

Article

ESMP: Exploring Efficient and Stable Multicast on Multiple Communication Paths

Xin Dong ¹ , Qiuling Yang ² and Deshun Li ^{2,*}¹ School of Information and Communication Engineering, Hainan University, Haikou 570228, China; dongxin@hainanu.edu.cn² School of Computer Science and Technology, Hainan University, Haikou 570228, China; qlyang@hainanu.edu.cn

* Correspondence: lideshun@hainanu.edu.cn

Abstract

Modern communication networks may provide several heterogeneous links between the same pair of devices, including Wi-Fi, 5G, Bluetooth, and SparkLink. Existing multicast schemes often use simple-graph abstractions and therefore cannot distinguish these parallel links. We present *ESMP*, a multi-graph-based heuristic framework for efficient and stable multicast construction over heterogeneous parallel communication links. *ESMP* represents parallel channels as edges with delay and stability attributes. We show that an aggregate-edge-delay-constrained decision variant of the formulation is NP-hard. The framework includes six polynomial-time heuristics: delay-based *DMA* and *DSMA*, stability-based *SMA* and *SDMA*, and stability-delay-ratio-based *RMA* and *MRMA*. Each algorithm derives a metric-specific graph from the original multi-graph and constructs a tree according to its delay-stability preference. We also develop local adjustment strategies for vertex joins, vertex exits, and link dynamics. Experiments on connected synthetic multi-graphs reveal distinct metric preferences. Delay-oriented methods reduce delay, stability-oriented methods improve stability, and ratio-based methods provide stability-aware trade-offs at relatively low delay. In particular, *RMA* favors low delay, whereas *MRMA* uses pair-level average stability-delay information and shows comparatively favorable stability preservation and tree compactness in the evaluated scenarios. These findings characterize heuristic behavior in the evaluated synthetic settings and do not establish general optimality.

Keywords: multicast; multiple communication paths; multi-graph; network dynamics

Academic Editor: Andreas Mauthe

Received: 5 July 2026

Revised: 1 August 2026

Accepted: 4 August 2026

Published: 5 August 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article

distributed under the terms and

conditions of the [Creative Commons](https://creativecommons.org/licenses/by/4.0/)[Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Modern devices may communicate through heterogeneous technologies such as Wi-Fi, 5G, Bluetooth, and SparkLink. These alternatives can improve capacity and resource use, but their delay and reliability characteristics may differ. Multicast delivers data to multiple receivers simultaneously and can reduce redundant transmissions in applications such as video conferencing and media distribution [1,2]. Selecting suitable links and forwarding structures is therefore important for both transmission efficiency and stable delivery.

Multicast optimization has been investigated in satellite, mobile edge, opportunistic, optical network-on-chip, and software-defined networks [3–7]. These studies optimize bandwidth, routing cost, resource allocation, or service quality. Most formulations, however, represent adjacent devices by a single logical edge and thus cannot select among heterogeneous parallel channels.

Multipath protocols and systems exploit path diversity at the transport or application level [8–14]. They improve throughput, resilience, latency, or video quality for connections and sessions. They do not directly construct a shared multicast tree or jointly model its receiver-average delay, path stability, and membership changes.

SDN/NFV-enabled multicast uses centralized control, traffic engineering, service-function placement, and dynamic rule management [7,15–17]. These approaches offer relevant design principles but generally operate on a simplified logical topology. They do not explicitly retain heterogeneous parallel links as independently selectable edges with distinct delay and stability attributes.

We address this gap with *ESMP*, a multi-graph-based heuristic framework for heterogeneous multipath environments. Devices are vertices, while parallel channels are distinct edges characterized by delay and stability. Multicast construction must therefore select both forwarding vertices and physical communication edges. *ESMP* extends shortest-path and Steiner tree ideas to this setting rather than replacing them with an exact optimizer. It also applies the same metric preferences to local adjustment under network dynamics.

The framework comprises six polynomial-time heuristics with distinct metric preferences. *DMA* and *DSMA* are delay-based, *SMA* and *SDMA* are stability-based, and *RMA* and *MRMA* use stability-delay ratios. Experiments on connected synthetic multi-graphs characterize their delay-stability behavior without claiming universal dominance or approximation guarantees.

The main contributions of this paper are as follows:

- We formulate multicast construction on a multi-graph whose parallel communication paths are distinct edges with delay and stability attributes.
- We prove that an aggregate-edge-delay-constrained decision variant is NP-hard and develop six polynomial-time heuristics evaluated by receiver-average delay and multicast-tree stability.
- We design local tree-adjustment strategies for vertex joins, vertex exits, and link-dynamics repair.
- We compare the algorithms with *SPT*, *KMB*, and an equal-weight weighted-sum baseline, *WSM*, in static and dynamic scenarios.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the *ESMP* model. Sections 4 and 5 introduce the multicast generation and adjustment algorithms, respectively. Section 6 reports the experimental evaluation, and Section 7 concludes the paper.

2. Related Work

Multicast enables a source to deliver data to multiple receivers simultaneously, improving bandwidth utilization and data distribution efficiency [18–20]. Research relevant to multicast construction and deployment spans data center networks [21,22], wireless and mobile networks [23,24], Internet of Vehicles scenarios [25], and 5G/6G systems [26,27]. These studies provide useful network-optimization contexts, although not all of them explicitly address multi-graph multicast-tree construction. To clarify the current research status, this section reviews existing studies from five closely related perspectives: classical graph-based multicast-tree construction, efficiency-oriented multicast in specific network environments, reliability- and dynamics-oriented multicast, SDN/NFV-enabled multicast, and multipath transport.

Classical multicast routing is commonly formulated as a tree-construction problem on a weighted graph. Shortest-path tree methods based on Dijkstra’s algorithm [28] provide a natural delay-oriented reference because each receiver is connected through a shortest source-to-receiver path. Steiner tree-based methods, including KMB [29], Takahashi–

Matsuyama [30], and Mehlhorn's faster approximation [31], reduce the total tree cost by allowing non-terminal forwarding vertices. Recent learning-based methods have also been explored for related Steiner tree construction problems [32]. These methods form an important theoretical and algorithmic foundation for multicast routing. However, they are usually defined on simple weighted graphs, where each adjacent vertex pair is represented by at most one edge. As a result, the selection among heterogeneous parallel communication channels and the joint use of delay and stability attributes are outside their original modeling scope.

