e.g., DC1–DC6 and DC1–DC7 reach 30% and 27% utilization, respectively) while UCMP may avoid them entirely (0% in Fig. 1b). These placement choices directly raise median and tail FCTs. These observations motivate a routing-centric design that fuses stable path quality with timely congestion signals to guide per-flow placement.

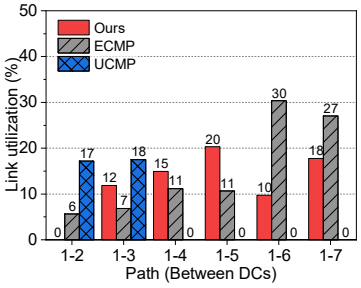
However, designing such a framework introduces three core challenges (details in §2.3):

- Ⓒ1 **Heterogeneous and asymmetric topology**: how can we define a compact "path-quality" score that captures both propagation delay and link capacity? (Solved in §3.2)
- Ⓒ2 **Slow and easily outdated congestion signals**: how can a datacenter interconnection (DCI) switch rapidly and robustly detect imminent congestion on inter-DC paths so that routing decisions remain effective despite long RTTs? (Solved in §3.3)
- Ⓒ3 **Simultaneous flow arrivals**: how can we avoid selection conflicts when many flows choose paths simultaneously? (Solved in §3.4)

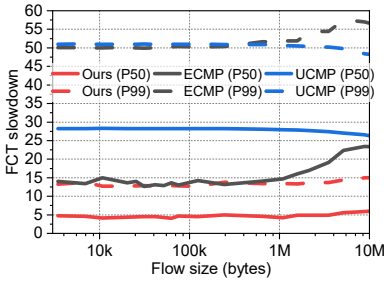
To address these challenges, we present LCMP, a distributed **L**ong-haul **C**ost-aware **M**ulti-**P**ath routing framework for inter-DC RDMA. LCMP fuses a compact, control-plane precomputed path-quality score C_{path} (encoding delay and capacity) with an integer-friendly on-switch congestion



(a) Inter 8-DC topology[1].



(b) Per-link utilization.



(c) Median and tail FCT slowdown Median and tail FCT slowdown for Web Search under 30% load using DCQCN.

Figure 1. [Motivation] Capacity–delay asymmetry causes ECMP and UCMP to make poor placement choices; LCMP balances utilization and reduces both median and tail FCT.

score C_{cong} (instantaneous queue level, short-term trend, and persistence). The switch computes a fused cost per candidate, filters high-cost suffixes, and performs a diversity-preserving hash within the reduced set.

Importantly, LCMP is orthogonal to end-host congestion control and requires only modest upgrades to DCI switches; end hosts and intra-DC fabrics remain unchanged. We evaluate LCMP on a small-scale testbed and with large-scale NS-3 simulations against ECMP, UCMP reproduction, and several ablations. Across heterogeneous topologies and bursty workloads (including a 2,000 km inter-DC scenario), LCMP substantially reduces median and tail FCTs; we further present sensitivity and ablation studies to justify our parameter choices.

Contributions. This paper makes three main contributions:

- We introduce LCMP, a distributed cost-fusion routing framework in long-haul inter-DC network that enables fast, line-rate routing decisions with low deployment cost.
- We develop a compact path-quality representation and an on-switch congestion estimator that together permit accurate comparison of heterogeneous inter-DC paths.
- We demonstrate that, in testbed and large-scale NS-3 experiments across realistic heterogeneous topologies (under the 2000 km inter-DC scenario). LCMP reduces median and tail FCT slowdown by up to 76% and 64%, respectively, compared to the SOTA routing baselines.

The rest of the paper is organized as follows. §2 presents background and challenges. §3 details the design of LCMP. §4 gives resource and feasibility analysis. §5 describes deployment considerations. §6 evaluates the system. §7 discusses limitations and future work. §8 reviews related work and §9 concludes.

2 Background & Challenges

2.1 Long-Haul RDMA Background

Remote Direct Memory Access (RDMA) is widely used in clouds because it bypasses the kernel and offloads the network stack to RNICs, delivering very low latency, high throughput, and low CPU overhead [4, 5]. RDMA workloads are latency-sensitive. They favor in-order delivery and they suffer when packets are reordered.

Operators increasingly deploy RDMA across geographically distributed datacenters to support geo-replicated storage, distributed ML training, and remote memory services [1–3]. Cross-region RDMA preserves RNIC-level performance benefits and simplifies application design. At the same time, it exposes RDMA flows to wide-area conditions that stress both transport and routing.

However, inter-DC links differ sharply from intra-DC links. Typical intra-DC propagation delays are on the order of microseconds; inter-DC propagation delays range from milliseconds up to hundreds of milliseconds. Provisioned capacities across inter-DC links are heterogeneous (tens to hundreds of Gbps). Topologies are sparser and less regular than leaf–spine fabrics. These differences change how routing choices affect performance. Below we summarize the key distinctions and their routing implications.

1) Large RTTs and outdated feedbacks. Inter-DC links span hundreds to thousands of kilometers; one-way delays grow from microseconds to milliseconds and RTTs can be tens to hundreds of milliseconds. Long RTTs make controller- or host-driven feedback slow to reflect current congestion, so routing decisions that rely on recent global signals become outdated.

2) Path asymmetry and heterogeneous topology. Inter-DC topologies are sparser and less regular than intra-DC

[1]The propagation delay of 1000 km is 5 ms = $\frac{1000\text{ km}}{2\times10^8\text{ m/s}}$, where 2×10^8 m/s is the transmission speed of light in fiber[12].

fabrics. Candidate routes that look equivalent at the topology level often show asymmetric delay–capacity trade-offs. Oblivious hashing (e.g., ECMP) or uniform-cost choices ignore these asymmetries and can systematically place flows on suboptimal paths.

These differences imply two requirements for inter-DC RDMA routing. First, routing must explicitly account for both propagation delay and provisioned capacity when ranking paths. Second, routing must use timely signals that indicate imminent congestion (so decisions remain useful despite long RTTs). We use these requirements to motivate the design of our cost-fusion, on-switch scoring, and diversity-preserving selection mechanisms (§3).

2.2 Existing Routing Approaches and Their Gaps

Existing DC routing and traffic-engineering techniques are mature but have gaps when applied to long-haul RDMA traffic. Equal-Cost Multipath (ECMP[6, 7]) is simple and widely deployed but hashes obliviously and ignores capacity/delay asymmetry. Weighted schemes (e.g., WCMP[13]) incorporate static weights to address asymmetry, yet they are based on slow topology information and lack timely congestion awareness. Utility/capacity-aware approaches (e.g., UCMP[8]) blend bandwidth and latency considerations but were designed for specific architectures (e.g., reconfigurable DCNs) and often rely on assumptions—like circuit wait costs, that do not hold in conventional WANs. Centralized SDN traffic engineering (e.g., B4-style controllers[9, 10, 14–17]) can optimize global utilization but incurs control-plane latency that makes it hard to react to fast congestion in high-RTT environments. Flowlet or packet-spraying techniques[18] improve utilization but risk RDMA reordering or require host/ASIC changes.

In short, most prior schemes either (a) ignore static path heterogeneity, (b) depend on slow feedback, or (c) require host or heavy switch changes. These gaps map directly to our design challenges Ⓒ1- Ⓒ3 and motivate a routing approach that fuses slow control-plane path quality with timely, hardware-friendly on-switch congestion cues while preserving RDMA constraints (see §3).

2.3 Key Challenges and Solutions

The background above can be summarized into three challenges that any practical inter-DC RDMA routing design must address.

Ⓒ1: ***How can we define a concise "path quality" representation that captures both propagation delay and link capacity? (Solved in §3.2)***

Inter-DC topologies exhibit substantial heterogeneity: different candidate paths vary widely in propagation delay and in provisioned capacity. A routing metric must compress these partly static, partly slow-varying attributes into a form that switches can compare at line rate.

The path representation should (i) jointly reflect propagation delay and nominal capacity, (ii) be stable enough to be computed or normalized by the control plane and installed on the switch as compact per-path scores, and (iii) avoid expensive per-packet arithmetic on the data plane (i.e., the data plane should only do lookups and integer comparisons).

If path heterogeneity is ignored, capacity-aware methods may choose high-bandwidth but high-latency routes (hurting FCT), while latency-only choices underutilize available capacity. A concise, precomputed Path Quality score enables fast on-switch comparisons and informed trade-offs between delay and throughput.

Ⓒ2: ***How can a DCI switch rapidly and robustly detect imminent congestion on inter-DC paths so that routing decisions remain effective despite long RTTs? (Solved in §3.3)***

In inter-DC links, conventional congestion feedback (ECN) is delayed by large propagation times; moreover, instantaneous queue snapshots confuse transient bursts with sustained growth. As a consequence, control or host-based signals are often too outdated for timely route decisions, while naive use of instantaneous samples causes noisy, oscillatory behavior.

A practical routing-oriented congestion signal must (i) be responsive to imminent queue buildup, (ii) suppress high-frequency noise to avoid undue re-routing, (iii) be representable as a compact quantized value (e.g., an 8-bit score), and (iv) be computable in the data plane using only hardware-friendly primitives (integer add/sub, bit shifts, comparisons, and small lookup tables).

Without such timely and implementable congestion sensing, switches either make decisions too late — causing transient tail-latency spikes — or over-react to bursts and cause frequent path churn; both outcomes degrade flow completion times and overall system predictability.

