USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Express Lane to Efficiency and Reliability: Multi-Dimensional Control in Meta's Express Backbone Network

Faisal Iqbal, Vitaly Neganov, Brian Chang, Rong Rong, Yuanjun Yao, Marek Denis, Qin Zhang, Alexandru Manea, Anton Marchenko, and Ulas Kozat, *Meta;* Aditya Akella, *The University of Texas at Austin;* Ying Zhang, *Meta*

https://www.usenix.org/conference/nsdi26/presentation/iqbal

This paper is included in the Proceedings of the 23rd USENIX Symposium on Networked Systems Design and Implementation.

May 4–6, 2026 • Renton, WA, USA

ISBN 978-1-939133-54-0

Open access to the Proceedings of the 23rd USENIX Symposium on Networked Systems Design and Implementation is sponsored by

جامعة الملك عبدالله للعلوم والتقنية

King Abdullah University of Science and Technology

Express Lane to Efficiency and Reliability: Multi-Dimensional Control in Meta's Express Backbone Network

Faisal Iqbal[1], Vitaly Neganov[1], Brian Chang[1], Rong Rong[1], Yuanjun Yao[1], Marek Denis[1], Qin Zhang[1], Alexandru Manea[1], Anton Marchenko[1], Ulas Kozat[1], Aditya Akella[2], and Ying Zhang[1]

[1]Meta
[2]The University of Texas at Austin

Abstract

Meta's Express Backbone (EBB) network interconnects data centers across the globe. As such, it is one of the largest WANs in terms of capacity, traffic and global reach. EBB's unique multi-plane Software Defined Network (SDN) architecture has passed the test of time with its flexibility, reliability and performance. Nonetheless, in parallel to its success, it has also seen an unprecedented growth in traffic demand and capacity footprint. Along with the rapid rise of AI workloads, challenges in resource utilization and failure recovery started to emerge, driving us to enhance and evolve EBB's control design to increase its efficiency and resiliency. From our operational experiences, we realize that while foundational per-plane Traffic Engineering (TE) control offers simplicity, there still exists a gap to support the above two goals. We propose to introduce control in two additional dimensions: globally across planes and locally at individual device. Specifically, Unequal Cost Multi-Path (UCMP) enables control across planes for efficient load balancing, and Fast Re-Route (FRR) provides control across time slices for facilitating rapid detection and recovery from failures. We present the design, implementation, and production results of UCMP and FRR, demonstrating how EBB can adapt to AI-era demands rapidly without fundamental architectural changes.

1 Introduction

At Meta, we operate one of the world's largest wide area networks, Express Backbone (EBB), that interconnects Meta's data centers globally. EBB is built on a novel multi-plane availability architecture, where the topology is physically divided into parallel topologies (called *planes*), and combined with per-plane Software-Defined Networking (SDN) based centralized control and traffic engineering (Figure 1). As such, each plane serves as an isolated fault domain where traffic engineering decisions, path programming, failures, as well as software/hardware updates are contained within a plane.

EBB's operational model is built around *capacity contracts* that specify service requirements through Class of Service (CoS) classifications and data rates [1]. Each CoS has corresponding Service Level Objectives (SLOs) that guarantee packet delivery rates, ranging from 99.98% availability for highest priority traffic class to 99% availability for the lower priority, cost optimized traffic. EBB's architecture was designed to efficiently utilize available network capacity while consistently meeting these SLO commitments across different service classes, scaling sustainably to accommodate steady demand and capacity growth [6]. Its strict planar isolation and global path optimization worked well during periods of stable, predictable growth and scale of Meta's network, satisfying SLOs of different service classes.

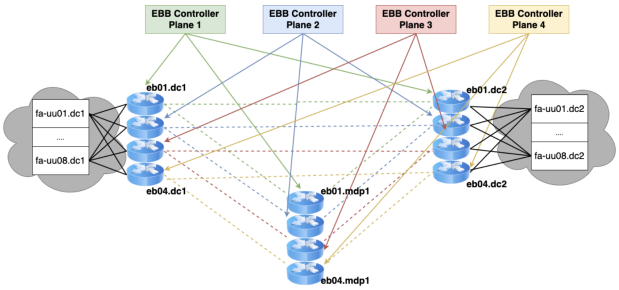


Figure 1: EBB's planar architecture.

However, the emergence of the AI era has fundamentally altered EBB's operational landscape. The scale of this transformation is unexpected in backbone networking: In addition to organic service growth, AI-induced workloads now generate unprecedented cross-region demand, such as bulk data transfers and storage traffic. To meet this demand, EBB's capacity footprint has grown dynamically and is expected to sustain this pace of growth in the coming years. Nonetheless, capacity growth remains outpaced by demand growth, continuously shrinking our operational buffers.

This new operational reality has manifested as increasing SLO violations across traffic classes, with different classes experiencing violations from different aspects of the capacity

crunch. High-priority classes, which handle important production workloads and signaling traffic, have become particularly susceptible to rapid link failures as centralized SDN-based end-to-end failure recovery times now scale with EBB's expanding network diameter. Meanwhile, traffic with lower traffic class — primarily used for bulk operations like offline AI training and data archival, experiences SLO violations due to insufficient capacity, with the shrinking demand-to-capacity ratio combined with strict per-plane isolation, preventing cross-plane traffic redistribution during congestion.

The time-sensitive nature of AI-era operational requirements and these SLO violations, demanded solutions that could be deployed rapidly while maintaining EBB's fundamental architectural strengths. While hardware fabric evolution remains essential for long-term capacity scaling, fast-changing business and performance needs required solutions that can evolve EBB's capabilities on timescales that fabric evolution alone cannot address rapidly enough.

We realized that the fundamental planar architecture remains sound for its primary purposes of fault isolation and sophisticated traffic engineering, suggesting that extensions to the current EBB control stack dimensions, rather than wholesale architectural replacement can address the operational constraints in the needed timely manner. To this end, we evolved EBB's control stack with two complementary SDN-enabled control mechanisms that extend the operational envelope: Unequal Cost Multi-Path (UCMP) routing for fast cross-plane capacity optimization, and a Fast Reroute (FRR) mechanism for high-frequency temporal response. UCMP addresses low-traffic class SLO violations by relaxing strict plane isolation from a demand-splitting perspective, enabling dynamic utilization of global capacity resources while maintaining fault isolation properties, while EBB FRR tackles high traffic class SLO violations by loosening the dependency on end-to-end path protection, introducing local path fail-over capabilities that provide rapid failure recovery independent of global coordination overhead and network diameter scaling issues.

These SDN-based solutions allow us to quickly adapt them to fit EBB's characteristics and keep up with the new operating velocity. Our modular approach enables continuous refinement of UCMP and EBB FRR's algorithmic configurations based on production observations, an important capability given the ever-evolving nature of EBB's workload and scaling characteristics. This flexible approach has also unblocked important EBB migrations and updates. For instance, during the migration of an EBB site, UCMP played a pivotal role in sequentially redirecting traffic away from individual planes. This approach minimized low traffic class congestion while enabling uninterrupted operational processes. Concurrently, EBB FRR mechanisms provided robust failure protection for the available capacity in that site, ensuring that high traffic class SLOs were maintained throughout the migration. EBB FRR's SDN-based solution further contributed to operational resilience: We were able to rapidly migrate FRR path calculation to a new algorithm that prevented any substantial calculation time increase. As a result, the migration proceeded without delays to the established timeline.

This paper presents the design, implementation, and deployment results of these systems to demonstrate how EBB's architecture evolved to meet the demands of the AI era. Our work offers a template for hyperscale backbone network evolution without fundamental redesign.

2 Background and Motivation

In this section, we describe EBB's goals, our operational experiences, and how they made us rethink the design of EBB's control stack.