Efficiency-oriented multicast studies mainly optimize bandwidth consumption, routing cost, resource allocation, energy consumption, throughput, or end-to-end delay in specific network scenarios. In data center and cloud environments, Cepheus [33] accelerates datacenter applications through high-performance RoCE-capable multicast, Cloudcast [34] studies high-throughput and cost-aware overlay multicast in cloud environments, and REMUS [35] improves reliable and scalable multicast services in IaaS datacenters. In wireless and mobile networks, D2D multicast has been optimized through coalitional game theory and fractional programming [36], energy-efficient multi-hop broadcast has been formulated through cost-sharing games [37], and directional transmission with reflection-aware scheduling has been used to reduce multicast delay in millimeter-wave networks [38]. These studies show that multicast optimization must be adapted to the target network environment. Nevertheless, their network models are usually logical or scenario-specific abstractions, and they do not generally preserve several parallel links between the same two devices as separate selectable multicast-tree edges.

Reliability-oriented and dynamic multicast studies focus on maintaining service quality under changing network states, group membership, or forwarding rules. Hawkeye [39] combines deep reinforcement learning and temporal modeling to support dynamic and stateless multicast. Protection-based multicast strategies improve content distribution stability in wireless networks [40]. Delay-aware NFV-enabled multicasting further considers resource sharing and multicast service placement [18]. These works address important aspects of robustness, failure handling, and dynamic operation. However, the stability considered in these studies is usually tied to a particular system architecture, protection mechanism, or service-placement model. They do not explicitly define multicast-tree stability as an average of source-to-receiver path availabilities over selected parallel communication edges, which is the evaluation perspective adopted in *ESMP*.

SDN and NFV have also been widely used to improve multicast routing and traffic engineering. SDN-assisted IP multicast supports delay- and bandwidth-aware video delivery over 5G networks [15]. NFV-enabled multicast has been studied through service-function tree embedding and resource-aware routing [16,41]. Multi-domain SDN multicast traffic engineering further studies distributed multicast optimization across domains [17]. These approaches benefit from centralized or logically centralized network control and can flexibly manage forwarding rules, service functions, and traffic-engineering constraints. Their main abstraction is still commonly a logical topology or a service-function graph, rather than an attributed multi-graph in which Wi-Fi, 5G, Bluetooth, SparkLink, or other heterogeneous channels between the same device pair are represented as separate edges with different delay and stability values.

Multipath transport protocols provide another important line of related work because they explicitly exploit path diversity. MPTCP [8] enables a single transport connection to use multiple paths simultaneously. QUIC [9] provides a modern transport foundation with connection migration, and QUIC-based multipath extensions further explore path management and scheduling at the transport layer [10]. Recent studies have also improved multipath transport for large-scale video services [11], vehicle-to-cloud video streaming [12],

reliable latency-bounded communication [13], and adaptive multipath congestion-control selection [14]. These protocols and systems demonstrate the practical value of using multiple paths for throughput improvement, resilience, and latency reduction. However, they mainly operate at the connection, transport, or application session level. They do not directly construct a shared multicast tree for a receiver group, and they do not address how parallel device-to-device communication edges should be selected when building such a tree.

The above studies indicate that multicast and multipath communication have each been extensively investigated, but their modeling assumptions and algorithmic objectives differ from those considered here. Existing multicast algorithms provide strong references for tree construction, but they typically assume a simple graph. Existing multipath transport methods exploit path diversity, but they are mainly designed for unicast or session-level delivery. SDN/NFV-based multicast methods provide flexible control, but they commonly operate over logical topologies and service-function abstractions. The remaining gap addressed in this paper is therefore a narrower but explicit intersection of these directions: multicast-tree construction on an attributed undirected multi-graph, where parallel communication links between the same device pair remain selectable and where receiver-average delay, path-availability-based stability, and local dynamic adjustment are considered together. This positioning is intentionally limited in scope. *ESMP* should be viewed as a heuristic extension of multicast-tree construction to a multipath link model, rather than as a replacement for existing multicast, SDN/NFV, or multipath transport methods.

3. *ESMP* Model

This section defines efficient and stable multicast over heterogeneous parallel communication links.

3.1. Multi-Path Network Abstraction

Two devices may communicate through channels with different delay and reliability characteristics. *ESMP* represents this network as a connected undirected multi-graph $MG(V, ME)$ without self-loops. Vertices denote devices, and $e_{ijk} \in ME$ denotes the k -th channel between adjacent vertices v_i and v_j . Its delay and stability are d_{ijk} and p_{ijk} , respectively, with forwarding and switching time included in d_{ijk} . Unlike a simple-graph projection, this abstraction does not collapse heterogeneous channels into one logical edge. Retaining them allows tree construction to select both forwarding vertices and physical communication channels.

3.2. Efficient and Stable Multicast on Multiple Paths

Given a multicast group $V_m \subset V$ and source $v_0 \in V_m$, *ESMP* selects forwarding vertices and edges from $MG(V, ME)$ to construct a multicast tree.

Definition 1 (*ESMP* Model). *Given: A connected undirected multi-graph $MG(V, ME)$ without self-loops, where $e_{ijk} \in ME$ denotes the k -th communication edge between vertices v_i and v_j . Each edge has delay d_{ijk} and stability p_{ijk} . Let $V_m \subset V$ be the multicast group and $v_0 \in V_m$ be the source.*

Solution: *Construct a multicast tree $MT(V', E')$ that minimizes delay and maximizes stability, where $V_m \subseteq V'$, $V' \subseteq V$, and $E' \subseteq ME$.*

Delay and stability may conflict because low-delay links can be unstable and stable links can lengthen routes. We distinguish the aggregate edge delay used in the complexity result from the receiver-average delay used in evaluation.

For the complexity analysis, the aggregate edge delay is

$$D_{\text{sum}}(MT) = \sum_{e_{ijk} \in E'} d_{ijk}. \quad (1)$$

Given thresholds D_0 and S_0 , the decision problem asks whether a tree $MT(V', E')$ exists such that $V_m \subseteq V'$, $E' \subseteq ME$, $D_{\text{sum}}(MT) \leq D_0$, and $S(MT) \geq S_0$. Section 3.3 defines stability and receiver-average delay. The aggregate measure is used only in the following result and not in the experimental evaluation.