Ⓒ3: ***How can we efficiently avoid selection conflicts when many flows make routing choices at the same time? (Solved in §3.4)***

Inter-DC traffic often involves bursts of new flows that start near-simultaneously. If each new flow independently selects the currently cheapest path, many flows may concentrate on the same next-hop (a selection cascade, we call it herd effect), quickly saturating its egress queue and producing severe short-term tail latency.

A deployable mitigation must (i) rely on atomic, low-cost operations (register add/sub, comparisons), and (ii) preserve path diversity (e.g., by filtering high-cost candidates then randomizing among the low-cost set).

Without an efficient and bounded-state mechanism to prevent selection cascades, locally optimal per-flow choices will collectively create global congestion spikes and tail-latency degradation. Practical herd mitigation is therefore essential for robust routing in high-concurrency inter-DC environments.

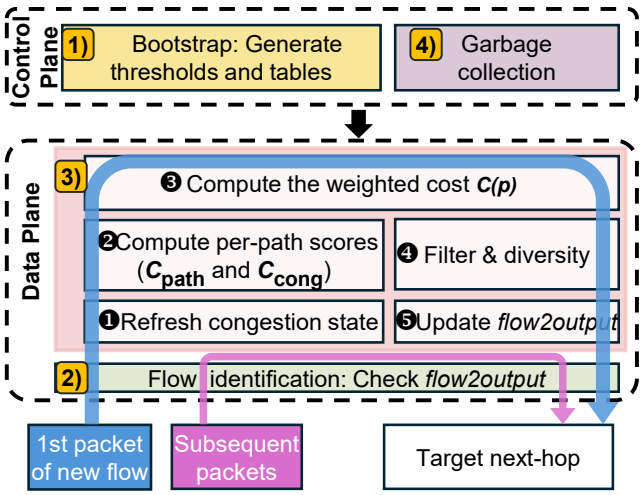


Figure 2. LCMP architecture overview.

Solutions. LCMP addresses the three challenges above with the following solutions for a practical inter-DC routing system.

1. Providing a compact, deployable path-quality representation. (See §3.2)
2. Designing a timely, data-plane friendly congestion estimator. (See §3.3)
3. Enabling herd mitigation with diversity-preserving selection under simultaneous flow arrivals. (See §3.4)

3 LCMP Design

3.1 Design Overview

3.1.1 High-Level Abstraction. LCMP makes per-flow next-hop decisions by fusing a control-plane view of path quality with on-switch congestion signals. Concretely, for a candidate path p we compute an integer cost,

$$C(p) = \alpha \cdot C_{\text{path}}(p) + \beta \cdot C_{\text{cong}}(p), \quad (1)$$

where C_{path} is a precomputed control-plane score that encodes propagation delay and provisioned capacity, and C_{cong} is an on-switch congestion score derived from instantaneous queue level, short-term trend, and a persistence penalty. The switch picks the final egress from a low-cost candidate set. The abstraction directly targets the three challenges identified in §2.3:

Addressing (C1) heterogeneous, asymmetric topologies. We separate slowly-varying path attributes from transient congestion by precomputing a compact per-path quality score in the control plane (§3.2). Encoding delay and provisioned capacity into a score lets the data plane rapidly compare heterogeneous paths without global queries.

Addressing (C2) slow and easily outdated congestion signals. Rather than relying on end-to-end or controller-roundtrip feedback, each DCI switch maintains on-switch signals: a quantized instantaneous queue level, short-term trend accumulator, and a duration counter (§3.3). These signals focus the decision on imminent, local queue growth and are normalized to be robust to sampling noise and long RTTs.

Addressing (C3) many simultaneous flows and herd effects. To avoid simultaneous choices collapsing onto the same low-cost path, LCMP applies a two-stage selection: (i) filter out the high-cost suffix of candidate paths, and (ii) perform hash-based selection within the reduced, low-cost set (§3.4).

3.1.2 Runtime Workflow. Fig. 2 provides a high-level overview of LCMP. Below we describe the packet-time decision workflow.

1) DCI Switch Bootstrap. At switch initialization time LCMP installs a small set of tables and threshold vectors that the data plane uses for fast mapping and normalization:

Link capacity thresholds. A small vector of increasing link capacity thresholds (e.g., $N = 10$ classes) is created: each class boundary is proportional to a configured link capacity. These thresholds map link rates into a discrete link score lookup.

Queue thresholds. The switch divides its per-port egress buffer capacity into levels and records per-level thresholds. These thresholds are used to map instantaneous queue bytes to a quantized queue level Q.

Level score table. A linear mapping from level index to a 0–255 score is precomputed. This avoids per-packet floating computation.

Trend normalization tables. For each coarse link-rate bucket (e.g., 25/100/400 Gbps), a small per-level trend threshold vector is created. These tables normalize the raw trend accumulator into a trend level T. If a rate bucket is not present at initialization the data plane can create a small normalized table on-demand from the link rate.

These compact data structures (a few small vectors and lookup tables per switch) are sized to fit on programmable switch memory and to be installed/updated by the control plane as link or provisioning information changes (see Fig. 3).

2) Flow Identification. On packet arrival the switch forms a flow identifier (e.g., a five-tuple hash). If the packet belongs to an *established* flow (a $flow2output$ mapping exists), the switch refreshes the flow's last-seen timestamp and forwards the packet via the previously chosen egress. This guarantees path consistency and prevents out-of-order packets[19].

3) Flow Routing. If the packet is the first packet of a flow, the switch executes the full LCMP decision path:

Refresh congestion state (❶). It invokes a light-weight monitor to sample per-port queue depth and update the short-term trend estimator. This step updates three signals for each candidate port: (1) Q: queue occupancy mapped to a level via preinstalled thresholds; (2) T: short-term trend obtained via a shift-based EWMA, $T = T_{\text{old}} - (T_{\text{old}} \gg K) + (\Delta \gg K)$, where

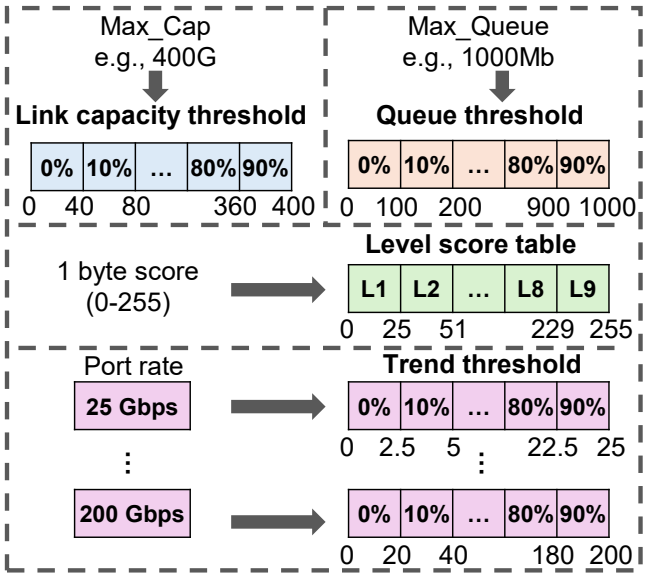


Figure 3. Switch bootstrap tables and mappings. Control plane installs a small set of vectors.

Δ is the queue-byte delta between samples, K is an integer (e.g., 3), and $\gg$ denotes a right-bit-shift normalization; (3) D: a duration (persistence) penalty that accumulates when Q stays above a high-water mark and decays otherwise.

Compute per-path scores (❷). For each candidate path computes:

- *delayScore* via a shift-based mapping;
- *linkCapScore* via a control-plane installed capacity-class lookup (data plane compares configured link capacity against threshold table and returns a score);
- C_{path} by combining *delayScore* and *linkCapScore* with integer weights and a right-shift normalization;
- C_{cong} by combining quantized Q, T, D with integer weights and a right-shift normalization.

Compute the weighted cost $C(p)$ (❸). Compute the weighted cost of each path with Eq. (1).

Filter and diversity-preserving selection (❹). Sort candidate paths by the fused cost $C(p)$; remove the high-cost suffix (paths above a cut), keep a reduced candidate set (we use the top 50% by cost in our implementation), and perform a hash-based ECMP selection within that reduced set to pick the final egress.

Update flow2output mapping (❺). The selected mapping is then recorded in a flow table so that subsequent packets of the flow follow the same egress.

4) Garbage Collection. Per-flow consistency is necessary to avoid reordering and ensure stable path utilization. LCMP therefore maintains a bounded flow cache that maps a flow identifier to the chosen egress and a last-seen timestamp.

Algorithm 1: CalcDelayCost: saturating, shift-based mapping from delay to delayScore.

Input: one_way_delay
Output: delayScore in $[0, 255]$

```
MAX_DELAY = 32      // configured saturation point (ms)
SHIFT = 5           // right-shift equivalent to dividing by MAX_DELAY_MS
if one_way_delay >= MAX_DELAY then
    return 255                  // at worst score
end
delayScore ← (one_way_delay * 255) ≫ SHIFT)
return delayScore
```

Flow cache entry and operations. Each entry contains (1)Flow ID, (2) `outDevIdx`: chosen egress port/index, and (3) `lastSeen`: last packet arrival time. On packet arrival an established flow entry is refreshed and the packet forwarded via the recorded egress. Only the first packet of a flow executes the full cost computation and selection.