2.1 Operational Goals of EBB

The high-level goal of EBB is to provide a robust backbone network that can efficiently serve traffic between geographically distributed data centers while meeting strict performance requirements of various Meta applications and services against time-varying demand and capacity conditions. This primary objective translates to ensuring adherence to specific Service Level Objectives (SLOs) for each traffic class providing distinct packet delivery guarantees (see Table 1), while simultaneously enabling steady network growth, ensuring better network utilization, and reducing overall path latency.

To measure adherence to these SLOs, we define a Service Level Indicator (SLI) based on a granular unit we call the *site-pair-qos-minute*. Each minute, for every ordered pair of datacenter regions and each traffic class, we classify the interval as *Good* or *Bad* based on packet loss measurements obtained from a network-wide active probing system that continuously measures packet loss between all region pairs across each class of service. A site-pair-qos-minute is classified as Bad when the mean packet loss ratio across all probe samples for that region pair and class exceeds 0.1% during the interval. The SLI for each traffic class is then computed as the fraction of Good site-pair-qos-minutes out of total measurable minutes over a reporting window. An SLO violation occurs when this SLI falls below the target specified in Table 1 for any traffic class.

2.2 EBB Overview

EBB represents such a purpose-built global backbone network designed to address these operational goals. EBB employs an 8-plane parallel architecture where each plane operates as an independent, parallel network topology fully-connecting all regional sites. This planar design ensures fault isolation—failures within one plane remain contained and

Class	Description	SLO (%)
ICP	Essential traffic for infrastructure; highest priority.	99.98%
Gold	User and business, latency-sensitive traffic.	99.97%
Silver	Default class for most application traffic.	99.95%
Bronze	Cost-optimized (e.g., bulk transfers) traffic.	99%

Table 1: EBB's Traffic classes and availability SLOs.

do not cascade to affect other planes, providing inherent resilience against localized outages.

Each EBB plane incorporates centralized traffic engineering (TE) through per-plane controllers that optimize capacity utilization while ensuring performance objectives for differentiated traffic classes. High-priority traffic demands receive path allocations optimized for minimal latency, while lower-priority traffic is allocated using remaining capacity according to a hierarchical traffic class framework. The centralized approach of TE enables sophisticated traffic optimization and control that would be difficult to achieve through distributed protocols alone.

To maintain operational stability, EBB incorporates demand forecasting coupled with systematic capacity expansion programs. These programs help maintain comfortable operational buffers designed to accommodate both scheduled maintenance windows and unexpected failures or traffic bursts. The combination of architectural fault resilience, precise traffic engineering, and proactive capacity planning has enabled high operational efficiency and network resilience while delivering consistent performance across diverse traffic patterns.

2.2.1 Core Concepts and Terminology

Traffic classes: Application traffic that enters EBB is divided into four classes of service: ICP for essential infrastructure traffic, gold for latency-sensitive and important core services, silver as the default for most applications, and bronze for bulk and heavy network use. Gold, silver, and bronze make up most network traffic and require careful management. Strict priority queueing ensures higher priority classes have better availability and lower packet loss during congestion. This approach allows flexible coordination between centralized network control and individual hosts.

LSP, LSP bundle, and LSP mesh: An LSP (Label Switched Path) is a single MPLS [24] path through the network from source to destination router, computed by the EBB controller based on topology and traffic demands. An LSP Bundle is a collection of multiple LSPs (typically 16) between the same source-destination site pair, providing load balancing and redundancy. An LSP Mesh represents the complete traffic engineering solution containing all LSP bundles across the entire EBB network for all site-to-site connections. The hierarchy flows as LSP Mesh contains LSP Bundles, which contain individual LSPs, where the mesh is recomputed periodically to adapt to network changes.

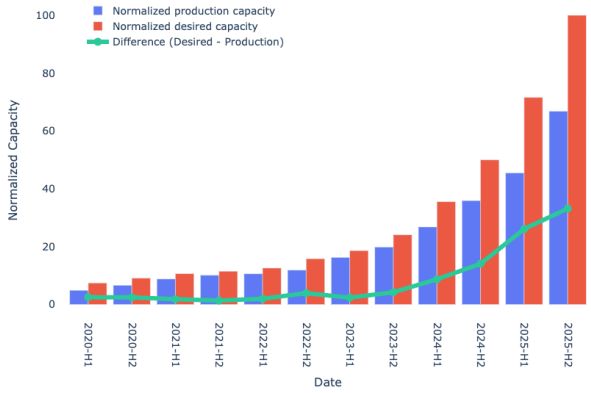


Figure 2: Increasing gap in backbone demand and capacity

Protected capacity. Protected capacity represents the max-flow from a data center site to all other data center sites or from a continent to the other continent after the worst failure. The operations in EBB are constrained to provide enough protected capacity for high-priority (ICP, Gold) traffic classes and Silver traffic class for a cross-oceanic capacity.

2.3 Operational Reality in the AI Era

The fundamental risks of not meeting demand and SLO requirements are tied to having insufficient usable capacity. This manifests in three primary scenarios: (1) capacity becomes unavailable due to network failures and/or scheduled maintenance operations, (2) insufficient capacity construction relative to demand growth results in shrinking demand-to-capacity ratios, and (3) inefficient usage of available capacity.

The AI era has introduced explosive and often unexpected demand growth on EBB. Traffic associated with AI training, checkpointing, data storage, and inference workloads—combined with the organic growth of our existing services—has drastically increased overall network traffic. The velocity of AI-driven demand growth has exceeded the rate at which new network capacity can be constructed and deployed. Network capacity delivery involves complex interdependencies including fiber infrastructure deployment, equipment procurement and installation, and integration testing—processes that require substantial lead times. Consequently, the operational buffers that previously provided resilience against failures and maintenance operations are rapidly diminishing. Figure 2 illustrates the widening gap between the desired EBB capacity, as determined by each period's active demands, and the actual constructed capacity in production. In recent years, both the demand itself, as well as the gap between production and desired capacity, have been increasing exponentially, highlighting the problem of an ever-shrinking operational buffer

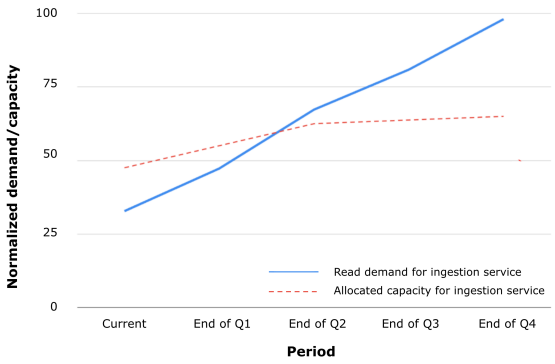


Figure 3: Demand-to-capacity ratio for a data ingestion service used by LLM training in 2024.

in EBB. A closer examination of specific AI-related services further highlights this issue. Figure 3 depicts the reduction in available headroom for the data ingestion service utilized by the LLM training pipeline. The dashed line represents the bandwidth allocated to the ingestion service following TE's path allocation. In summary, the mismatch between demand growth and capacity expansion is impacting not only EBB as a whole, but also individual, business-sensitive services.

This capacity constraint created a cascading effect on EBB's operations. The reduced operational buffers made it increasingly difficult to schedule routine maintenance operations, as taking capacity offline for upgrades or repairs now risked violating SLO commitments. When maintenance operations cannot be scheduled predictably, necessary hardware upgrades and capacity additions are delayed, further exacerbating capacity shortages. Simultaneously, the reduced margin for capacity inefficiency means that sub-optimal traffic engineering decisions can have amplified operational impact. TE solutions that may have been adequate under previous traffic demand conditions now approach congestion thresholds more frequently, requiring more optimized traffic engineering strategies. The increased scale of network operations also introduced new operational challenges in failure recovery. As EBB's network diameter increased with expansion, failure recovery procedures occurred more often and required longer convergence times. Path re-establishment following failures traverses longer topological distances, extending the duration of reduced capacity availability.