Theorem 1. *The aggregate-edge-delay-constrained decision variant of ESMP is NP-hard.*

Proof. We reduce the NP-hard Steiner tree decision problem [42]. Consider an instance $I = (G = (V, E), R, c, B)$ with terminals R , positive edge costs $c(e)$, and bound B . Copy every vertex to MG and replace each $(v_i, v_j) \in E$ with one edge e_{ij1} . Set $d_{ij1} = c(v_i, v_j)$ and $p_{ij1} = 1$. Finally, let $V_m = R$, choose any $v_0 \in R$, and set $D_0 = B$ and $S_0 = 1$. This construction is polynomial.

A Steiner tree of cost at most B spans V_m , has aggregate delay at most D_0 , and has stability 1. Conversely, any feasible multicast tree spans R , and its corresponding Steiner tree costs at most B because $d_{ij1} = c(v_i, v_j)$. The instances are therefore equivalent. Stability is always 1 in the constructed instance, so its constraint is automatic. Hence, Steiner tree reduces to this decision variant, which is NP-hard. $\square$

The result establishes the computational difficulty of the aggregate-edge-delay-constrained variant. The proposed algorithms are therefore polynomial-time heuristics and provide no approximation guarantees. They terminate on finite connected simplified graphs and construct feasible trees when those graphs remain connected. The theorem does not directly establish hardness for the receiver-average-delay objective used in the experiments. Complexity and approximation results for that formulation remain future work.

3.3. Properties of ESMP

3.3.1. Multicast Delay in ESMP

Let $V_R = V_m \setminus \{v_0\}$ be the receiver set, with $m = |V_R|$. For $v_r \in V_R$, reception delay is the total delay on its tree path P_r from v_0 . The receiver-average multicast delay is

$$md = \frac{1}{m} \sum_{v_r \in V_R} \sum_{e_{ijk} \in P_r} d_{ijk}. \quad (2)$$

3.3.2. Multicast Stability in ESMP

Multicast stability describes receiver reachability. Let t_{on}^r and t_{off}^r be the observed online and disconnected times of receiver v_r . The time-based stability is

$$ms = \frac{\sum_{r=1}^m t_{on}^r}{\sum_{r=1}^m (t_{on}^r + t_{off}^r)}. \quad (3)$$

In *ESMP*, p_{ijk} is the availability probability of edge e_{ijk} and is the link-level input for construction and adjustment. Under the independent-link approximation, the stability of source-to-receiver path P_r is

$$S(P_r) = \prod_{e_{ijk} \in P_r} p_{ijk}. \quad (4)$$

Multicast-tree stability is the receiver-average path stability:

$$S(MT) = \frac{1}{m} \sum_{v_r \in V_R} S(P_r) = \frac{1}{m} \sum_{v_r \in V_R} \prod_{e_{ijk} \in P_r} p_{ijk}. \quad (5)$$

Here, p_{ijk} , $S(P_r)$, and $S(MT)$ are link-, path-, and tree-level quantities. The observed ms and probability-based $S(MT)$ are related but not identical. The time-based stability measure is time-weighted, whereas multicast-tree stability is an unweighted average of path availabilities. The experiments use the probability-based path and tree stability definitions, and ms serves only as motivation for the stability interpretation. Correlated failures and unequal observation horizons are not modeled.

4. Multicast Generation Algorithms in ESMP

This section presents polynomial-time heuristics for the receiver-average-delay and multicast-stability metrics in Section 3. They provide metric-specific constructions and are not exact algorithms for the NP-hard decision variant.

4.1. Generation Principle

Receiver delay and stability are determined by source-to-receiver tree paths. Delay accumulates along each path, whereas stability is estimated from its link-availability probabilities. Standard path procedures require a simple graph, so each algorithm first simplifies or aggregates the parallel edges in $MG(V, ME)$. It then grows a multicast tree using a path metric.

DMA/DSMA retain minimum-delay edges and optimize path delay/stability, respectively. *SMA/SDMA* retain maximum-stability edges and optimize path stability/delay. *RMA/MRMA* use single-edge or pair-level ratios and additive inverse-ratio path costs. The paired algorithms therefore separate the metric used to simplify the multi-graph from the metric used to construct the tree. Table 1 summarizes these choices.

Table 1. Graph-simplification and tree-construction metrics of the six ESMP multicast generation algorithms.

Algorithm	Group	Graph Simplification	Tree Construction	Preference
<i>DMA</i>	Delay-based	Minimum link delay	Minimum accumulated path delay	Low delay
<i>DSMA</i>	Delay-based	Minimum link delay	Maximum path stability	Delay-aware stability
<i>SMA</i>	Stability-based	Maximum link stability	Maximum path stability	High stability
<i>SDMA</i>	Stability-based	Maximum link stability	Minimum accumulated path delay	Stability-aware delay
<i>RMA</i>	Ratio-based	Maximum single-edge stability-delay ratio	Minimum accumulated inverse-ratio cost	Delay–stability trade-off
<i>MRMA</i>	Ratio-based	Average pair-level stability-delay ratio	Minimum accumulated pair-level inverse-ratio cost	Pair-level trade-off

4.2. Delay-Based Multicast Generation Algorithms

DMA retains the minimum-delay edge for each adjacent pair to form $G_d(V, E)$. It then grows a tree by repeatedly adding the minimum-delay path from the current tree to an uncovered multicast member (Algorithm 1).

Algorithm 1: DMA**Input:** $MG(V, ME)$, $V_m \subseteq V$, $v_0 \in V_m$ **Output:** MT_d Construct $G_d(V, E)$ by retaining the minimum-delay edge for each adjacent pair (v_i, v_j) ;Initialize $MT_d(V'_m, E'_m)$ with $V'_m = \{v_0\}$ and $E'_m = \emptyset$; set $U = V_m \setminus \{v_0\}$;**while** $U \neq \emptyset$ **do** Set $bestDelay = +\infty$, $bestSource = \emptyset$, $bestReceiver = \emptyset$, and $bestParent = \emptyset$; **foreach** $s \in V'_m$ **do** Initialize $dist(v) = +\infty$, $parent(v) = \emptyset$, and $visited(v) = false$ for all $v \in V$; Set $dist(s) = 0$; **while** there exists an unvisited vertex with finite distance **do** Select the unvisited vertex u with the smallest $dist(u)$ and set $visited(u) = true$; **foreach** neighbor w of u in G_d **do** **if** $dist(u) + d(u, w) < dist(w)$ **then** Set $dist(w) = dist(u) + d(u, w)$; Set $parent(w) = u$; **end** **end** **end** **foreach** $r \in U$ **do** **if** $dist(r) < bestDelay$ **then** Set $bestDelay = dist(r)$, $bestSource = s$, $bestReceiver = r$,
 and $bestParent = parent$; **end** **end** **end** Recover the path P^* from $bestSource$ to $bestReceiver$ using $bestParent(\cdot)$; Let x be the last vertex of P^* that belongs to $V(MT_d)$ when traversing from $bestSource$ to $bestReceiver$; Insert the suffix of P^* from x to $bestReceiver$ into MT_d ; Update $U \leftarrow U \setminus (V(MT_d) \cap V_m)$;**end**