Garbage collection and boundedness. A periodic garbage collection evicts entries whose `lastSeen` exceeds a configured idle timeout (e.g., a fraction of RTT-based shortTimeout or a conservative fixed value). This keeps the flow cache bounded and prevents outdated mappings from persisting indefinitely.

Importantly, the storage overhead of LCMP is *small*, a 50k-entry simultaneous flow cache requires only 1.2 MB (see §4 for details).

3.2 Compact Control-Plane Path-Quality Representation

Inter-DC topologies exhibit largely static but heterogeneous attributes (propagation delay and provisioned capacity) that should be respected by any path-selection policy. LCMP separates these slowly-varying, control-plane-friendly attributes from fast on-switch signals by precomputing a compact per-path *path-quality* score $C_{\text{path}} \in [0, 255]$ and installing it as a small table on each DCI switch.

The control plane obtains per-link one-way propagation delay and configured link capacity, maps each metric to a score, and fuses them with integer weights:

$$pathScore = w_{dl} \cdot \text{delayScore}(p) + w_{lc} \cdot \text{linkCapScore}(p),$$

$$C_{\text{path}}(p) = \min\Big(pathScore \gg S_{\text{path}},\ 255\Big).$$

The mapping functions are deliberately simple and integer-only. As shown in Alg. 1 and Alg. 2, delayScore linearly maps one-way delay to 0–255 (saturating at a configured maximum, e.g., 32, 64 ms), and linkCapScore maps link rate into a small number of classes via preinstalled thresholds.

Algorithm 2: CalcLinkCapCost: link capacity-class lookup mapping link capacity to linkCapscore.

Input: linkCap, linkCapThresholds[0..N-1], levelScore[0..N-1]
Output: linkCapScore in $[0, 255]$

```
for i ← N − 1 to 0 By −1 do
    if linkCap ≥ linkCapThresholds[i] then
        return 255 - levelScore[i]        // higher capacity ⇒ smaller cost
    end
end
return 255
```

3.3 Realtime, On-Switch Congestion Estimator

Timely and noise-robust congestion signals are central to LCMP's effectiveness in long-RTT environments. LCMP generates a on-switch congestion score C_{cong} by fusing three signals: instantaneous queue level Q, a short-term trend level T, and a duration (persistence) penalty D.

Instantaneous queue level Q. The monitor samples per-port queue bytes and maps the sampled byte count into a discrete level via the preinstalled `qThresh` vector. The level index is then converted to a score via `levelScore`.

Short-term trend T. LCMP uses a shift-based EWMA-style accumulator:

$$T = T_{\text{old}} - (T_{\text{old}} \gg K) + (\Delta \gg K). \tag{2}$$

The raw trend is mapped to a discrete trend level by comparing it to a normalization vector and converting the matched level to a score. Non-positive trends map to zero to focus reactions on growing queues.

Duration penalty D. A counter increases while Q exceeds a high-water mark and decays when Q is low. This persistence counter is right-shifted to produce a penalty score.

Fusion into C_{cong}. The three signals are combined with integer weights and a right-shift normalization:

$$congScore = w_{ql} \cdot Q + w_{tl} \cdot T + w_{dp} \cdot D, \tag{3}$$

$$C_{\text{cong}}(p) = \min\Big(congScore \gg S_{\text{cong}},\ 255\Big). \tag{4}$$

Sampling and robustness. A lightweight monitor routine iterates over device ports at a modest cadence. Trend normalization uses the observed sampling interval when comparing the trend accumulator to per-rate thresholds, making T robust to modest variations in sampling frequency. This design balances responsiveness to imminent queue growth with suppression of high-frequency noise.

3.4 Diversity-Preserving Selection for Herd Mitigation

To prevent simultaneous new-flow from choosing the same single low-cost port (called "herd effect"), LCMP performs a two-stage selection: cost-based filtering followed by randomized selection within the reduced set.

Two-stage selection. For a new flow the data plane computes the fused cost $C(p)$ for each candidate path p. The switch forms a vector of $(C(p), p)$ pairs, sorts them by cost (small N so sorting is cheap), and removes the high-cost suffix. By default LCMP retains the lower half of candidates. From the remaining paths, the switch performs ECMP inside the low-cost subset.

Fallbacks and corner cases. If all candidate paths are highly congested, LCMP falls back to selecting the minimum-cost path to avoid pointless randomization among uniformly bad choices. The per-flow mapping is then recorded in the local flow cache to preserve path consistency for subsequent packets.

4 Analysis of Resource Cost

Before describing a concrete implementation, we quantify LCMP's resource and decision compute requirements to demonstrate that the design is practical on modern DCI switches. We provide a parameterized accounting of per-port and per-flow storage, a conservative example deployment (48 ports, 50k-entry flow cache), and a breakdown of the *per-new-flow* integer operations and table lookups required for the full cost computation and selection.

Importantly, LCMP performs the relatively expensive cost computation only once per new flow: subsequent packets of the same flow hit the local flow cache, incur a simple lookup, refresh the last-seen timestamp, and are forwarded via the recorded egress. Our accounting therefore focuses on the per-new-flow decision cost, roughly a few dozen table lookups and $O(m \log m)$ comparisons for m candidate next-hops. The numbers below show that LCMP's working set and its per-new-flow compute comfortably fit within typical programmable-switch budgets.

Per-element sizes. We assume the following conservative storage sizes typical in switch registers: (1) 32-bit integer fields (e.g., `queueCur`, `queuePrev`, `trend`, `durCnt`): 4 bytes (B) each. (2) 64-bit timestamps (e.g., `lastSample`, `lastSeen`): 8 B each. (3) Per-path or per-level 8-bit scores: 1 B each (stored in table entries).

Per-port and per-flow memory overhead.

$$\text{Per-port bytes} = \underbrace{4}_{\text{queueCur}} + \underbrace{4}_{\text{queuePrev}} + \underbrace{4}_{\text{trend}} + \underbrace{4}_{\text{durCnt}} + \underbrace{8}_{\text{lastSample}} = 24 \text{ B/port},$$

$$\text{Per-flow bytes} = \underbrace{8}_{\text{flowId}} + \underbrace{4}_{\text{portIdx}} + \underbrace{8}_{\text{lastSeen}} = 20 \text{ B/flow}.$$

Demonstration. Consider a DCI switch with 48 ports and a bounded flow cache sized for 50,000 entries. Using the formulas above:

- **All port cache:** 24 B/port × 48 ports = 1152 B.
- **All flow cache:** 24 B/flow × 50,000 flows = 1.2 MB.
- **Control tables:** bandwidth thresholds and levelScore for $N = 10$ classes: approximately a few dozen bytes each. Per-path C_{path} table size depends on the number of installed paths P; e.g., $P = 10$K paths ≈ 10 KB for scores.

These totals (roughly 1.2 MB) are well within typical on-switch memory budgets (and can be kept smaller if resources are constrained with some methods[20]).

Per-new-flow computational cost. Let m be the number of candidate next-hops (typical $m \in [2, 8]$). For each candidate the pipeline performs:

- 2–4 table lookups (bandwidth class, levelScore, trend thresholds),
- a handful of integer ops (compute delayScore, combine weights: 8–12 adds/shifts),
- compare operations to form sort keys.

A conservative per-candidate estimate is 15 integer primitives; thus for $m = 6$ the cost is 90 primitives plus a small sorting cost (for $m = 6$, sorting requires on the order of $m \log_2 m = 6 \times 2.6 \approx 15$ comparisons). Total primitive count 105 integer operations for a new-flow decision, which is trivial for modern ASIC pipelines or programmable switch.

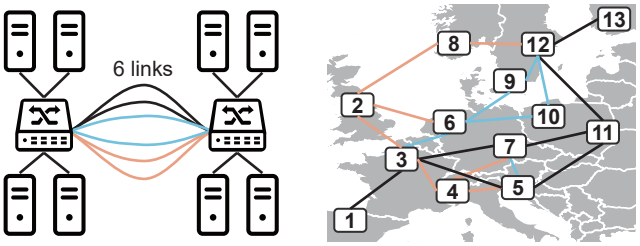
5 Implementation

We implemented LCMP with deployability and low operational cost as primary goals. Only DCI (inter-DC) edge switches require an upgrade; end hosts and the intra-DC fabric remain *unchanged*, enabling low-risk, incremental deployment.

Dataplane requirements. LCMP targets modest, commonly available dataplane primitives: a few compact lookup tables (per-path C_{path}, bandwidth thresholds and level-score vectors), a small set of 32-bit per-port registers (queue, trend, duration) and a bounded per-flow cache, and integer-only operations (adds, right-shifts, comparisons) plus cheap sorts over a small candidate set. These requirements fit modern P4-capable devices or a SmartNIC-assisted dataplane.

Control-plane provisioning. The controller performs only slow-path work: installing per-path C_{path} scores and threshold vectors (minutes–hours cadence), pushing conservative default weights (e.g., $(\alpha, \beta) = (3, 1)$) for operator tuning, and collecting lightweight telemetry (per-port queue levels, flow-cache occupancy) for verification. Because updates are infrequent, LCMP integrates cleanly with existing SDN workflows.

Incremental rollout and safe fallbacks. LCMP supports partial upgrades: upgraded DCIs apply LCMP locally while legacy devices continue normal forwarding. Decisions are local next-hop choices and do not require new packet headers or remote upgrades. If LCMP tables are missing or outdated, or all candidates are uniformly poor, switches fall back to ECMP.