3 Challenges

These converging factors—explosive demand growth, capacity delivery constraints, shrinking operational buffers, and increased operational complexity—collectively result in more frequent and prolonged periods of "insufficient capacity." While the original EBB architecture successfully addressed the operational challenges of its time, the aforementioned operational reality introduced by modern AI workloads revealed fundamental design constraints that limited adaptive usable capacity management.

3.1 Cross Plane Heterogeneity

Insufficient capacity manifested as congestion for bronze-class traffic, resulting in subsequent SLO violations. This outcome can be attributed to the rigid planar architecture of EBB's original design. Under this design, each parallel plane takes an equal share of the traffic via equal-cost multi-path (ECMP), under the operational assumption that planes maintain high levels of homogeneity, have similar usable network capacity, and that the network had sufficient operational buffer to accomodate sudden demand bursts. This assumption held well for an extended period of time under stable, accurate demand and capacity forecasting.

However, as operational buffers shrunk in the AI workload era, EBB started observing frequent congestion under this design. Since ECMP does not take each plane's individual capacity into consideration, the planes with a failed or drained devices can be congested while the other healthy planes remain underutilized.

The ongoing reduction in the operational buffer-defined as the margin between total network capacity and demand-has resulted in operational challenges. In particular, we observed a substantial increase in instances where total demand exceeded available usable network capacity. In addition to congestion issues, which predominantly affected Bronze traffic, the period between June 2021 and July 2022 saw a tenfold increase (Figure 6) in protected capacity risks—situations where the next major network failure could lead to loss of ICP or Gold traffic. These trends highlighted the need to address both under- and over-utilization of network planes, which stem from imbalanced demand-to-capacity ratios.

As an illustrative example, Figure 4 demonstrates how straightforwardly partitioning demands across planes with ECMP can lead to inefficiency, and how draining the congested plane is the only way to increase usable capacity without causing congestion. Equal-Cost Multi-Path (ECMP) routing distributes traffic evenly across all available planes. Thus, the maximum usable ECMP capacity without inducing congestion is determined by multiplying the minimum per-plane capacity by the number of operational planes. In Figure 4's case, Plane 8 represents the bottleneck plane with a capacity of 3T, resulting in a total usable ECMP capacity of 24T out of an aggregate 38T. Therefore, traffic demands exceeding 24T will lead to congestion.

Although it may appear counter-intuitive, draining Plane 8 in this scenario enables the system to accommodate a greater volume of traffic. By draining Plane 8, the usable ECMP capacity increases to 5 x 7 = 35T. However, this mitigation strategy requires comprehensive analysis and manual intervention by the on-call team and cannot be executed ad-hoc during every event involving plane imbalance. This was also the

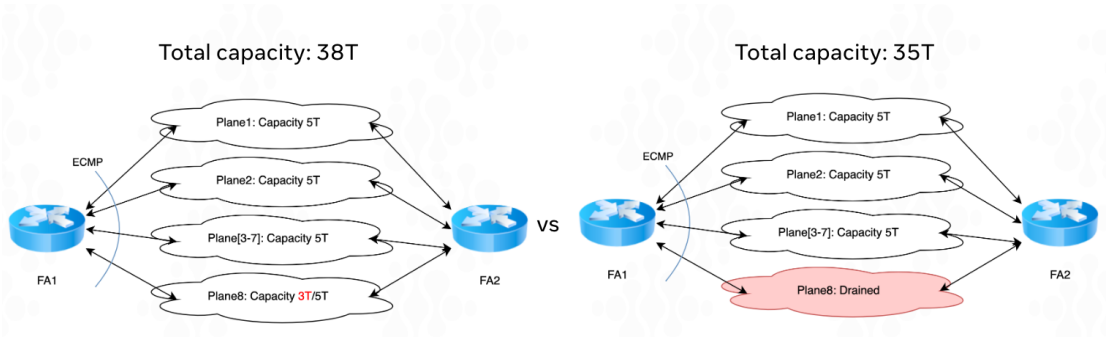


Figure 4: The capacity inefficiency paradox.

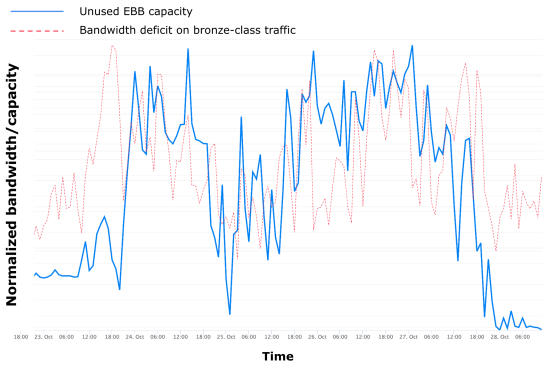


Figure 5: EBB capacity inefficiency

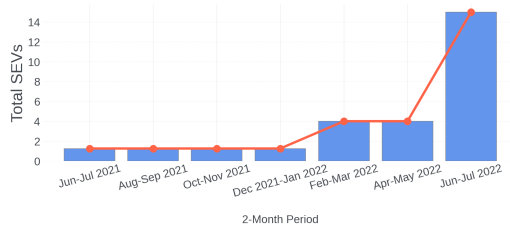


Figure 6: 10-fold increase in protected capacity-related risk events.

approach employed by our on-call team to mitigate capacity-related issues at that time.

EBB faced the challenge of not having enough flexibility to accommodate asymmetry induced by ECMP and imbalanced capacity across planes. Figure 5 shows how we observed occasional bronze traffic congestion, despite having sufficient constructed capacity across all planes. Without the ability to dynamically shift demands across planes, the isolated and single-plane control design for EBB provides limited functionalities to tackle these types of capacity asymmetry. This represents a missing "cross-plane lever" that could significantly improve overall network utilization, important under capacity-constrained conditions where maximizing network resource efficiency becomes paramount.

3.2 Prolonged Failure Recovery Times

For high priority traffic classes, increased scale and operational complexity resulted in longer failure recovery times, which in turn led to extended periods of reduced usable capacity. EBB's centralized per-plane control architecture, while enabling sophisticated traffic optimization, introduces scalability constraints in failure response times. End-to-end Software-Defined Networking (SDN) based backup path switching during link failures requires coordination across the entire network diameter, resulting in recovery times that increase proportionally with network scale.

Link failures, defined as the loss of connectivity due to hardware link outages, represent the most prevalent type of failure within the EBB network. These incidents may result in the complete loss of a link aggregation group (LAG), or alternatively, a reduction in LAG capacity. The duration of such failures varies, ranging from transient events (e.g., brief flaps or short interruptions) to prolonged outages (e.g., fiber cuts). Partial LAG failures diminish the available network capacity and may consequently induce congestion. On the other hand, complete LAG failures lead to traffic black-holing until EBB stops the utilization of the affected LAG. The failures have a huge impact on our Service Level Objectives (SLOs), especially as EBB grew in scale.

Despite controllers pre-calculating and programming backup paths for potential failures, the expansion of EBB in both geography and topology increases the time required to propagate failure events to source routers and reprogram affected LSPs. Topology growth results in more devices and links, and as a result, more failure events even if the mean time between failure remains constant. Additionally, failures on heavily utilized ("hot") links trigger the reprogramming of numerous LSPs, introducing further latency until the new Forwarding Information Base (FIB) is updated in the router ASIC.