Prune non-member leaves until all leaves are multicast members;

Return MT_d ;

Algorithm 1 specifies distance initialization, edge relaxation, parent updates, path recovery, suffix insertion, and pruning. The recovered path can intersect the current tree before reaching its receiver. Inserting only the suffix after the last intersection preserves acyclicity. Thus, the inserted suffix is attached to the existing tree at a single vertex, and all multicast members covered by the updated tree are removed from U . On a finite connected graph, each iteration removes at least one member from U , so the loop executes at most $|V_m| - 1$ times. Pruning also terminates because the tree has finitely many non-member leaves.

SPT and *DMA* use the same G_d but construct different trees. *SPT* merges shortest paths from v_0 to individual receivers. *DMA* instead expands from any current tree vertex and may share more path segments. Consequently, *SPT* is a strong delay-oriented reference

on G_d ; DMA is not expected to outperform it on receiver-average delay. Its purpose is to provide a greedy expansion alternative with different structural behavior.

DSMA uses G_d but selects the path with maximum stability rather than minimum delay. It has the same termination properties as DMA.

Maximum-stability paths use the additive cost $\ell_{ijk} = -\log(p_{ijk})$. Because $0 < p_{ijk} \leq 1$, we have $\ell_{ijk} \geq 0$. Minimizing $\sum_{e \in P} \ell_e$ is equivalent to maximizing $\prod_{e \in P} p_e$. Dijkstra's algorithm therefore applies with the source initialized to zero, other vertices to infinity, and standard non-negative edge relaxation. An edge with $p_{ijk} = 1$ has zero cost and remains valid. DSMA consequently has the same polynomial-order path-search complexity as DMA.

4.3. Stability-Based Multicast Generation Algorithms

SMA retains the most stable edge for each adjacent pair to form $G_s(V, E)$ and applies the same greedy growth procedure. Candidate paths maximize multiplicative stability through the $-\log(p_{ijk})$ transformation. SDMA also uses G_s but minimizes accumulated delay. Thus, SMA is stability-first, whereas SDMA is a delay-oriented construction constrained to stability-preferred parallel edges. Both terminate after at most $|V_m| - 1$ iterations and have the same polynomial-order path-search complexity as the delay-based methods.

4.4. Stability-Delay-Ratio-Based Multicast Generation Algorithms

The stability-delay ratio is introduced as a heuristic scalarization of link delay and stability. It is not intended to represent an exact solution to the joint multi-objective optimization problem. In RMA, the ratio is defined for each physical communication edge, whereas MRMA uses a pair-level aggregate ratio for parallel edges between the same vertex pair. In both algorithms, the inverse ratio is used as the path-construction cost. Under the assumptions $d_{ijk} > 0$ and $0 < p_{ijk} \leq 1$, the resulting edge cost is positive and can be accumulated along a path.

The ratio-based formulation provides a metric-specific way to rank candidate communication links or vertex pairs according to their stability-delay characteristics. However, it is not equivalent to the exact optimization of receiver-average delay and multicast-tree stability. It may also produce different results from weighted-sum scalarization or Pareto-based optimization. Therefore, RMA and MRMA are treated as heuristic alternatives with explicit stability-delay preferences, rather than as universally optimal or computationally superior methods.

4.4.1. RMA

For edge e_{ijk} , RMA defines

$$r_{ijk} = \frac{p_{ijk}}{d_{ijk}}. \quad (6)$$

A larger r_{ijk} denotes greater stability per unit delay. RMA retains the edge with the largest ratio for each adjacent pair, while final performance is evaluated using Section 3.

Because positive ratios are not naturally additive and could favor unnecessarily long paths, path construction uses $c_{ijk}^r = 1/r_{ijk} = d_{ijk}/p_{ijk}$. RMA minimizes the sum of these stability-adjusted delay penalties. Lower delay and higher stability reduce the cost, whereas unstable links increase it. Since every cost is positive, extending a path increases its total cost, and Dijkstra's algorithm applies (Algorithm 2).

Algorithm 2 follows Algorithm 1 but evaluates paths by summed inverse-ratio cost. Each iteration inserts the minimum-cost suffix from the current tree to an uncovered receiver. Non-member leaves are pruned after all receivers are connected. The algorithm provides neither optimality nor approximation guarantees.

Algorithm 2: RMA**Input:** $MG(V, ME)$, $V_m \subseteq V$, $v_0 \in V_m$ **Output:** MT_r For each $e_{ijk} \in ME$, calculate $r_{ijk} = p_{ijk}/d_{ijk}$ and $c_{ijk}^r = 1/r_{ijk} = d_{ijk}/p_{ijk}$;Construct $G_r(V, E)$ by retaining the edge with the minimum c_{ijk}^r for each adjacent pair (v_i, v_j) ;Initialize $MT_r(V'_m, E'_m)$ with $V'_m = \{v_0\}$ and $E'_m = \emptyset$; set $U = V_m \setminus \{v_0\}$;**while** $U \neq \emptyset$ **do** Set $bestCost = +\infty$, $bestSource = \emptyset$, $bestReceiver = \emptyset$, and $bestParent = \emptyset$; **foreach** $s \in V'_m$ **do** Initialize $cost(v) = +\infty$, $parent(v) = \emptyset$, and $visited(v) = false$ for all $v \in V$; set $cost(s) = 0$; Apply shortest-path relaxation on G_r using c^r as the edge cost, and update $cost(\cdot)$ and $parent(\cdot)$; **foreach** $r \in U$ **do** **if** $cost(r) < bestCost$ **then** Set $bestCost = cost(r)$, $bestSource = s$, $bestReceiver = r$, and $bestParent = parent$; **end** **end** **end** Recover the path P^* from $bestSource$ to $bestReceiver$ using $bestParent(\cdot)$; Let x be the last vertex of P^* that belongs to $V(MT_r)$ when traversing from $bestSource$ to $bestReceiver$; Insert the suffix of P^* from x to $bestReceiver$ into MT_r ; Update $U \leftarrow U \setminus (V(MT_r) \cap V_m)$;**end**