(a) Testbed: 8-DC topology. **(b)** BSONetwork: real-world Europe-spanning topology.

Figure 4. Topologies used in evaluation.

Compatibility with transport. LCMP is orthogonal to end-host CC: it requires *no* RNIC or host-stack changes and interoperates with DCQCN, TIMELY, DCTCP, etc.

Summary. In short, LCMP trades only modest, localized dataplane resources on DCI switches for meaningful placement and tail-latency improvements, and is designed for incremental, low-risk deployment with simple operational telemetry and safe fallbacks.

6 Evaluation

Our evaluation across a small-scale testbed and large-scale NS-3 simulations reveals the following key findings:

1. On the 8-DC testbed (Fig. 4a) LCMP reduces median and tail FCT slowdown by up to 76% and 64%, respectively, compared to the SOTA method UCMP (§6.1).
2. For endpoint pairs with many candidate routes under the *2000 km inter-DC scenario*, LCMP delivers clear benefits: median FCT improves by 7%–11% and P99 by 15%–18% versus ECMP (even larger improvements versus UCMP) (§6.2).
3. Improvements persist across realistic workloads and across several RDMA-capable CCs: LCMP reduces median FCT slowdown by 32%–35% and 74%–75%, and P99 slowdown by 39%–45% and 40% compared to ECMP and UCMP, respectively (§6.3).

6.1 Small-Scale Testbed Experiments

Testbed topology. As shown in Fig. 1a, we use a 8-DC topology and each DC is a small leaf–spine fabric (1 DCI switch, 2 spine switches, 4 leaf switches, and 16 servers). Servers attach to leaf switches via a single NIC. All intra-DC links run at 100 Gbps and use a 1 µs propagation delay. To avoid artificial bottlenecks inside a DC, links between DCI

switches and spine switches are set to 400 Gbps. The inter-DC link capacities are set to 40 Gbps, 100 Gbps, and 200 Gbps and propagation delays are set from 5 ms to 250 ms.

Workloads. Here we use a realistic DCN workload Web Search[21]. We synthesize an all-to-all inter-DC traffic pattern by randomly pairing senders and receivers between DC1 and DC8.

Baselines. We compare LCMP to two practical baselines that represent widely deployed (ECMP[6, 7]) and SOTA (UCMP[8]) routing strategies for DCNs. ECMP is the common default routing scheme in DCNs, which hashes flows across paths deemed to have equal cost. ECMP ignores link capacity heterogeneity and propagation delay differences, leading to suboptimal flow placement in heterogeneous inter-DCs where path quality varies significantly beyond hop count. UCMP is a recent scheme proposed for reconfigurable datacenter networks that combines circuit-waiting latency and link capacity considerations into a unified cost to guide path selection. Conventional inter-DC deployments do not incur circuit-waiting delays, so for a fair comparison we reimplement UCMP's capacity-aware component only.

Metrics. Our primary metric is **FCT slowdown**[22]. It means a flow's actual FCT normalized by its ideal FCT. Ideal FCT is the FCT of the same flow when run alone in the network with the shortest propagation delay in its topology, which isolates queueing effects due to multiplexing. We repeat the experiment three times and the presented figures report average(median) and tail(P99) behavior.

Setup. We build a small-scale testbed consisting of 9 servers (see Fig. 4a), which is simplified form Fig. 1a. Each server is equipped with 12-core CPU, 32 GB RAM. 4 machines are grouped behind a DCI switch and act as DC1; another 4 machines serve as DC8. The remaining host runs Mininet[23] and emulates the intermediate DCs (DC2-DC7) and the inter-DC links. Per-path delay and link capacity is realized on the Mininet via virtual links to match the delays and capacities used in Fig. 1a. DCQCN[4] is used as the default CC. We run the workload at 30%, 50% and 80% load(i.e., light, medium and heavy load) on the 8-DC topology.

Results. As shown in Fig. 5, across three loads LCMP reduces median FCT slowdown by 36%–41%, 76% and P99 slowdown by 56%–68%, 45%–64% compared to ECMP and UCMP, respectively. These improvements arise because LCMP avoids ECMP's random placement on high-delay links and UCMP's capacity-only bias by fusing control-plane path quality with on-switch congestion signals.

Simulator fidelity. Fig. 6 compares FCT slowdown measured on our testbed and in the NS-3 simulator under 30% load with the same setting. The line shows the near-linear correlation between them (the Pearson correlation values are

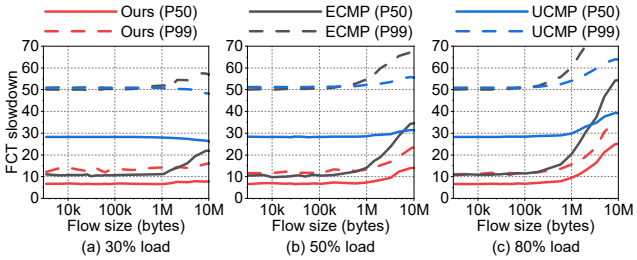


Figure 5. Median and tail FCT slowdown for *Web Search* on the testbed topology under 30%, 50%, 80% load.

95% for P50 and 97% for P99), which validates NS-3 as a faithful platform for the larger-scale experiment. Consequently, all remaining experiments use NS-3 results.

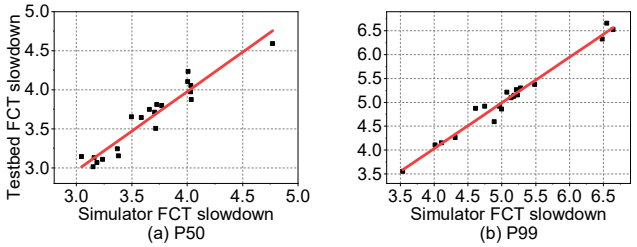


Figure 6. [Simulator fidelity] NS-3 vs testbed FCT slowdown.

6.2 Large-Scale NS-3 Simulations

Real-world topology. Fig. 4b provides a realistic European network topology (*BSONetworkSolutions*) drawn from the Internet Topology Zoo[24]. This topology contains backbone, customer and transit links across regions and therefore captures realistic heterogeneity in both delay and capacity. There are *13 DCs* and we set inter-DC propagation delays to 1 ms (for 200 km), 5 ms (for 1000 km) and 10 ms (for 2000 km), and increase switch buffer sizes to 6 GB for the long distances [12] to reflect long-haul provisioning and to satisfy PFC headroom requirements for RDMA traffic.

Workloads. In addition to Web Search[21], we use two more realistic DCN workloads in our experiments, which is Facebook Hadoop[25], and Alibaba Storage[22]. For each workload we synthesize an all-to-all inter-DC traffic pattern by randomly pairing senders and receivers *across all DCs*. We also vary the offered load to achieve average link utilizations of 30%, 50% and 80%.

Baselines. The same methods used in testbed: ECMP and UCMP.

6.2.1 System-Wide Validation: Aggregate FCT for All-to-All Inter-DC Flows.

Setup. We use NS-3 for simulations under 30%, 50%, and 80% loads of Web Search. All 13 DCs participate in an all-to-all inter-DC traffic matrix.

Results. As shown in Fig. 7, LCMP does not harm overall median performance and yields modest tail improvements. Compared to ECMP the median FCT slowdown is essentially unchanged across the three loads, while the P99 FCT falls by roughly 2%–9%. Against UCMP, UCMP sometimes produces slightly lower medians because it prefers high-capacity paths, but tail reductions are comparable. These modest aggregate improvements arise because most inter-DC flows in the all-to-all experiments traverse only a small number of candidate next-hops, so averaging across all flows hides the much larger gains we observe for flows that have many candidate paths. Nonetheless, LCMP reduces tail latency without increasing median FCT.

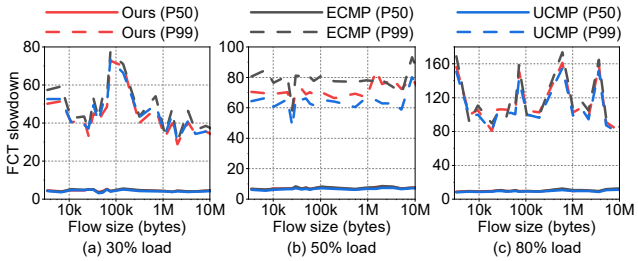


Figure 7. [System-wide validation] Median and tail FCT slowdown across all inter-DC flows at 30%, 50% and 80% loads.

6.2.2 Representative DC-Pair Case Study: (DC1, DC13).

Setup. To highlight LCMP's mechanism, We filter the same runs used above to extract flows between DC1 and DC13, which exhibit multiple candidate routes with differing delay/capacity trade-offs.

Results. When we focus on a representative DC-pair with multiple candidate routes (DC1–DC13), LCMP 's benefits become clear in Fig. 8. For flows between DC1 and DC13, LCMP reduces median slowdown by 7%–11% and P99 slowdown by 15%–18% relative to ECMP across the three loads. Versus UCMP the improvements are larger for medians (median slowdown drops by 25%–30%) while tails fall by 13%–16%. These focused improvements arise because DC1–DC13 runs have multiple viable next-hops with differing delay and capacity trade-offs: LCMP 's fusion of path-quality with on-switch congestion signals both (i) avoids systematically placing latency-sensitive flows on high-delay or high-capacity paths and (ii) mitigates transient herding on a single low-cost port, producing substantially better median and tail FCTs in multi-path inter-DC scenarios.