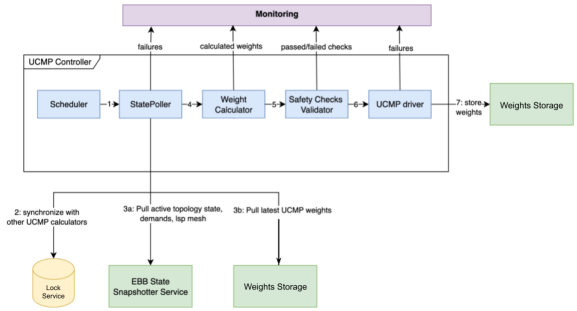


Figure 7: Each controller cycle follows a coordinated sequence, with the following key operations: (1) fetch own plane's network topology, (2) acquire distributed lock, (3) fetch demand and latest weights from other planes, (4) calculate weights, (5) verify safety of new state, (6) update weights to driver, (7) store weights with updated TTL in Weights Storage, then finally release lock.

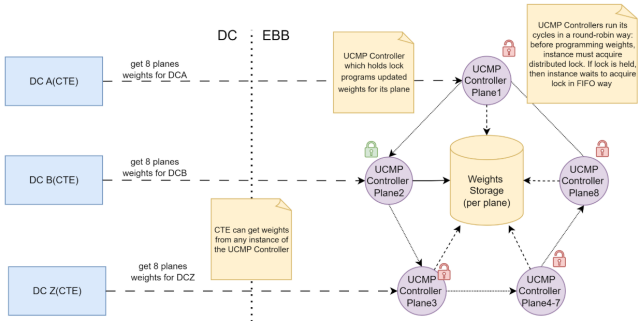


Figure 8: High-level overview of UCMP controller mechanics

4 Cross-plane Control with UCMP

To use available EBB capacity across all planes more efficiently, we need to steer traffic dynamically to EBB planes so that the amount of traffic sent to each plane would be proportional to the plane's available capacity. In order to achieve this, we added cross-plane traffic control to EBB's control stack through a new system named unequal-cost multi-path (UCMP).

4.1 UCMP Overview

UCMP solves cross-plane demand imbalance by providing dynamic per-plane weights that control traffic distribution proportionally to current plane capacity. Instead of equal traffic splitting, UCMP enables weighted traffic distribution based on real-time network conditions. For each source and destination data center pair, UCMP calculates a set of weights $W = \{w_1, w_2, \ldots, w_8\}$ where weight w_i determines the proportion of traffic that plane i should receive. Traffic splitting follows the proportion:

$$\text{Traffic}_{\text{plane}_i} = \text{Total\_Traffic} \times \frac{w_i}{\sum_{j=1}^{8} w_j} \quad (1)$$

UCMP operates at the *per-site-pair* granularity, calculating independent weight sets for each source-destination data center pair across all planes. This fine-grained approach enables localized optimization, as different site pairs may encounter unique capacity bottlenecks and therefore require distinct weight distributions.

4.2 UCMP Software Architecture

The UCMP system is architected as a distributed control plane, with a dedicated controller instance per plane. Figure 7 describes the major components of the UCMP controller, and how they interact with other infrastructure components. Each UCMP controller periodically computes site-pair weights based on the current network state, validates new weights, and persists them with a time-to-live (TTL) parameter to a distributed storage layer. Centralized Traffic Engineering (CTE) [15] applications in datacenters poll these weights at regular intervals, translate them into routing policies, and program the resulting weights into the forwarding infrastructure.

The UCMP controller follows EBB's planar fault-isolation model: a failure in one controller instance affects only one plane's weights, while the other seven planes remain unaffected. Planes can be independently disabled or restarted without disrupting weight computations elsewhere. In production, we roll out changes one plane at a time, using Plane 1 as a canary before expanding to additional planes. Although UCMP weight updates cannot guarantee strict single-plane isolation, the planar design minimizes blast radius and bounds potential disruption.

Although the eight controllers run independently, their weight updates are serialized to preserve global safety. UCMP enforces this via a distributed lock (Figure 8) that provides a FIFO queue, causing controllers to execute in a round-robin manner—only one controller computes and programs weights at a time. Serialization is required because each controller's safety checks must incorporate weight changes from earlier controllers in the same round; otherwise, multiple controllers could act on the same stale global state and produce updates that are individually safe but jointly congestive. If a controller loses its distributed lock session, it terminates immediately to prevent unsupervised concurrent writes. While the execution order may vary across rounds, sequential execution is guaranteed.

Input consistency across controllers is ensured via three mechanisms. (1) At the start of each cycle, every controller reads a centralized snapshot of the global network state (topology and demands for all eight planes). (2) Before writing, it re-reads the currently programmed weights for all planes to incorporate updates committed by the preceding controller. (3) It computes normalized weights for all planes but pro-

grams only its own plane, while enforcing global safety constraints. As an additional safeguard, weights include a TTL: if a controller repeatedly fails to produce safe weights, the stale weights expire and traffic falls back to ECMP, bounding the impact of anomalous distributions.

4.3 Safety Validation and Risk Management

Unlike other components in the EBB control stack, the UCMP system cannot fully adhere to a strictly planar architecture. This limitation stems from the intrinsic coupling within the UCMP routing protocol: an adjustment to the traffic weights on any single plane can immediately influence traffic distribution across all planes. As a result, a fundamental requirement for UCMP is to ensure that any modification to traffic weights does not compromise overall network reliability. To address this requirement, the UCMP architecture incorporates a risk-checking mechanism that operates during every UCMP control cycle. This mechanism evaluates the safety of the newly proposed demand distribution across planes. If the risk checker identifies a potential reliability issue, the UCMP controller withholds the updated weights and does not propagate them. For example, one sub-process of the risk checker simulates the impact of planned weight adjustments on the network and assesses whether the available capacity for high-priority (gold/silver) traffic remains sufficient. If congestion is observed through the simulation, the proposed UCMP weight changes will not be programmed. In cases where failures exhaust the weight TTL installed, UCMP globally falls back to ECMP, so that UCMP operations do not negatively affect network performance or reliability.

4.4 UCMP Algorithm Design

The UCMP controller performs weight computation by modeling a graph with per-plane physical links and several important operational constraints as edges, and combined with a list of site-pair demands as the computation's input. Next, the UCMP controller computes a TE solution under the ECMP assumption that splits all demands equally across planes, generating an *ECMP*-LSP mesh for the paths to be used in waterfilling. The reason to use *ECMP*-LSP mesh rather than using the production mesh computed from demands based on previous ucmp weights, is to break cycle dependency and program ucmp weights on a converged network state to prevent weights oscillation. Iterative exhaustive waterfill is then run over all the edges in the graph repersentation using the paths in the ECMP LSP mesh to decide how much demand can be accepted on each of the paths. The ratio of demands satisfied on each plane for each site pair is then normalized into a set of UCMP weights for each site pair, which is then used to dynamically distribute traffic across planes for the next path allocation cycle.

Algorithm 1 UCMP Demand-Aware Graph Modeling

Input: Physical topology $G = (V, E)$; protected capacity map P; Per-site overload percentage limits L_{site}; Per-site pair overload percentage limits L_{pair}; demand matrix D
Output: UCMP problem graph $G' = (V, E')$
for each $(u, v) \in E$ **do**
    for each plane p **do**
        // Phase 1a: Add Circuit Edges
        ADDEDGE$(u, v, p, c_{u,v})$
        // Phase 1b: Add Protected Capacity Edges
        ADDEDGE$(\emptyset, s, p, P_{in}(s, p))$
        ADDEDGE$(s, \emptyset, p, P_{out}(s, p))$
        // Phase 1c: Add Site Overload Edges
        ADDEDGE$(\emptyset, s, p, L_{site} * \sum_{d \in D: d.dst = s} d.bps)$
        ADDEDGE$(s, \emptyset, p, L_{site} * \sum_{d \in D: d.src = s} d.bps)$
        // Phase 1d: Add Site Pair Overload Edges
        ADDEDGE$(s, d, p, L_{pair} * D[s, d].bps)$

    end for
end for
    return G'

4.4.1 Graph Modeling

One of the important designs in UCMP system is to model both the physical topology, as well as operational constraints, into a graph representation so that the final weights take all of them into consideration. Besides *physical* edges that straightforwardly represent the physical links in the topology, UCMP's graph modeling also adds two types of virtual edges, *protected capacity* edges and *overload* edges.