Prune every non-member leaf until all leaves are multicast members;

Return MT_r ;

4.4.2. MRMA

To summarize all parallel edges rather than only the best one, MRMA defines the arithmetic-mean pair score

$$mr_{ij} = \frac{1}{n} \sum_{k=1}^n r_{ijk} = \frac{1}{n} \sum_{k=1}^n \frac{p_{ijk}}{d_{ijk}}, \quad (7)$$

where n is the number of edges between v_i and v_j . A larger mr_{ij} indicates better average stability per unit delay across the pair. This mean measures average stability-delay quality, not path availability or failover reliability. A poor edge can reduce mr_{ij} , whereas an identical backup does not increase it. Thus, mr_{ij} is not a redundancy-aware reliability metric and should not be interpreted as a direct measure of multipath resilience.

In G_{mr} , a virtual edge derived from mr_{ij} represents each adjacent pair and guides path construction. MRMA uses the pair-level aggregate ratio mr_{ij} to rank adjacent pairs before tree growth. The pair-level score determines which vertex pairs enter the tree, while the selected physical edge with the largest individual ratio determines the realized delay and stability. In this way, MRMA differs from RMA not only in graph simplification but also in the information retained at the pair level. The final tree remains a subgraph of MG .

MRMA minimizes additive pair-level costs $c_{ij}^{mr} = 1/mr_{ij} > 0$ using the tree-growing procedure in Algorithm 2. Like DMA, it connects at least one uncovered member per iteration and terminates within $|V_m| - 1$ iterations.

Figure 1 summarizes the overall architecture of *ESMP*, including multi-graph abstraction, metric-specific graph construction, multicast-tree generation, and dynamic adjustment.

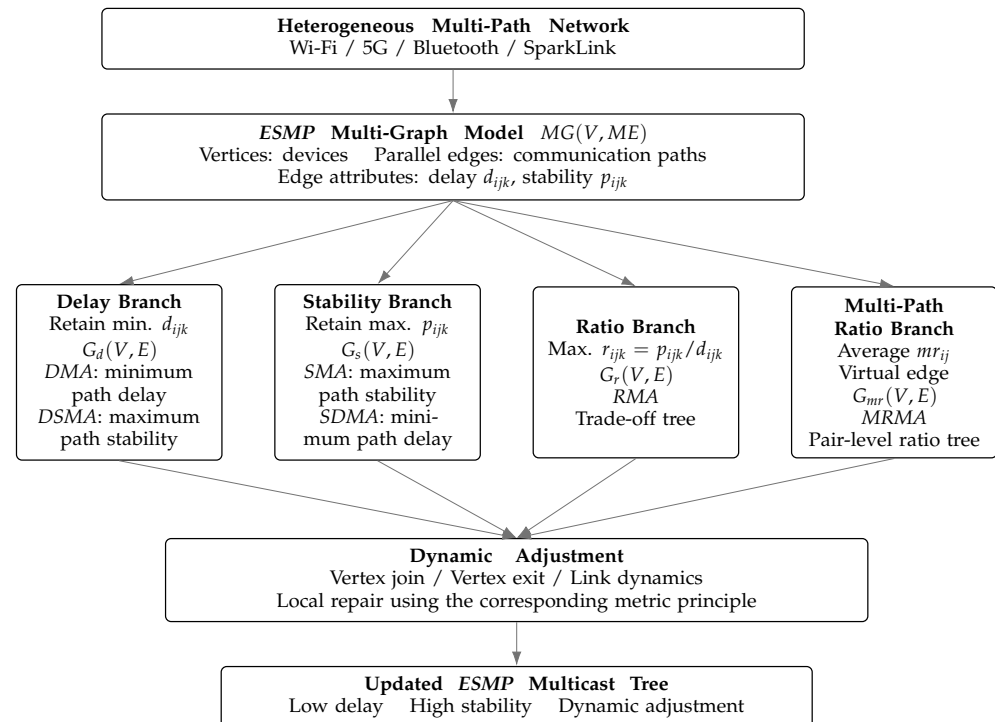


Figure 1. Architecture of *ESMP*. A heterogeneous multipath network is represented by $MG(V, ME)$, converted into metric-specific graphs, and locally updated after vertex joins, vertex exits, or link dynamics.

4.5. Practical Complexity and Scalability Discussion

Let $n = |V|$, $M = |ME|$, $q = |V_m|$, and $m_s \leq M$ be the number of simplified-graph edges. Scanning parallel edges costs $O(M)$. With binary-heap Dijkstra, each path search costs $O(m_s \log n)$. At most q iterations compare paths from up to n tree vertices, giving the conservative bound $O(qn m_s \log n) \subseteq O(qn M \log n)$. This is an upper bound; actual runtime depends on tree growth and the path-search implementation. All six algorithms have the same polynomial order and differ in their simplification and path metrics.

Space is determined by the multi-graph, simplified graph, metric arrays, and tree. Scalability depends mainly on vertices, parallel edges, and group size. Dynamic adjustment usually searches the affected region but may require full reconstruction when local repair cannot preserve connectivity or the selected metric preference.

5. Multicast Adjustment Under Network Dynamics

ESMP locally repairs an existing tree after vertex joins, vertex exits, or link dynamics and reconstructs the tree only when local repair cannot preserve connectivity or the selected metric preference. Joining receivers are connected through metric-specific paths to limit disruption to current receivers. Departing vertices become unavailable for forwarding, and affected downstream branches are reconnected. When a selected tree edge fails, *ESMP* first attempts to replace it with another available parallel edge and then uses a local metric-specific path if no suitable parallel edge exists.

These strategies extend established local-repair principles to the *ESMP* multi-graph rather than defining a new control paradigm. They reuse the delay-, stability-, or ratio-oriented rules developed for static construction. Experiments 2–3 compare adjusted trees

with direct reconstruction, whereas Experiment 4 evaluates conditional repair after sampled tree-edge failures relative to each algorithm's pre-fault tree.