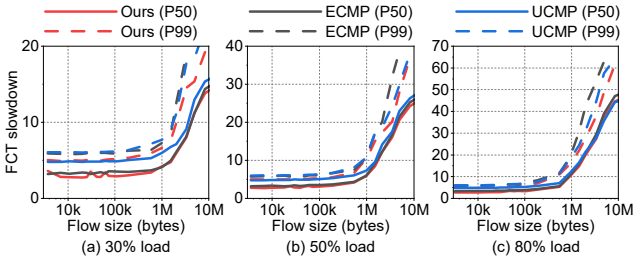


Figure 8. [DC-pair case study] Median and tail FCT slowdown for flows between DC pair (DC1, DC13) at 30%, 50% and 80% loads.

6.3 Deep Dive

Having established system-wide behavior in the previous section, here we omit repeat aggregate results and focus on the representative DC pair (DC1, DC8) in the Fig. 1a. We will further demonstrate LCMP's robustness across realistic workloads and common CC algorithms.

6.3.1 Workload Sensitivity.

Setup. We run three DC workloads (Web Search, Facebook Hadoop, Alibaba Storage) at 30% load using DCQCN as the default CC.

Results. Fig. 9 shows that, for *Web Search* LCMP reduces median slowdown by 36% and P99 slowdown by 58% versus ECMP, and by 76% (median) and 82% (tail) versus UCMP. For *Alibaba Storage* LCMP cuts median/tail by 32%/68% versus ECMP and by 80%/68% versus UCMP. For *Facebook Hadoop* LCMP reduces median/tail by 26%/69% versus ECMP and by 78%/69% versus UCMP. These results show that median improvements primarily stem from LCMP respecting path-quality (avoiding high-delay, high-capacity routes), while the large tail reductions come from the on-switch congestion estimator and diversity-preserving selection.

Takeaway. LCMP 's benefits are robust to realistic variations in flow-size distributions: improvements in both p50 and P99 persist across workloads.

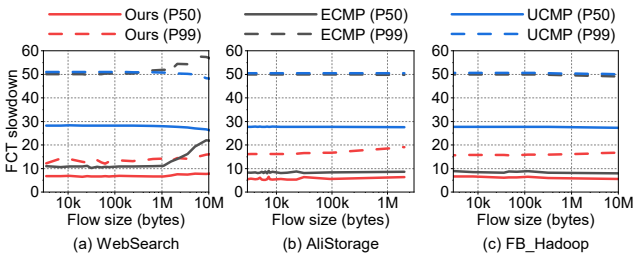


Figure 9. Workload sensitivity: median and tail FCT slowdown different three workloads.

6.3.2 Congestion-Control Orthogonality.

Setup. we evaluate LCMP 's interaction with multiple end-host CCs: DCQCN (shown in Fig. 5), HPCC, TIMELY and DCTCP. All experiments use the Web Search workload at 30% load.

Results. Across all tested CC algorithms, Fig. 10 shows that, LCMP delivers highly consistent benefits: LCMP reduces median FCT slowdown by 32%–35% and 74%–75%, and P99 slowdown by 39%–45% and 40% compared to ECMP and UCMP, respectively. The numbers are stable across the four CCs we tested (DCQCN earlier, plus HPCC, TIMELY and DCTCP here), indicating that LCMP's improvements are largely orthogonal to the choice of end-host CC.

This pattern has two implications. First, it shows LCMP is plug-and-play: operators can deploy LCMP alongside existing CCs and expect similar improvements without changing host stacks. Second, the similarity across CCs suggests a broader lesson: many CC algorithms developed for intra-DCs rely on timely feedback and small RTTs, assumptions that weaken in an inter-DCs (large-RTT) . Consequently, future CC research for Inter-DCs should (i) revisit feedback mechanisms to provide faster, more informative signals over long RTTs, and (ii) explore cross-layer designs that let routing and CC share concise path-quality and imminent-congestion costs. These directions would complement routing-centric solutions like LCMP and further improve both FCT performance in multi-DCs.

Takeaway. These results confirm LCMP 's orthogonality: operators can adopt LCMP without changing RNICs or transport protocols and still obtain consistent median/tail reductions. This makes LCMP a low-risk, deployable addition to current inter-DC stacks.

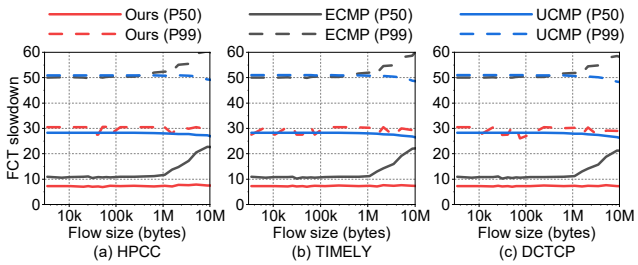


Figure 10. Congestion-control orthogonality: median and tail FCT slowdown under different CCs.

7 Sensitivity Analysis and Discussion

We present ablation and parameter-sensitivity results in this section. These experiments show how to configure LCMP and why each component matters in practice. The experiments measure the impact of the control-plane path-quality term and the data-plane congestion term. They also identify robust integer-weight defaults for heterogeneous inter-DC deployments. Unless noted otherwise, all runs use the Web Search workload at 30% load using DCQCN as the default CC.

7.1 Ablation Sensitivity Analysis

We run three variants on the 8-DC topology (1a):

- `rm-alpha` — path-quality removed ($\alpha = 0$);
- `rm-beta` — congestion removed ($\beta = 0$);
- full LCMP with representative (α, β) settings.

Key findings. Fig. 11a shows two clear failure modes. First, the `rm-alpha` run (path-quality removed) severely degrades performance across almost all flow sizes. For example, the median for a 3,438 B flow rises from 6.8 (normal) to 26.0 when $\alpha = 0$ (+280%). The P99 for the same size rises from 12.1 to 50.0 (+312%). The `rm-alpha` curve stays well above the others for the entire flow-size range. This pattern means that using only on-switch congestion signals tends to place flows on high-delay routes in this heterogeneous topology. Second, the `rm-beta` run (congestion removed) preserves medians for small and mid-sized flows but fails for large transfers. For the largest flows (29.7 MB) the median increases from 8.7 (normal) to 31.2 (+260%) and P99 jumps from 17.1 to 58.4 (+240%). This shows that path-only selection cannot prevent contention among long-lived elephants. The full LCMP run consistently achieves the lowest and most stable p50 and P99 across sizes.

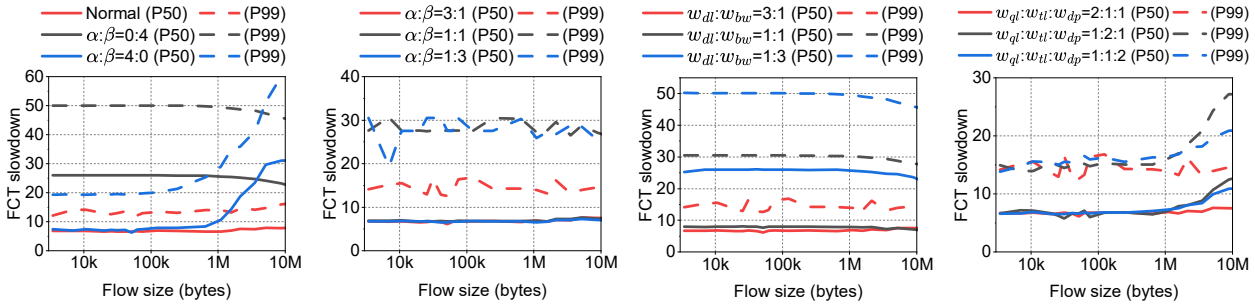
Takeaway. Both components are necessary. The control-plane path-quality term prevents systematic placement on high-delay links and thus keeps medians low. The on-switch congestion term prevents herd-driven contention among large flows and thus controls tails. In practice, operators should use a fused cost with non-zero α and β. A modest bias toward path quality (e.g., $\alpha = 3, \beta = 1$) yields a robust trade-off between median and tail in capacity–delay asymmetric inter-DC deployments.

7.2 Global Fusion-Weight Sensitivity Analysis

We sweep global fusion weights $(\alpha, \beta) \in \{(3, 1), (1, 1), (1, 3)\}$ on the 8-DC topology.

Key findings. As shown in Fig. 11b, all three weight settings produce similar medians. The delay-biased setting $(3, 1)$ matches others on p50. The delay-biased setting, however, yields much smaller tails. Typical P99 values under $(3, 1)$ fall in the 12–16 range. The balanced $(1, 1)$ and congestion-biased $(1, 3)$ settings show P99 values around 24–30 for many sizes. In short, prioritizing the control-plane path-quality term reduces P99 by roughly half compared to balanced or congestion-heavy choices, while leaving medians essentially unchanged.

Takeaway. When bandwidth and delay misaligned, favor path-quality in the fusion. A delay-biased fusion (e.g., $\alpha =$



(a) [Ablation analysis]

(b) [Global weight analysis] Weight tuples $(\alpha, \beta) = (3,1), (1,1), (1,3)$

(c) [Path-quality weights analysis] Weights tuples $(w_{dl}, w_{lc}) = (3,1), (1,1), (1,3)$

(d) [Congestion-cost weights analysis] Weight tuples $(w_{ql}, w_{tl}, w_{dp}) = (2,1,1), (1,2,1), (1,1,2)$.