Protected capacity. Protected capacity (Section 2.2.1) virtual edges in UCMP's graph represents this operational constraint. Protected capacity edges are created for each site and direction on each plane, representing ingress and egress protected capacity of that site. Two trans-Atlantic protected capacity edges are also created between North America and Europe bidirectional *NA-EU* or *EU-NA* on each plane, representing the protected capacity across continent. The amount of protected capacity set on each edge is calculated and provided by a max flow calculation of the physical network topology with the worst failure.

Overload. The operations in EBB are only allowed to drain at most a defined number of routers from the same site simultaneously. To be compliant with this policy, overload edges are added to prevent excessive traffic concentration between specific site pairs on certain planes, as well as prevent any planes from becoming a traffic hot spot on certain site. Site pair overload constraints are created per-plane, per-site-pair (directional), where the edge capacity is set as the total demand between the site pair multiplied by a configured *overloadLimitPercentage*. Similarly, per-site overload constraints are also created for both ingress and egress direc-

tions, and the edge capacity is set as the total ingress/egress demand to/from the specified site multiplied by a configured *overloadLimitPercentage*.

4.4.2 Demand-Aware Water-Filling

From capacity-aware to demand-aware weight calculation. The initial UCMP design computed weights proportional solely to each plane's capacity ratio (available capacity divided by constructed (i.e., steady-state) capacity), taking the minimum across source and destination sites per site pair. While this addressed protected capacity risks by steering traffic away from degraded planes, it had limited efficiency: weights reflected only local site capacity without modeling end-to-end path-level contention or transit bottlenecks. In practice, a plane could appear healthy at its endpoints, but be congested at intermediate links, leading to suboptimal load distribution. To address this, we reformulated weight calculation as a *demand-aware*, multi-commodity flow problem solved by an *iterative exhaustive water-filling* algorithm.

Iterative Exhaustive Water-Filling (IEWF) [4] is used to generate the final UCMP weights after the graph modeling phase. IEWF formulates the weight calculation problem as a constrained optimization problem that aims to maximize network utilization while respecting operational constraints and safety requirements represented as the virtual edges during graph modeling.

Demand allocation ratio on each edge is a prerequisite input to waterfill as well. Paths allocated in ECMP LSP mesh carry a proportion of site pair demands. For each site pair demand, *physical* edge has a utilization as the demand ratio for summing up all the paths traversing through this edge, *protected capacity* edge utilization is the high traffic-class (gold) demand ratio on source site egress edge and destination site ingress edge, *overload* edge utilization is 100% on site pair edge, source site egress edge and destination site ingress edge. If the site pair demand is crossing continent, the trans-Altantic directional protected capacity edge utilization is also the high traffic-class (gold) demand ratio. In order to better scaling the waterfilling process, the problem inputs for the waterfilling is a set of site pair demands, each demand has a number of plane active paths, and each path has all the physical edges, protected capacity edges, overload edges it utilized with a known demand ratio. There is no separation of different QoS traffic classes as well as only 1 path for each plane to be waterfilled. As the name suggests, the exhaustive waterfilling algorithm takes a set of active edges with their own capacities to fill them with the minimum capacitydemand ratio and freeze the edges with that ratio. The process will be exhausted until no active edges to be waterfilled. Each active edge is associated with a set of active paths. In each iteration, IEWF also freezes the paths that traverse through the frozen edges (bottleneck). Each freeze path has an acceptable bandwidth equal to the bottleneck capacitydemand ratio multiply its own site pair demand. The remaining site pair demands are redistributed to other alternative paths for next iteration of waterfill. IEWF ensures max-min fairness by gradually increasing allocation to all paths simultaneously. Paths are frozen only when they encounter bottlenecks, and their traffic allocation is redistributed proportionally among remaining alternatives. This mechanism naturally prioritizes paths with more available capacity while respecting operational constraints. After waterfill, a site pair demand has an accepted bandwidth of each of its path. Normalize each path accepted bandwidth against the largest accepted bandwidth among all the paths becomes the final ucmp weight.

Stabilizing demand-aware weights. Demand-aware weight computation introduces stability challenges because weights depend on time-varying demand as well as topology. To limit unnecessary weight churn, we apply four techniques: (1) compute weights assuming a converged state, where each plane assumes all others run the same algorithm, reducing cross-plane coupling and speeding convergence; (2) maximize weights when aggregate demand is well below capacity to avoid updates from minor, non-congestive imbalances; (3) reserve headroom so small demand fluctuations do not trigger safety-check failures and recomputation; and (4) during congestion, allow bounded overflow, letting one plane exceed its fair share slightly instead of repeatedly redistributing small demand deltas.

5 Failure Recovery with EBB Fast ReRoute

Fast ReRoute (FRR) is a well-known mechanism that provides fast failure recovery on link failure by immediately redirecting traffic over an alternate path. Traditional Interior Gateway Protocol (IGP) [19] based FRR approaches such as Topology-Independent Loop-Free Alternative (TI-LFA) [18] or RSVP-TE [2, 23] based FRR use distributed shortest path first (SPF) algorithms to calculate FRR paths. Network vendors implement these mechanisms in their Network Operating Systems (NOS), generalized for all supported networks and customers.

However, SPF-based FRR mechanisms are not suitable in a capacity constrained network like EBB where the shortest paths is often congested while other, longer paths have available capacity. Reliance on vendor-specific implementations and NOS release cycles also restricts potential customizations by any operator with evolving requirements or scale. Furthermore, a vendor implemented customization will have significant lag between the feature request and until its eventual implementation, delivery, and roll-out in the operator's network.

EBB FRR addresses these challenges through an SDN based FRR mechanism that gives control back to the operator.

5.1 EBB Fast ReRoute

EBB FRR is a FRR solution that provides fast, congestion-aware traffic restoration for link failures by decoupling FRR path calculation from failure mitigation. It leverages the SDN controller's global view of network topology and traffic demands to let it calculate optimal FRR paths for the current network state. The controller incorporates available link capacities, current traffic demands, and link SRLGs to calculate FRR paths for every link in the network. These paths are regularly updated and communicated to the network devices, which only need to monitor network links and perform failure mitigation.

Our simulation results demonstrate that EBB FRR yields substantial improvements in congestion mitigation over SPF-based FRR mechanisms. Furthermore, EBB FRR allows the SDN controller to dictate the quality, quantity, length, and route of the FRR paths. This complete control over path constraints, and more importantly the TE algorithm, differentiates EBB FRR from vendor based path protection mechanisms. We can develop and deploy new TE algorithms on the fly that evolve as our network evolves. We can also expand or relax our path constraints as traffic patterns or network needs and characteristics change. This flexibility is particularly advantageous in a fast growing, multi-vendor environment, as it ensures that traffic restoration mechanisms can keep up with network growth while not being dependent on proprietary vendor implementations.

EBB FRR provides fast restoration for link failures when a link loses most or all of its bandwidth. Due to network constraints, we aim for best-effort protection rather than guaranteeing quick restoration across all traffic and failure types. EBB FRR primarily focuses on fast restoration for high-priority traffic. We do not always have sufficient headroom capacity across the network to guarantee fast protection for low- or medium- priority traffic. On congested FRR paths, lower priority traffic may get dropped in favor of higher priority traffic. We believe this to be a worthwhile trade-off. While the SDN controller attempts to provide SRLG diverse FRR paths, this is also done on a best-effort basis and not guaranteed. In rare cases, the network may experience multiple SRLG failures in quick succession, and both primary and FRR paths may be unavailable until the next controller cycle recalculates paths based on the updated graph.