5.1. Link Dynamics

Link dynamics include switching and interruption between adjacent vertices. When a tree edge fails, *ESMP* first selects another available parallel edge using the corresponding metric. For example, if e_{ijk} fails in a *DMA* tree, the algorithm chooses $e_{ijk'}$ with the next-lowest delay. The recovered edge may be restored if it again provides the preferred metric value. If no parallel edge remains, repair is restricted to the interrupted branch whenever possible. The affected vertex or branch is then reconnected to an existing tree vertex through a metric-specific local path.

5.2. Vertex Join

For joining receivers $V_+ \subset V$, the tree is expanded through metric-specific paths. In *RMA*, candidate paths from every current tree vertex to each receiver are evaluated on G_r using c_{ijk}^r . Selecting the minimum-cost candidate preserves the existing tree while adding only the required connection.

Algorithm 3 details the vertex-join procedure for an *RMA* tree.

Algorithm 3: Vertex join in an *RMA* tree

Input: $G_r(V, E)$, current multicast tree MT_r , current multicast group V_m , joining set V_+

Output: updated multicast tree MT_r

```

foreach  $v_i \in V_+$  do
    Set  $bestCost = +\infty$  and  $bestPath = \emptyset$ ;
    foreach  $v_j \in V(MT_r)$  do
        Find the shortest path  $P(v_j, v_i)$  on  $G_r$  using  $c^r$  as the edge cost;
        if  $P(v_j, v_i)$  exists and its cost is smaller than  $bestCost$  then
            Set  $bestCost$  to the path cost and set  $bestPath = P(v_j, v_i)$ ;
        end
    end
    if  $bestPath \neq \emptyset$  then
        Let  $x$  be the last vertex of  $bestPath$  that belongs to  $V(MT_r)$  when traversing toward  $v_i$ ;
        Insert only the suffix of  $bestPath$  from  $x$  to  $v_i$  into  $MT_r$ ;
        Set  $V_m = V_m \cup \{v_i\}$ ;
        Prune leaves that do not belong to  $V_m$ ;
    end
end
Return the updated  $MT_r$ ;

```

Joining vertices are inserted sequentially through the minimum accumulated inverse-ratio path, after which non-member leaves are pruned. Other algorithms use the same framework with their respective metrics.

5.3. Vertex Exit

Departing multicast vertices become unavailable for forwarding. A departing leaf is removed directly. Removing an intermediate vertex disconnects its downstream branches, which must be reattached to the remaining tree. For *RMA*, vertices V_- and their incident edges are excluded from G_r . Affected branches then reconnect through available inverse-ratio paths that avoid the departing vertices. Other algorithms apply the same procedure using their respective graphs and path costs.

6. Simulation Experiments

This section evaluates the proposed algorithms.

6.1. Experimental Settings

We evaluate controlled heterogeneous multipath conditions using connected synthetic multi-graphs. Synthetic generation allows network scale, parallel-link diversity, and link heterogeneity to be varied independently. It is therefore suitable for isolating the metric preferences of the proposed heuristics, although it does not reproduce the structure of Clos, Fat-Tree, or access-network topologies.

A random spanning-tree backbone first guarantees connectivity. A neighbor-selection procedure then adds non-self, non-duplicate adjacent pairs, with each vertex drawing an augmentation target from $[1, 5]$. Existing backbone connections may make the final degree exceed this target, so the interval is not a strict final-degree bound. Each adjacent pair has 1–3 parallel edges, with $d_{ijk} \sim U[1, 10]$ and $p_{ijk} \sim U[0.5, 1]$. After collapsing parallel edges into adjacencies, the simple graphs have mean density 0.01352 and mean degree 4.04. The networks are therefore globally sparse but retain local parallel-link diversity.

The algorithms were implemented in MATLAB R2022a with built-in graph functions and no external libraries. Tests used 64-bit Windows, an Intel Core i5-1240P processor, and 16 GB of memory. Python 3.12 scripts performed statistical aggregation and figure generation. Each setting comprised 50 independently generated instances and multicast groups. Within each trial, every algorithm used the same graph and group to provide matched static comparisons.

MATLAB tic/toc was used to record wall-clock time around each algorithm call. Timing includes tree construction and metric calculation but excludes graph generation, file output, and figure rendering. The reported values reflect the current MATLAB implementations rather than implementation-independent algorithmic efficiency.

The topology generator, proposed and baseline implementations, dynamic-adjustment routines, metric functions, timing code, statistical and figure scripts, processed tables, and trial-level outputs are retained by the authors. These materials are available from the corresponding author upon reasonable request.

Seeds are fixed and recorded. For scale $s \in \{1, \dots, 5\}$ and trial $t \in \{1, \dots, 50\}$, graph seeds are $100000 + 1000s + t$, $200000 + 1000s + t$, $300000 + 1000s + t$, and $400000 + 1000s + t$ for Experiments 1–4. Membership and event streams use deterministic offsets +17, +31, +47, and +61, respectively. These rules regenerate the topologies, multicast groups, membership changes, and failure samples.

Experiments 2–3 use common joining or exiting sets across algorithms. In Experiment 4, each algorithm has a different initial tree. The experiment therefore samples an equal number of failed edges separately from each tree using the experiment seed and algorithm index. This design evaluates conditional repair, not paired responses to one network-level failure event. Table 2 summarizes the settings.

Table 2. Simulation settings for the four experiments. Each setting is evaluated over 50 independent trials.

Exp.	Network	Multicast Setting	Scenario
1	$ V = 300$	$ V_m = 20\text{--}100$	Static construction
2	$ V = 300$	Initial $ V_m = 80$, $ V_+ = 10\text{--}50$	Vertex joins
3	$ V = 300$	Initial $ V_m = 120$, $ V_- = 10\text{--}50$	Vertex exits
4	$ V = 300$	Initial $ V_m = 120$, 10–50 sampled tree-edge failures	Conditional tree-edge-failure repair

Delay is a dimensionless synthetic cost, whereas p_{ijk} is the probability that a link remains available during transmission.

Dynamic Comparison Metrics

For each trial, md^{adj} and S^{adj} denote the adjusted tree's delay and stability; md^{ref} and S^{ref} denote its reference values. The reference is direct reconstruction in Experiments 2–3 and the pre-fault tree in Experiment 4. The delay ratio is

$$DR = \frac{md^{adj} - md^{ref}}{md^{ref}} \times 100\%. \quad (8)$$

The stability-loss ratio is

$$SLR = \frac{S^{ref} - S^{adj}}{S^{ref}} \times 100\%. \quad (9)$$

Positive DR denotes increased delay, whereas negative DR denotes lower delay than the reference. Positive SLR denotes stability loss; negative SLR denotes higher adjusted-tree stability. Ratios are calculated per trial and then averaged. The 95% confidence interval is $\bar{x} \pm t_{0.975, N-1}s / \sqrt{N}$, where $N = 50$ and s is the sample standard deviation.