Figure 11. [Sensitivity analysis] Median and tail FCT slowdown for WebSearch on the 8-DC topology at 30% load.

3, $\beta = 1$) gives the most stable tails without hurting medians. Balanced or congestion-heavy weightings make the system more likely to over-react to transient signals and to send latency-sensitive flows onto high-capacity but slow links.

7.3 Path-Quality Weight Sensitivity Analysis

We vary $(w_{dl}, w_{lc}) \in \{(3,1), (1,1), (1,3)\}$ inside C_{path}.

Key findings. As shown in Fig. 11c, The delay-biased path score $(3,1)$ gives the best medians and tails. Under $(3,1)$ p50 values cluster near 6.1–7.6 and P99 near 12–17. The balanced $(1,1)$ choice yields slightly worse medians (7.0–8.1) and much larger tails (27–31). The capacity-biased $(1,3)$ choice performs worst: it raises medians and tails dramatically (p50 often > 20 and P99 in the 43–50 range for many sizes). Overall, weighting delay more than bandwidth halves P99 versus balanced settings and reduces medians by roughly 10–20% compared to the balanced choice.

Takeaway. When capacity and latency trade off, give higher weight to delay in C_{path}. A delay-biased setting (e.g., w_{dl}:w_{lc} = 3:1) avoids placing latency-sensitive flows on high-capacity but slow links. This choice improves both median and tail FCT.

7.4 Congestion-Cost Weight Sensitivity Analysis

We compare allocations $(w_{ql}, w_{tl}, w_{dp}) \in \{(2,1,1), (1,2,1), (1,1,2)\}$ for C_{cong}.

Key findings. In Fig. 11d, the three allocations show similar medians for small and mid flows. They diverge for large flows and in the tail. The queue-focused setting $(2,1,1)$ gives the most stable behavior: p50 stays near 6.1–7.6 and P99 near 12–17. The trend-heavy $(1,2,1)$ and duration-heavy $(1,1,2)$ settings raise P99 for the largest flows. These settings also increase p50 for the largest sizes (from ≈6–7 up to ≈ 8–14). The queue-focused choice keeps both medians and tails lower.

Takeaway. These results indicate that putting most weight on instantaneous queue level is the safest and most robust choice. A queue-first allocation (e.g., 2:1:1) limits P99 inflation while keeping medians stable. Emphasizing short-term trend or persistent-duration penalties can help very short flows but risks concentrating elephants onto fewer paths and amplifying noise. Therefore we recommend a conservative, queue-focused default (e.g., 2:1:1) for production deployments where path diversity and capacity–delay trade-offs exist.

7.5 Limitations

While LCMP reduces placement inefficiencies caused by topology heterogeneity, it has two practical limitations that point to future work.

Flow-level stickiness limits responsiveness. LCMP pins a flow to a chosen egress to preserve RDMA in-order delivery. This design keeps correctness on today's RNICs. It also reduces the switch's ability to rebalance traffic quickly when short-lived bottlenecks appear. In other words, per-flow stickiness improves stability but sacrifices agility under sudden congestion.

Root cause: RNIC out-of-order handling. The stickiness stems from RNICs' sensitivity to out-of-order (OoO) packets and their loss-recovery semantics. Many commodity RNICs treat OoO arrivals as losses and trigger retransmission. Aggressive per-packet or per-flowlet steering can therefore increase retransmits and hurt latency. Recent work shows promising directions to relax this constraint (e.g., in-network reordering and lightweight OoO tracking)[11, 12, 19, 26–28], but such techniques are not yet widely deployed.

Future directions. We highlight two practical research directions. First, explore fine-grained steering with OoO tolerance. We will combine selective per-flowlet or per-packet routing with lightweight in-network reordering or RNIC-side OoO tracking. The goal is to trade a small, controlled

amount of reordering for much faster congestion reaction. Second, pursue cross-layer co-design with congestion control and loss recovery. We will align routing decisions with transport-layer signals so steering does not conflict with senders' recovery logic.

8 Related Work

Long-haul Link Transport Optimization. The expansion of large-scale DCs is constrained by limited land, power, and network connectivity resources. To overcome these limitations, major cloud service providers (CSPs) deploy multiple DCs interconnected through dedicated optical fibers to cover a given region. Recent efforts have focused on optimizing transport over long-haul networks. SWING[29] proposes a PFC relay mechanism that extends high-performance, lossless RDMA to long-haul links. It minimizes the remote RDMA buffer requirements while maintaining high throughput across long-distance connections, without requiring any modifications to existing intra-DC networks. Bifrost[30] introduces a downstream-driven lossless flow control mechanism to support cross-DC data transfers over long distances, achieving low buffer reservation, sustained throughput, and zero packet loss. Considering the characteristics of long-haul links with large RTT and BDP, LSCC[31] proposes a link-segmentation-based congestion control algorithm for inter-DC networks, which leverages more fine-grained control signals to achieve high throughput and low latency over long-haul links.

Inter-DC transport and routing optimization. Inter-DC fabrics, with millisecond-scale RTTs and heterogeneous link capacities, have been addressed largely by two strands of work: control-plane traffic engineering and CC, but not by routing algorithm that jointly considers path quality and on-switch signals. Centralized TE[9, 10, 14–17] yields high steady-state utilization via global optimization yet acts at coarse timescales and cannot make per-flow packet-time choices to avoid short-lived tail spikes. Transport and hybrid proposals that fuse ECN, delay or in-band telemetry[32, 33] improve end-to-end rate control but generally leave path selection to ECMP/UCMP and thus do not resolve placement problems arising from delay–capacity asymmetry. Recent systems[12, 34, 35] reduce feedback latency or strengthen transport semantics, yet they either require costly deployment changes or retain default multipath routing that ignores joint delay–capacity tradeoffs. In short, prior inter-DC work improves global planning or transport behavior but does not provide a lightweight, distributed, data-plane feasible routing layer that compresses path heterogeneity into compact scores and fuses those scores with timely on-switch congestion signals; LCMP is designed to fill precisely this gap by combining control-plane path precomputation (C_{path}) with an integer-friendly on-switch congestion estimator (C_{cong}) to enable RDMA-friendly, cost-aware multipath routing at packet time.

Intra-DC routing, load balancing and CC. Research on intra-DC routing and CC addresses lossless delivery, reordering sensitivity, and sub-µs per-hop latency, but existing schemes typically assume µs-scale feedback, abundant flowlets, or centralized coordination—assumptions incompatible with the long RTTs, path heterogeneity, and high-concurrency herd effects we identify in Ⓒ1– Ⓒ3. Early multipath adaptations[13, 36–45] improve fairness or throughput via static weights or flow-splitting, yet they either lack realtime congestion awareness or risk RDMA-unfriendly reordering. RDMA congestion controllers and telemetry-driven designs[4, 22, 46–58] provide valuable signals for rate control but typically leave path choice to ECMP/UCMP and their feedback is outdated across inter-DC RTTs. Flowlet and sequencing approaches[45, 59, 60] reduce reordering or enable finer steering but depend on host changes, abundant inter-packet gaps, or central schedulers, constraints that limit their usefulness across inter-DCs. Recent programmable-hardware efforts[8, 44] advance switch-side steering but typically do not fuse lightweight, timely on-switch congestion signals with precomputed path-quality scores. LCMP differs by preserving per-flow path consistency, fusing control-plane path quality with integer-friendly on-switch congestion estimates, and using a diversity-preserving, low-state selection that is implementable on commodity programmable switches.

9 Conclusion

We presented LCMP, a distributed long-haul cost-aware multi-path routing framework for inter-DC networks. LCMP fuses a path-quality score with on-switch congestion signals and applies a diversity-preserving selection step to make line-rate multi-path decisions using only integer arithmetic, bit shifts and small lookup tables.

Our evaluation on a small-scale testbed and large-scale NS-3 simulations under the *2000 km inter-DC scenario* demonstrates that this design consistently improves flow-completion behavior and is robust across realistic workloads and CC algorithms. LCMP is low-cost to deploy in practice: it requires upgrades only at DCI switches and uses modest on-switch resources.

We currently enforce per-flow stickiness to preserve RDMA in-order delivery, which limits aggressive rebalancing under sudden congestion. Future work will explore fine-grained steering with lightweight out-of-order tolerance and tighter routing–congestion-control co-design to restore responsiveness without sacrificing correctness.

References

[1] Yixiao Gao, Qiang Li, Lingbo Tang, Yongqing Xi, Pengcheng Zhang, Wenwen Peng et al. 2021. When Cloud Storage Meets RDMA. In *18th*

USENIX Symposium on Networked Systems Design and Implementation (NSDI 21). USENIX Association, 519–533.

[2] Wei Bai, Shanim Sainul Abdeen, Ankit Agrawal, Krishan Kumar Attre, Paramvir Bahl, Ameya Bhagat et al. 2023. Empowering Azure Storage with RDMA. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. USENIX Association, Boston, MA, 49–67.