Note that EBB FRR complements rather than replaces the existing end-to-end path protection mechanism. EBB FRR will immediately protect high-priority traffic until the LSP ingress learns about the link failure. At that point, end-to-end path protection kicks in to reprogram paths for all traffic classes to redirect traffic away from the failed link.

5.2 Software Design

As a decoupled SDN based mechanism, EBB FRR interacts across both the EBB controller and on-device software.

The TE module on the EBB controller consumes network topology, demands, and constraints and used TE algorithm to calculate the FRR paths for every link in the plane.

The driver module on the EBB controller translates the calculated TE paths (intent) into per-device programmable objects. It communicates these programmable objects to their respective network routers.

The on-device agent on the EBB routers monitors link state and capacity, and programs the received FRR paths into hardware on a link failure.

Failure monitoring and FRR activation currently happens over the slow path through on-device agent due to existing NOS software limitations. We will address this limitation by pre-programming the FRR paths in ASIC to enable automatic activation on link failure in *O(1)* time.

5.3 Handling Partial-LAG Failures

EBB FRR was originally designed for full LAG failures, where the entire LAG becomes unusable. In practice, LAGs often degrade instead: some member links stay up, so the LAG remains operational but drops traffic above its reduced capacity. In this case, neither FRR nor end-to-end path protection triggers.

Traditional NOSs mitigate partial failures with *LAG min-links*: if active members fall below a threshold, the NOS brings the whole LAG down to trigger FRR. This is unsuitable for EBB, which aims to use any remaining capacity. Instead, EBB keeps the LAG up and relies on the controller to learn the reduced capacity and reprogram TE LSPs to reduce load.

Since a LAG stays up as long as any member is up, partial failures can cause prolonged loss—including for high-priority traffic—until the controller shifts traffic in a later programming cycle. This is undesirable for high-priority services.

EBB FRR, unlike traditional FRR mechanisms, can be customized to protect against this scenario without sacrificing network capacity. We implemented *EBB FRR for partial-LAG failure* to monitor available LAG capacity against protected, high priority demands. EBB controller tracks the amount of high-priority traffic going over a link (protected demands) and communicates them to the on-device agent. When the available LAG capacity drops below the protected demands, the agent triggers FRR for the LAG to avoid loss to the high-priority traffic. All traffic is rerouted over FRR paths in this case, but only until EBB controller reflects the reduced LAG capacity in subsequent cycle(s) to redirect the excess traffic away from the link. After that, we would continue operating at the reduced capacity without incurring congestion or performance degradation, rather than losing the capacity entirely. Similar to full link failure, FRR activation may result in

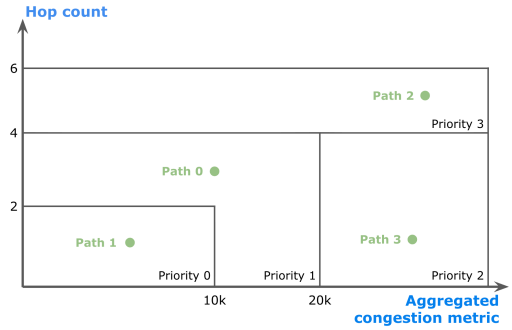


Figure 9: Grid-based policy for MOO in FRR.

congestion drops for low priority traffic to save the protected demands. The protected demands threshold is configurable and currently tracks gold class traffic.

5.4 FRR Algorithm

We aim to allocate FRR paths that minimize congestion loss during link failures while keeping a constraint on FRR path's hop length due to hardware limitations on MPLS label stack size. We also attempt to find SRLG diverse paths to minimize the risk of FRR path going down with the path it's protecting. With these constraints in mind, we define an *edge congestion* metric for FRR paths that accounts for the amount of existing traffic on the edge. We add a penalty on the path if it is sharing SRLG with the link it's protecting. This approach is similar to Reserved Bandwidth Allocation algorithm in EBB's traffic engineering module [6], and results in TE preferring FRR paths with lower total congestion metrics.

We further incorporate a multi-objective optimization (MOO) [5] framework that computes the pareto-optimal paths with two objectives: minimizing congestion metric and the number of hops. From these, we select the least congested FRR path that satisfies the hop constraint. The grid-based approach [25] offers an intuitive and scalable way to specify MOO policies, with each axis representing an objective with multiple thresholds. The space will be divided into multiple rectangles, where each rectangle forms a MOO policy with different priority. Figure 9 visualizes how an FRR path is chosen among candidate paths based on the MOO policy. Each path's objective value falls into one of the policies as shown in the figure. Path 1 is ultimately selected since it satisfies the highest priority policy. This algorithm helps us select a set of FRR paths that meet our constraints and can optimally route the protected traffic with minimal congestion.

6 Results and Deployment

In this section, we present our evaluation of the EBB network after deploying UCMP and EBB FRR. Our experience after deployment demonstrates increasing network efficiency,

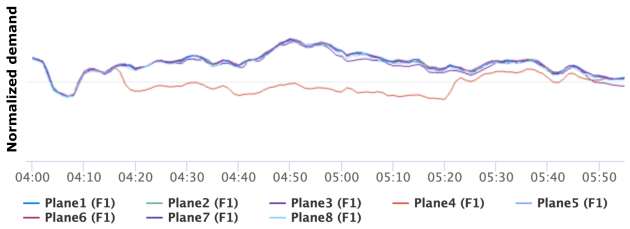


Figure 10: UCMP balancing demands across planes.

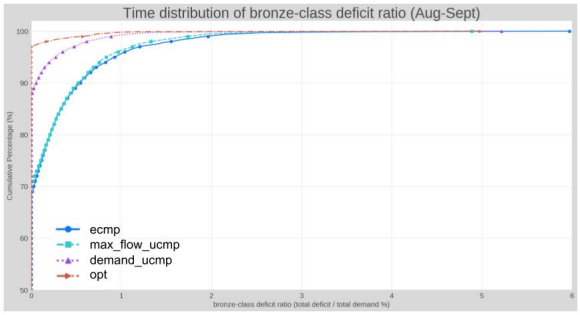


Figure 11: Congestion improvement under UCMP

subsequently resulting in SLO improvement, reduction of congestion, and reduced failure recovery times.

6.1 Balanced Cross-plane Demand

Demand Shifting. Figure 10 shows how UCMP rebalanced traffic across planes at one of the EBB sites during a device maintenance for software upgrade. During this timeframe, we rebuilt devices that collectively accounted from 25% of the sites' capacity one of the planes, resulting in a 25% reduction in capacity for that plane compared to others. UCMP detected the capacity reduction by polling topology state. It estimated the maximum demand sustainable on the affected plane and normalized it by the maximum demand on unaffected planes, yielding a 75% ratio. Thus, the degraded plane carried 75% of the traffic of other planes.

Consistent Congestion Reduction. We also conducted extensive simulations to analyze network congestion improvement for when UCMP was enabled. Figure 11 shows the cumulative distribution of bandwidth deficit ratios for bronze-class traffic, where the deficit ratio is defined as $\frac{TotalDemand - TotalAvailableCapacity}{TotalDemand}$. Four experiments were simulated: *ecmp* (pre-UCMP, equal demand distribution), *max_flow_ucmp* (UCMP based on plane capacity ratio), *demand_ucmp* (our proposed UCMP), and *opt* (estimated optimal by aggregating capacity and demand). *ecmp*, *max_flow_ucmp*, and *demand_ucmp* calculated bandwidth deficit ratios per plane/site pair. Results show that UCMP reduces the deficit ratio by 75% versus ECMP, demonstrating UCMP's effectiveness in demand shifting.