We compare the six *ESMP* algorithms with *SPT*, *KMB*, and *WSM*. Dijkstra-based *SPT* [28] and the *KMB* Steiner approximation [29] are originally defined on simple graphs. We adapt them by using the delay-simplified projection, which retains each pair's minimum-delay edge. *SPT* merges source-rooted shortest paths, whereas *KMB* provides a tree-sharing-oriented reference.

WSM is a weighted-sum reference. Delay and instability $1 - p_{ijk}$ are first normalized by min–max normalization. Each contributes 50% to the scalar edge cost. Every adjacent pair retains its minimum-cost physical edge, and the same weighted cost guides tree construction.

Because weighted sums define a family of solutions, their results depend on the selected weights and normalization. Our *WSM* results apply only to this 50/50 configuration and do not establish dominance over other weights. *SPT*, *KMB*, and *WSM* respectively provide delay, tree-sharing, and scalarized delay-instability references under common instances and metrics.

The experimental analysis focuses on three questions. First, we examine whether the algorithms exhibit the metric preferences implied by their design, namely delay-oriented, stability-oriented, and ratio-oriented behavior. Second, we compare the proposed methods with the adapted *SPT* and *KMB* baselines and the evaluated equal-weight *WSM* baseline to clarify whether the multi-graph-specific heuristics provide different delay-stability trade-offs. Third, we evaluate whether local adjustment under vertex joins, vertex exits, and sampled tree-edge failures preserves the main properties of the reconstructed or pre-fault reference trees. Therefore, the experiments are intended to characterize algorithmic behavior and trade-offs under controlled synthetic multi-graph settings, rather than to claim universal dominance of one method across all networks and parameter choices.

Unless otherwise stated, all reported values are means over 50 independent trials, and error bars denote 95% confidence intervals.

6.2. Experimental Results

6.2.1. Experiment 1

Experiment 1 evaluates static multicast-tree construction on connected 300-vertex synthetic multi-graphs as $|V_m|$ increases from 20 to 100.

Figure 2 shows clear metric preferences among the tested algorithms. *SPT* achieves the lowest mean delay, 14.508, because it directly follows shortest paths, while *SMA* provides the highest mean stability, 0.448, at the cost of substantially larger delay. Among the proposed trade-off methods, *RMA* remains more delay-oriented, whereas *MRMA* achieves a more balanced behavior. Compared with the evaluated 50/50 *WSM* baseline, *MRMA* reduces the mean delay by 14.9% and improves the mean stability by 67.7%. This indicates that the pair-level average stability-delay information used by *MRMA* provides a more favorable trade-off than the evaluated equal-weight scalarized baseline in these scenarios.

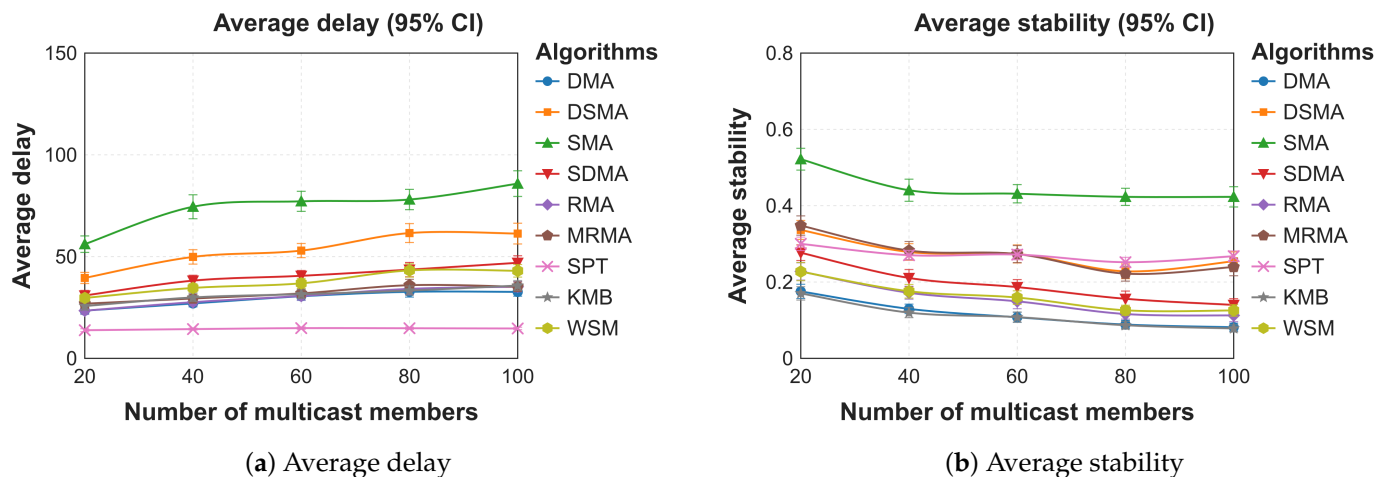


Figure 2. Average delay and stability versus the number of multicast members in Experiment 1. Error bars indicate 95% confidence intervals over 50 independent trials.

As the multicast group becomes larger, *MRMA* also shows a relatively stable scaling trend. When $|V_m|$ increases from 20 to 100, the delay of *MRMA* increases by 31.1%, whereas the delay of the evaluated 50/50 *WSM* baseline increases by 45.0%. Over the same range, the stability of *MRMA* decreases by 31.2%, while that of the evaluated 50/50 *WSM* baseline decreases by 44.8%. These results suggest that *MRMA* maintains its delay-stability trade-off more consistently as the multicast group size increases.

Figure 3 presents the structural and runtime results. *SPT* produces the shallowest but widest multicast trees. Its mean tree width is 27.780, which is much larger than *MRMA*'s 9.852. In contrast, *MRMA* uses the fewest tree vertices on average, 96.400, compared with 119.952 for *SPT* and 97.236 for *WSM*. This indicates that *MRMA* can obtain a compact forwarding structure while still preserving a favorable delay-stability balance.