[3] Adithya Gangidi, Rui Miao, Shengbao Zheng, Sai Jayesh Bondu, Guilherme Goes, Hany Morsy et al. 2024. RDMA over Ethernet for Distributed Training at Meta Scale. In *Proceedings of the ACM SIGCOMM 2024 Conference*. Association for Computing Machinery, New York, NY, USA, 57–70. doi:10.1145/3651890.3672233

[4] Yibo Zhu, Haggai Eran, Daniel Firestone, Chuanxiong Guo, Marina Lipshteyn, Yehonatan Liron et al. 2015. Congestion Control for Large-Scale RDMA Deployments. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, Vol. 45. Association for Computing Machinery, New York, NY, USA, 523–536. doi:10.1145/2829988.2787484

[5] Chuanxiong Guo, Haitao Wu, Zhong Deng, Gaurav Soni, Jianxi Ye, Jitu Padhye et al. 2016. RDMA over Commodity Ethernet at Scale. In *Proceedings of the 2016 ACM SIGCOMM Conference* (Florianopolis, Brazil) *(SIGCOMM '16)*. Association for Computing Machinery, New York, NY, USA, 202–215. doi:10.1145/2934872.2934908

[6] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. 2008. A Scalable, Commodity Data Center Network Architecture. In *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*. Association for Computing Machinery, New York, NY, USA, 63–74. doi:10.1145/1402958.1402967

[7] Christian Hopps. 2000. Analysis of an Equal-Cost Multi-Path Algorithm. RFC 2992. doi:10.17487/RFC2992

[8] Jialong Li, Haotian Gong, Federico De Marchi, Aoyu Gong, Yiming Lei, Wei Bai et al. 2024. Uniform-Cost Multi-Path Routing for Reconfigurable Data Center Networks. In *Proceedings of the ACM SIGCOMM 2024 Conference*. Association for Computing Machinery, New York, NY, USA, 433–448. doi:10.1145/3651890.3672245

[9] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh et al. 2013. B4: Experience with a Globally-Deployed Software Defined Wan. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*. Association for Computing Machinery, Hong Kong, China and New York, NY, USA, 3–14. doi:10.1145/2486001.2486019

[10] Andrew D. Ferguson, Steve Gribble, Chi-Yao Hong, Charles Killian, Waqar Mohsin, Henrik Muehe et al. 2021. Orion: Google's Software-Defined Networking Control Plane. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, 83–98.

[11] Cha Hwan Song, Xin Zhe Khooi, Raj Joshi, Inho Choi, Jialin Li, and Mun Choon Chan. 2023. Network Load Balancing with In-Network Reordering Support for RDMA. In *Proceedings of the ACM SIGCOMM 2023 Conference*. Association for Computing Machinery, New York, NY, USA, 816–831. doi:10.1145/3603269.3604849

[12] Wenxue Li, Xiangzhou Liu, Yunxuan Zhang, Zihao Wang, Wei Gu, Tao Qian et al. 2025. Revisiting RDMA Reliability for Lossy Fabrics. In *Proceedings of the ACM SIGCOMM 2025 Conference*. Association for Computing Machinery, New York, NY, USA, 85–98. doi:10.1145/3718958.3750480

[13] Junlan Zhou, Malveeka Tewari, Min Zhu, Abdul Kabbani, Leon Poutievski, Arjun Singh et al. 2014. WCMP: Weighted Cost Multipathing for Improved Fairness in Data Centers. In *Proceedings of the Ninth European Conference on Computer Systems*. Association for Computing Machinery, New York, NY, USA, 14 pages. doi:10.1145/2592798.2592803

[14] Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon et al. 2015. Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*. Association for Computing Machinery, London, United Kingdom and New York, NY, USA, 183–197. doi:10.1145/2785956.2787508

[15] Kok-Kiong Yap, Murtaza Motiwala, Jeremy Rahe, Steve Padgett, Matthew Holliman, Gary Baldus et al. 2017. Taking the Edge off with Espresso: Scale, Reliability and Programmability for Global Internet Peering. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. Association for Computing Machinery, Los Angeles, CA, USA and New York, NY, USA, 432–445. doi:10.1145/3098822.3098854

[16] Yuchao Zhang, Junchen Jiang, Ke Xu, Xiaohui Nie, Martin J. Reed, Haiyang Wang et al. 2018. BDS: A Centralized near-Optimal Overlay Network for Inter-Datacenter Data Replication. In *Proceedings of the Thirteenth EuroSys Conference*. Association for Computing Machinery, New York, NY, USA, 1–14. doi:10.1145/3190508.3190519

[17] Yuchao Zhang, Xiaohui Nie, Junchen Jiang, Wendong Wang, Ke Xu, Youjian Zhao et al. 2021. BDS+: An Inter-Datacenter Data Replication System With Dynamic Bandwidth Separation. *IEEE/ACM Transactions on Networking* 29, 2 (April 2021), 918–934. doi:10.1109/TNET.2021.3054924

[18] Srikanth Kandula, Dina Katabi, Shantanu Sinha, and Arthur Berger. 2007. Dynamic load balancing without packet reordering. *SIGCOMM Comput. Commun. Rev.* 37, 2 (March 2007), 51–62. doi:10.1145/1232919.1232925

[19] Peihao Huang, Guo Chen, Xin Zhang, Can Liu, Hongyu Wang, Huijun Shen et al. 2025. Fast and Scalable Selective Retransmission for RDMA. In *IEEE INFOCOM 2025 - IEEE Conference on Computer Communications*. 1–10. doi:10.1109/INFOCOM55648.2025.11044566

[20] Shawn Shuoshuo Chen, Keqiang He, Rui Wang, Srinivasan Seshan, and Peter Steenkiste. 2024. Precise Data Center Traffic Engineering with Constrained Hardware Resources. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. USENIX Association, Santa Clara, CA, 669–690.

[21] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C. Snoeren. 2015. Inside the Social Network's (Datacenter) Network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*. Association for Computing Machinery, New York, NY, USA, 123–137. doi:10.1145/2785956.2787472

[22] Yuliang Li, Rui Miao, Hongqiang Harry Liu, Yan Zhuang, Fei Feng, Lingbo Tang et al. 2019. HPCC: High Precision Congestion Control. In *Proceedings of the ACM Special Interest Group on Data Communication*. ACM, Beijing China, 44–58. doi:10.1145/3341302.3342085

[23] Zeling Zhang, Dongqi Cai, Yiran Zhang, Mengwei Xu, Shangguang Wang, and Ao Zhou. 2024. FedRDMA: Communication-Efficient Cross-Silo Federated LLM via Chunked RDMA Transmission. In *Proceedings of the 4th Workshop on Machine Learning and Systems*. Association for Computing Machinery, New York, NY, USA, 126–133. doi:10.1145/3642970.3655834

[24] Simon Knight, Hung X. Nguyen, Nickolas Falkner, Rhys Bowden, and Matthew Roughan. 2011. The Internet Topology Zoo. *IEEE Journal on Selected Areas in Communications* 29, 9 (2011), 1765–1775. doi:10.1109/JSAC.2011.111002

[25] Mohammad Alizadeh, View Profile, Albert Greenberg, View Profile, David A. Maltz, View Profile et al. 2010. Data Center TCP (DCTCP). *Proceedings of the ACM SIGCOMM 2010 conference* 40, 4 (Aug. 2010), 63–74. doi:10.1145/1851182.1851192

[26] Radhika Mittal, Alexander Shpiner, Aurojit Panda, Eitan Zahavi, Arvind Krishnamurthy, Sylvia Ratnasamy et al. 2018. Revisiting Network Support for RDMA. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. Association for Computing Machinery, New York, NY, USA, 313–326. doi:10.1145/3230543.3230557

[27] Zilong Wang, Layong Luo, Qingsong Ning, Chaoliang Zeng, Wenxue Li, Xinchen Wan et al. 2023. SRNIC: A Scalable Architecture for RDMA NICs. In *20th USENIX Symposium on Networked Systems Design and Implementation*. USENIX Association, Boston, MA, 1–14.

[28] Peihao Huang, Xin Zhang, Zhigang Chen, Can Liu, and Guo Chen. 2024. LEFT: LightwEight and FasT Packet Reordering for RDMA. In *Proceedings of the 8th Asia-Pacific Workshop on Networking*. Association for Computing Machinery, New York, NY, USA, 67–73. doi:10.1145/3663408.3663418

[29] Yanqing Chen, Chen Tian, Jiaqing Dong, Song Feng, Xu Zhang, Chang Liu et al. 2022. Swing: Providing long-range lossless rdma via pfc-relay. *IEEE Transactions on Parallel and Distributed Systems* 34, 1 (2022), 63–75.

[30] Chengyuan Huang, Feiyang Xue, Peiwen Yu, Xiaoliang Wang, Yanqing Chen, Tao Wu et al. 2024. Minimizing buffer utilization for lossless inter-DC links. *IEEE/ACM Transactions on Networking* (2024).

[31] Minfei Long, Jiangping Han, Wentao Wang, Jiayu Yang, and Kaiping Xue. 2024. Lscc: Link-segmented congestion control for rdma in cross-datacenter networks. In *2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS)*. IEEE, 1–10.