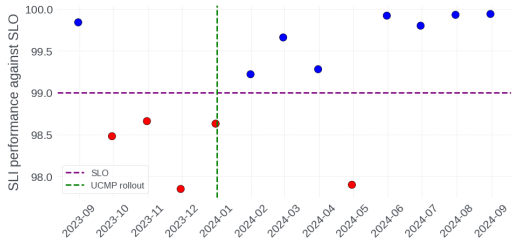


Figure 12: Bronze traffic SLOs after UCMP rollout

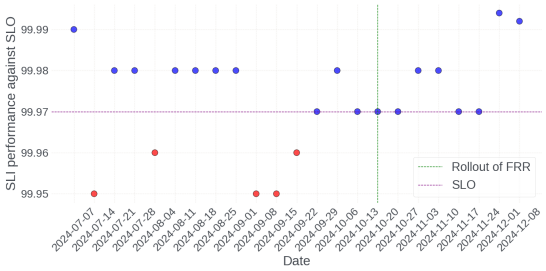


Figure 13: Gold traffic SLOs after EBB FRR rollout

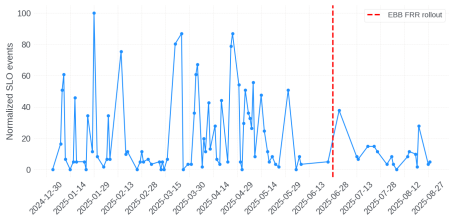
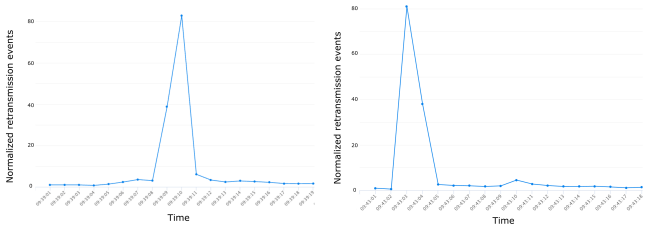


Figure 14: Drop in SLO violation events after partial LAG FRR rollout (after vertical line).



(a) Full LAG failure (b) Partial LAG failure

Figure 15: Traffic convergence times with EBB FRR.

SLO Improvement. Figure 12 illustrates SLO performance for bronze-class traffic from September 2023 to September 2024. Months in which the 99% target was achieved (i.e., performance above the horizontal threshold) are highlighted in blue. Notably, bronze-class SLOs began to improve significantly after the rollout of UCMP in January 2024, with targets consistently met thereafter. This improvement is attributed to UCMP's ability to avoid congestion by leveraging cross-plane demand shifting.

6.2 Failure Recovery

We rolled out EBB FRR for full LAG failures in Fall 2024, followed by partial LAG failures support in Summer 2025. Results show that EBB FRR has been effective in improving network convergence for high priority traffic. With EBB FRR, traffic can be restored in 3-5 seconds or less after a LAG failure. EBB processes over 1000 traffic-impacting FRR events daily, and EBB FRR helps save over 10 billion packets everyday during these events - packets that would otherwise be dropped.

Improved Failure Recovery Times. Traffic convergence was taking up to 18 seconds for full LAG failures before we rolled out EBB FRR. This problem was especially acute for partial LAG failures where it would take anywhere between 15-90 seconds to mitigate traffic loss. EBB FRR reduced restoration times to 3-5 seconds or less for both full/partial LAG failures. Figure 15 shows EBB FRR traffic convergence for full and partial LAG failures, as observed through TCP retransmission events. In both cases, TCP retransmits spiked after the LAG failure but returned to normal within 3 seconds, demonstrating rapid traffic restoration.

SLO Improvement. EBB FRR also positively impacted network SLOs and service experience for high priority traffic. As Figure 13 shows, we are now consistently meeting the network SLOs after completing the EBB FRR rollout. We further improved this after we rolled out EBB FRR support for partial-LAG failures. Figure 14 highlights the noticeable drop in the network events where the high priority traffic experienced loss for 15 seconds or more. These events typically occur due to significant loss in LAG capacity, physical interface flaps, or hardware failures, and EBB FRR reduced a significant source of such events.

Next steps: hardware FRR. EBB FRR failure detection and mitigation is currently handled in software (ASIC→agent propagation, event processing, and route programming), competing with IGP/BGP convergence for CPU and often taking several seconds—worsening with scale or load. To address this, we developed hardware FRR that pre-programs FRR paths in the ASIC, enabling immediate O(1) failover on link loss regardless of scale, CPU load, or event volume.

6.3 Case Study: EBB Site Migration

UCMP and EBB FRR have been instrumental in improving network efficiency and SLO performance in EBB. They have also been important factors in unblocking large maintenances or adapting to network growth after major capacity upgrades. We'll highlight these strength through a case study from late 2024 when we undertook a large migration as part of EBB's capacity expansion efforts.

This migration significantly increased the capacity of our major hub location. This site already boasted extensive connectivity to various regions, and the capacity upgrade would

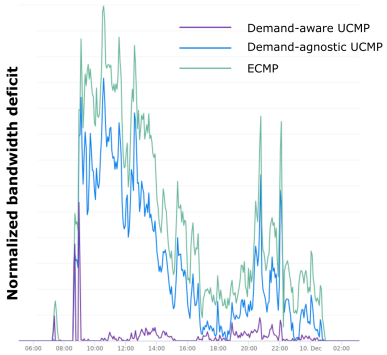


Figure 16: UCMP bandwidth deficit improvements.

further augment existing links and add many new links. However, this dense connectivity also made it a highly active site, making it exceptionally challenging to drain a single device for routine maintenance. Draining 25% of its total capacity across two planes for an extended capacity expansion operation presented an even greater challenge.

Without UCMP, it would have been difficult to prevent network congestion during the drain, as there was no mechanism to redistribute demand to unaffected planes. UCMP's deployment allowed us to execute the drain and complete the migration with minimal impact on bronze-class traffic. Figure 16 shows how different demand shifting algorithms affected the bandwidth deficit based on capacity and demand during the migration. Results show that demand-aware UCMP, the algorithm currently adapted (Section 4.4), reduced bandwidth deficit of bronze traffic by up to 95% compared to ECMP [1].

We also want to highlight EBB FRR's plug-and-play algorithm capability. Our initial CSPF+BFS implementation had scalability limits that became apparent after migration, when the larger topology increased FRR processing time by more than 10-fold. However, as we were already conducting canary testing of EBB FRR with the MOO algorithm (Section 5.4) at that time, we were able to seamlessly transition to MOO in real time and improve TE processing time. Without FRR's SDN-based approach, we would have faced an unacceptable choice: pause capacity expansion to sustain EBB's growth, or disable FRR for months while waiting for a new algorithm.

7 Related Work

Software Defined WANs. EBB, Google's B4 [9, 10], and Microsoft's Blastshield [12] are three inter-data center WANs managed by large enterprises that leverage SDN-based capabilities for operations and management. These WANs employ different mechanisms for reliability: B4 evolved from a flat network to a hierarchy of "supernodes," partitioning each site into control domains so a controller failure affects only traffic transiting that supernode [9]. Blastshield geographically partitions the WAN into disjoint slices, each managed by an independent slice controller, localizing failures to a slice [12]. EBB instead uses a multi-plane architecture with global centralized control for TE and flow programming, naturally containing blast radius and enabling rapid deployment of new control mechanisms [6]. This planar, global design enabled UCMP and FRR, and supports cross-plane demand shifting while preserving strong fault tolerance.

WAN traffic engineering. WAN TE has been studied extensively across ISP, interconnect, and data center settings [3, 7, 8, 11, 13, 14, 16, 17, 20–22]. B4, Blastshield, and EBB also differ in how they scale TE and manage complexity. B4 uses hierarchical partitioning and a two-layer TE abstraction (e.g., max-min waterfilling) to scale computation and contain failures. Blastshield relies on geographic slices with independent controllers and global LP solvers. EBB isolates the network into planes and uses hybrid TE: a central controller with traffic-class-specific algorithms plus distributed switch agents for fast local recovery. Most prior work focuses on improving TE algorithms for performance and scalability. In contrast, we **reconsider the control interface** by extending EBB's planar control dimensionalities with (1) cross-plane control for demand shifting and (2) time-frequent control to further improve performance and scalability.