[32] Gaoxiong Zeng, Wei Bai, Ge Chen, Kai Chen, Dongsu Han, Yibo Zhu et al. 2022. Congestion Control for Cross-Datacenter Networks. *IEEE/ACM Transactions on Networking* 30, 5 (2022), 2074–2089. doi:10.1109/TNET.2022.3161580

[33] Yantao Geng, Han Zhang, Xingang Shi, Jilong Wang, Xia Yin, Dongbiao He et al. 2023. Delay Based Congestion Control for Cross-Datacenter Networks. In *2023 IEEE/ACM 31st International Symposium on Quality of Service (IWQoS)*. 1–4. doi:10.1109/IWQoS57198.2023.10188700

[34] Minfei Long, Jiangping Han, Wentao Wang, Jiayu Yang, and Kaiping Xue. 2024. LSCC: Link-Segmented Congestion Control for RDMA in Cross-Datacenter Networks. In *2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS)*. 1–10. doi:10.1109/IWQoS61813.2024.10682909

[35] Kai Lv, Jinyang Li, Pengyi Zhang, Heng Pan, Luyang Li, Shuihai Hu et al. 2025. OmniDMA: Scalable RDMA Transport over WAN. In *Proceedings of the 9th Asia-Pacific Workshop on Networking*. Association for Computing Machinery, New York, NY, USA, 135–141. doi:10.1145/3735358.3735373

[36] Yuanwei Lu, Guo Chen, Bojie Li, Kun Tan, Yongqiang Xiong, Peng Cheng et al. 2018. Multi-Path Transport for RDMA in Datacenters. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. USENIX Association, Renton, WA, 357–371.

[37] Mohammad Alizadeh, Tom Edsall, Sarang Dharmapurikar, Ramanan Vaidyanathan, Kevin Chu, Andy Fingerhut et al. 2014. CONGA: Distributed Congestion-Aware Load Balancing for Datacenters. In *Proceedings of the 2014 ACM Conference on SIGCOMM*. Association for Computing Machinery, New York, NY, USA, 503–514. doi:10.1145/2619239.2626316

[38] Naga Katta, Mukesh Hira, Changhoon Kim, Anirudh Sivaraman, and Jennifer Rexford. 2016. HULA: Scalable Load Balancing Using Programmable Data Planes. In *Proceedings of the Symposium on SDN Research*. Association for Computing Machinery, New York, NY, USA, Article 10, 12 pages. doi:10.1145/2890955.2890968

[39] Soudeh Ghorbani, Zibin Yang, P. Brighten Godfrey, Yashar Ganjali, and Amin Firoozshahian. 2017. DRILL: Micro Load Balancing for Low-Latency Data Center Networks. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. Association for Computing Machinery, Los Angeles, CA, USA and New York, NY, USA, 225–238. doi:10.1145/3098822.3098839

[40] Naga Katta, Aditi Ghag, Mukesh Hira, Isaac Keslassy, Aran Bergman, Changhoon Kim et al. 2017. Clove: Congestion-Aware Load Balancing at the Virtual Edge. In *Proceedings of the 13th International Conference on Emerging Networking Experiments and Technologies*. Association for Computing Machinery, Incheon, Republic of Korea and New York, NY, USA, 323–335. doi:10.1145/3143361.3143401

[41] Hong Zhang, Junxue Zhang, Wei Bai, Kai Chen, and Mosharaf Chowdhury. 2017. Resilient Datacenter Load Balancing in the Wild. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. Association for Computing Machinery, Los Angeles, CA, USA and New York, NY, USA, 253–266. doi:10.1145/3098822.3098841

[42] Zhehui Zhang, Haiyang Zheng, Jiayao Hu, Xiangning Yu, Chenchen Qi, Xuemei Shi et al. 2021. Hashing Linearity Enables Relative Path Control in Data Centers. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. USENIX Association, 855–862.

[43] David Wetherall, Abdul Kabbani, Van Jacobson, Jim Winget, Yuchung Cheng, Charles B. Morrey III et al. 2023. Improving Network Availability with Protective ReRoute. In *Proceedings of the ACM SIGCOMM 2023 Conference*. Association for Computing Machinery, New York, NY, USA and New York, NY, USA, 684–695. doi:10.1145/3603269.3604867

[44] Yadong Liu, Yunming Xiao, Xuan Zhang, Weizhen Dang, Huihui Liu, Xiang Li et al. 2025. Unlocking ECMP Programmability for Precise Traffic Control. In *22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25)*. USENIX Association, Philadelphia, PA, 87–106.

[45] Huimin Luo, Jiao Zhang, Mingxuan Yu, Yongchen Pan, Tian Pan, and Tao Huang. 2025. SeqBalance: Congestion-Aware Load Balancing with No Reordering in Data Center Networks. *IEEE Internet of Things Journal* 12, 13 (2025), 25707–25719. doi:10.1109/JIOT.2025.3559878

[46] Radhika Mittal, Vinh The Lam, Nandita Dukkipati, Emily Blem, Hassan Wassel, Monia Ghobadi et al. 2015. TIMELY: RTT-Based Congestion Control for the Datacenter. *ACM SIGCOMM Computer Communication Review* 45, 4 (Sept. 2015), 537–550. doi:10.1145/2829988.2787510

[47] Gautam Kumar, Nandita Dukkipati, Keon Jang, Hassan M. G. Wassel, Xian Wu, Behnam Montazeri et al. 2020. Swift: Delay Is Simple and Effective for Congestion Control in the Datacenter. In *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication*. ACM, Virtual Event USA, 514–528. doi:10.1145/3387514.3406591

[48] Ahmed Saeed, Varun Gupta, Prateesh Goyal, Milad Sharif, Rong Pan, Mostafa Ammar et al. 2020. Annulus: A Dual Congestion Control Loop for Datacenter and WAN Traffic Aggregates. In *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication*. ACM, Virtual Event USA, 735–749. doi:10.1145/3387514.3405899

[49] Parvin Taheri, Danushka Menikkumbura, Erico Vanini, Sonia Fahmy, Patrick Eugster, and Tom Edsall. 2020. RoCC: Robust Congestion Control for RDMA. In *Proceedings of the 16th International Conference on Emerging Networking EXperiments and Technologies*. ACM, Barcelona Spain, 17–30. doi:10.1145/3386367.3431316

[50] Vamsi Addanki, Oliver Michel, and Stefan Schmid. 2022. PowerTCP: Pushing the Performance Limits of Datacenter Networks. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. USENIX Association, Renton, WA, 51–70.

[51] Prateesh Goyal, Preey Shah, Kevin Zhao, Georgios Nikolaidis, Mohammad Alizadeh, and Thomas E. Anderson. 2022. Backpressure Flow Control. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. USENIX Association, Renton, WA, 779–805.

[52] Xiaolong Zhong, Jiao Zhang, Yali Zhang, Zixuan Guan, and Zirui Wan. 2022. PACC: Proactive and Accurate Congestion Feedback for RDMA Congestion Control. In *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*. 2228–2237. doi:10.1109/INFOCOM48880.2022.9796803

[53] Yanqing Chen, Chen Tian, Jiaqing Dong, Song Feng, Xu Zhang, Chang Liu et al. 2023. Swing: Providing Long-Range Lossless RDMA via PFC-Relay. *IEEE Transactions on Parallel and Distributed Systems* 34, 1

(Jan. 2023), 63–75. doi:10.1109/TPDS.2022.3215517

[54] Jiao Zhang, Xiaolong Zhong, Zirui Wan, Yu Tian, Tian Pan, and Tao Huang. 2023. RCC: Enabling Receiver-Driven RDMA Congestion Control With Congestion Divide-and-Conquer in Datacenter Networks. *IEEE/ACM Transactions on Networking* 31, 1 (Feb. 2023), 103–117. doi:10.1109/TNET.2022.3185105

[55] Ke Wu, Dezun Dong, and Weixia Xu. 2024. COER: A Network Interface Offloading Architecture for RDMA and Congestion Control Protocol Codesign. *ACM Transactions on Architecture and Code Optimization* 21, 3 (Sept. 2024), 49:1–49:26. doi:10.1145/3660525

[56] Jiao Zhang, Yuqing Wang, Xiaolong Zhong, Mingxuan Yu, Haoyu Pan, Yali Zhang et al. 2024. PACC: A Proactive CNP Generation Scheme for Datacenter Networks. *IEEE/ACM Transactions on Networking* 32, 3 (June 2024), 2586–2599. doi:10.1109/TNET.2024.3361771

[57] Shaojun Zou, Yi Jiang, Jiacheng Qu, Tao Zhang, Yuanzhen Hu, and Yujie Peng. 2024. Achieving Ultra-Low Latency for Timeout-Less Congestion Control in Data Center Networks. In *2024 IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA)*. 1439–1444. doi:10.1109/ISPA63168.2024.00194

[58] Zirui Wan, Jiao Zhang, Yuxiang Wang, Kefei Liu, Haoyu Pan, Yongchen Pan et al. 2025. RHCC: Revisiting Intra-Host Congestion Control in RDMA Networks. *IEEE Transactions on Networking* 33, 3 (2025), 1–14. doi:10.1109/TON.2024.3524247

[59] Chuhao Chen, Jiarui Ye, Yongbo Gao, Sen Liu, and Yang Xu. 2024. HF^2T: Host-Based Flowlet Fine-Tuning for RDMA Load Balancing. In *Proceedings of the 8th Asia-Pacific Workshop on Networking*. ACM, Sydney Australia, 9–15. doi:10.1145/3663408.3663410

[60] Maciej Besta, Marcel Schneider, Marek Konieczny, Karolina Cynk, Erik Henriksson, Salvatore Di Girolamo et al. 2020. FatPaths: Routing in Supercomputers and Data Centers When Shortest Paths Fall Short. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–18. doi:10.1109/SC41405.2020.00031