8 Conclusion

In this paper, we have presented the evolution of Meta's Express Backbone (EBB) network in response to the rapidly increasing demands of global data center interconnectivity and emergence of AI-driven workloads. By augmenting EBB's multi-plane SDN control architecture with Unequal Cost Multi-Path (UCMP) and EBB Fast Re-Route (FRR) systems, we have witnessed significant improvements in both efficiency and resiliency. UCMP enables more granular and effective demand balancing across planes, while FRR ensures rapid recovery from failures. Our results validate that UCMP and FRR allow EBB to meet the stringent requirements of modern workloads without necessitating fundamental architectural changes. As traffic patterns and capacity needs continue to evolve, our approach provides a scalable and adaptable framework for WAN control. Looking forward, we believe that the principles and systems described here will be instrumental in supporting the next generation of distributed applications and AI workloads, ensuring that EBB remains robust, flexible, and future-ready.

Acknowledgments. We would like to thank our colleagues for their contributions to developing and deploying UCMP and EBB FRR. In particular, we thank Sameera Jagadish for his significant contributions to EBB FRR's algorithmic development, and Pavel Kiselev for his significant involvement in EBB FRR's rollout to production. We also thank our shepherd Adrian Perrig, and the anonymous NSDI reviewers, for their valuable feedback.

[1] Demand-agnostic UCMP was a previous iteration of UCMP, and was not adapted due to better performance of the current demand-aware solution.

References

[1] Satyajeet Singh Ahuja, Vinayak Dangui, Kirtesh Patil, Manikandan Somasundaram, Varun Gupta, Mario Sanchez, Guanqing Yan, Max Noormohammadpour, Alaleh Razmjoo, Grace Smith, et al. Network entitlement: contract-based network sharing with agility and slo guarantees. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 250–263, 2022.

[2] Daniel Awduche, Lou Berger, D Gan, Tony Li, Vijay Srinivasan, and George Swallow. Rsvp-te: extensions to rsvp for lsp tunnels. Technical report, 2001.

[3] Jeremy Bogle, Nikhil Bhatia, Manya Ghobadi, Ishai Menache, Nikolaj Bjørner, Asaf Valadarsky, and Michael Schapira. Teavar: striking the right utilization-availability balance in wan traffic engineering. In *Proceedings of the ACM Special Interest Group on Data Communication*, pages 29–43. 2019.

[4] Emilie Danna, Avinatan Hassidim, Haim Kaplan, Alok Kumar, Yishay Mansour, Danny Raz, and Michal Segalov. Upward max min fairness. In *2012 Proceedings IEEE INFOCOM*, pages 837–845. IEEE, 2012.

[5] Kalyanmoy Deb, Karthik Sindhya, and Jussi Hakanen. Multi-objective optimization. In *Decision sciences*, pages 161–200. CRC Press, 2016.

[6] Marek Denis, Yuanjun Yao, Ashley Hatch, Qin Zhang, Chiun Lin Lim, Shuqiang Zhang, Kyle Sugrue, Henry Kwok, Mikel Jimenez Fernandez, Petr Lapukhov, et al. Ebb: Reliable and evolvable express backbone network in meta. In *Proceedings of the ACM SIGCOMM 2023 Conference*, pages 346–359, 2023.

[7] Bernard Fortz, Jennifer Rexford, and Mikkel Thorup. Traffic engineering with traditional ip routing protocols. *IEEE communications Magazine*, 40(10):118–124, 2002.

[8] Victor Heorhiadi, Michael K Reiter, and Vyas Sekar. Simplifying {Software-Defined} network optimization using {SOL}. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pages 223–237, 2016.

[9] Chi-Yao Hong, Subhasree Mandal, Mohammad Al-Fares, Min Zhu, Richard Alimi, Kondapa Naidu B, Chandan Bhagat, Sourabh Jain, Jay Kaimal, Shiyu Liang, et al. B4 and after: managing hierarchy, partitioning, and asymmetry for availability and scale in google's software-defined wan. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 74–87, 2018.

[10] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, et al. B4: Experience with a globally-deployed software defined wan. *ACM SIGCOMM Computer Communication Review*, 43(4):3–14, 2013.

[11] Virajith Jalaparti, Ivan Bliznets, Srikanth Kandula, Brendan Lucier, and Ishai Menache. Dynamic pricing and traffic engineering for timely inter-datacenter transfers. In *Proceedings of the 2016 ACM SIGCOMM Conference*, pages 73–86, 2016.

[12] Umesh Krishnaswamy, Rachee Singh, Nikolaj Bjørner, and Himanshu Raj. Decentralized cloud wide-area network traffic engineering with {BLASTSHIELD}. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 325–338, 2022.

[13] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, Chiun Lin Lim, and Robert Soulé. {Semi-oblivious} traffic engineering: The road not taken. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 157–170, 2018.

[14] Nikolaos Laoutaris, Michael Sirivianos, Xiaoyuan Yang, and Pablo Rodriguez. Inter-datacenter bulk transfers with netstitcher. In *Proceedings of the ACM SIGCOMM 2011 Conference*, pages 74–85, 2011.

[15] Yikai Lin, Mohab Gawish, Shih-Hao Tseng, Lixin Gao, Cen Zhao, John Tracey, Sunyi Shao, Hyojeong Kim, and Ying Zhang. Centralium: A hybrid route-planning framework for large-scale data center network migrations. In *Proceedings of the ACM SIGCOMM 2025 Conference*, pages 395–408, 2025.

[16] Hongqiang Harry Liu, Srikanth Kandula, Ratul Mahajan, Ming Zhang, and David Gelernter. Traffic engineering with forward fault correction. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, pages 527–538, 2014.

[17] Abhinav Pathak, Ming Zhang, Y Charlie Hu, Ratul Mahajan, and Dave Maltz. Latency inflation with mpls-based traffic engineering. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 463–472, 2011.

[18] Liesbeth Roelens, Óscar González de Dios, Ignacio de Miguel, Edward Echeverry, and Ramón J Durán Barroso. Performance evaluation of ti-lfa in traffic-engineered segment routing-based networks. In *2023 19th International Conference on the Design of Reliable Communication Networks (DRCN)*, pages 1–8. IEEE, 2023.

[19] Sandra Sendra, Pablo A Fernández, Miguel A Quilez, and Jaime Lloret. Study and performance of interior gateway ip routing protocols. *Netw. Protoc. Algorithms*, 2(4):88–117, 2010.

[20] Kandula Srikanth, Dina Katabi, Bruce Davie, and Anna Charny. Walking the tightrope: responsive yet stable traffic engineering. 2005.

[21] Martin Suchara, Dahai Xu, Robert Doverspike, David Johnson, and Jennifer Rexford. Network architecture for joint failure recovery and traffic engineering. *ACM SIGMETRICS Performance Evaluation Review*, 39(1):97–108, 2011.

[22] Hao Wang, Haiyong Xie, Lili Qiu, Yang Richard Yang, Yin Zhang, and Albert Greenberg. Cope: Traffic engineering in dynamic networks. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 99–110, 2006.

[23] Wikipedia contributors. RSVP-TE. https://en.wikipedia.org/wiki/RSVP-TE, 2025. [Online; accessed 18-September-2025].

[24] Xipeng Xiao, Alan Hannan, Brook Bailey, and Lionel M Ni. Traffic engineering with mpls in the internet. *IEEE network*, 14(2):28–33, 2000.

[25] Shengxiang Yang, Miqing Li, Xiaohui Liu, and Jinhua Zheng. A grid-based evolutionary algorithm for many-objective optimization. *IEEE transactions on evolutionary computation*, 17(5):721–736, 2013.