The prototype runtimes in Figure 3 generally increase as the multicast-group size grows. *MRMA* has a mean runtime of 14.42 s, close to *WSM*'s 14.53 s, whereas *SMA* has the highest mean runtime among the six algorithms. *SPT* and *KMB* are faster in the current implementation because they are applied to projected simple graphs and rely more directly on optimized built-in graph routines. Therefore, the runtime results should be interpreted as implementation-level measurements rather than implementation-independent algorithmic comparisons.

Overall, *DMA* favors low delay, *SMA* favors stability, and *MRMA* provides a favorable delay-stability trade-off with compact tree structures. These findings show that the proposed metric-specific heuristics behave as intended, but the conclusions remain limited to the evaluated synthetic settings, selected baselines, and the 50/50 *WSM* configuration.

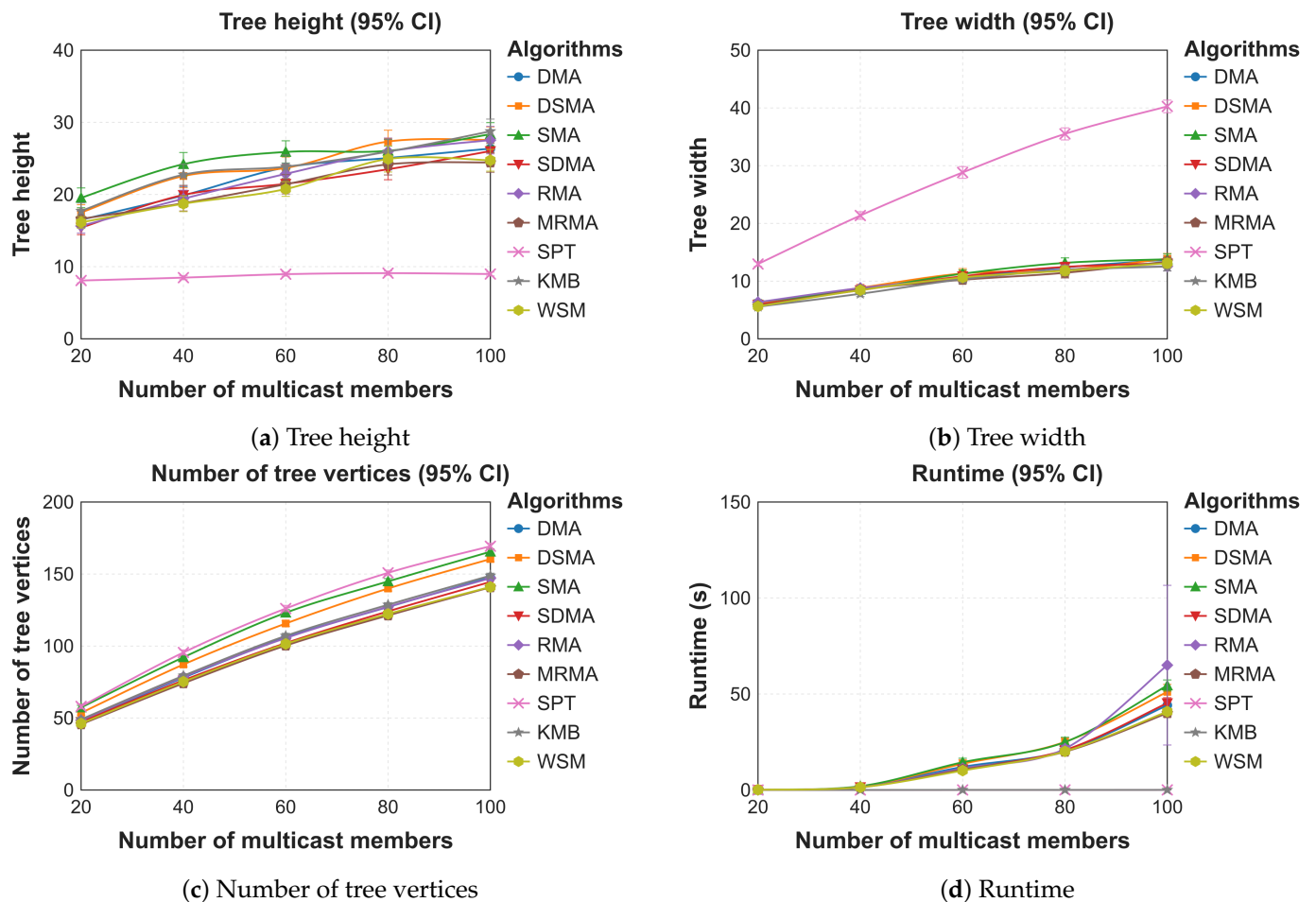


Figure 3. Tree height, tree width, number of tree vertices, and runtime versus the number of multicast members in Experiment 1. Error bars indicate 95% confidence intervals over 50 independent trials.

6.2.2. Experiment 2

Experiment 2 adds 10–50 receivers to an 80-member group on 300-vertex networks. Each adjusted tree is compared with direct reconstruction for the updated group.

In terms of delay, Figure 4a shows that *SPT* achieves the lowest mean delay, 14.665, because it directly follows shortest paths. Among the proposed *ESMP* algorithms, *DMA* keeps the lowest mean delay, 29.840, followed by *RMA* with 32.383 and *DSMA* with 32.609. Compared with the evaluated 50/50 *WSM* baseline, *RMA* reduces the mean delay by 27.6%, while *MRMA* reduces it by 7.6%. The delay-ratio results in Figure 4b further show that *SPT* has the most favorable relative-delay outcome, meaning that its adjusted trees have the largest delay reduction relative to direct reconstruction. Among the proposed *ESMP* algorithms, *DMA* has the most favorable mean delay ratio, whereas *SMA* is the most affected in relative delay.

In terms of stability, Figure 4c shows that *SMA* provides the highest mean stability, 0.409, but this comes with substantially larger delay. *SDMA* and *MRMA* also show stronger stability, with mean stabilities of 0.250 and 0.226, respectively. Compared with the evaluated 50/50 *WSM* baseline, *MRMA* improves the mean stability by 3.3%, whereas *RMA* has lower stability because it gives stronger priority to delay. The stability-loss results in Figure 4d show that *MRMA* is least affected in stability, with the smallest mean stability-loss ratio, 1.26%. This value is lower than *SMA*'s 2.89%, *RMA*'s 3.02%, and *WSM*'s 4.90%.

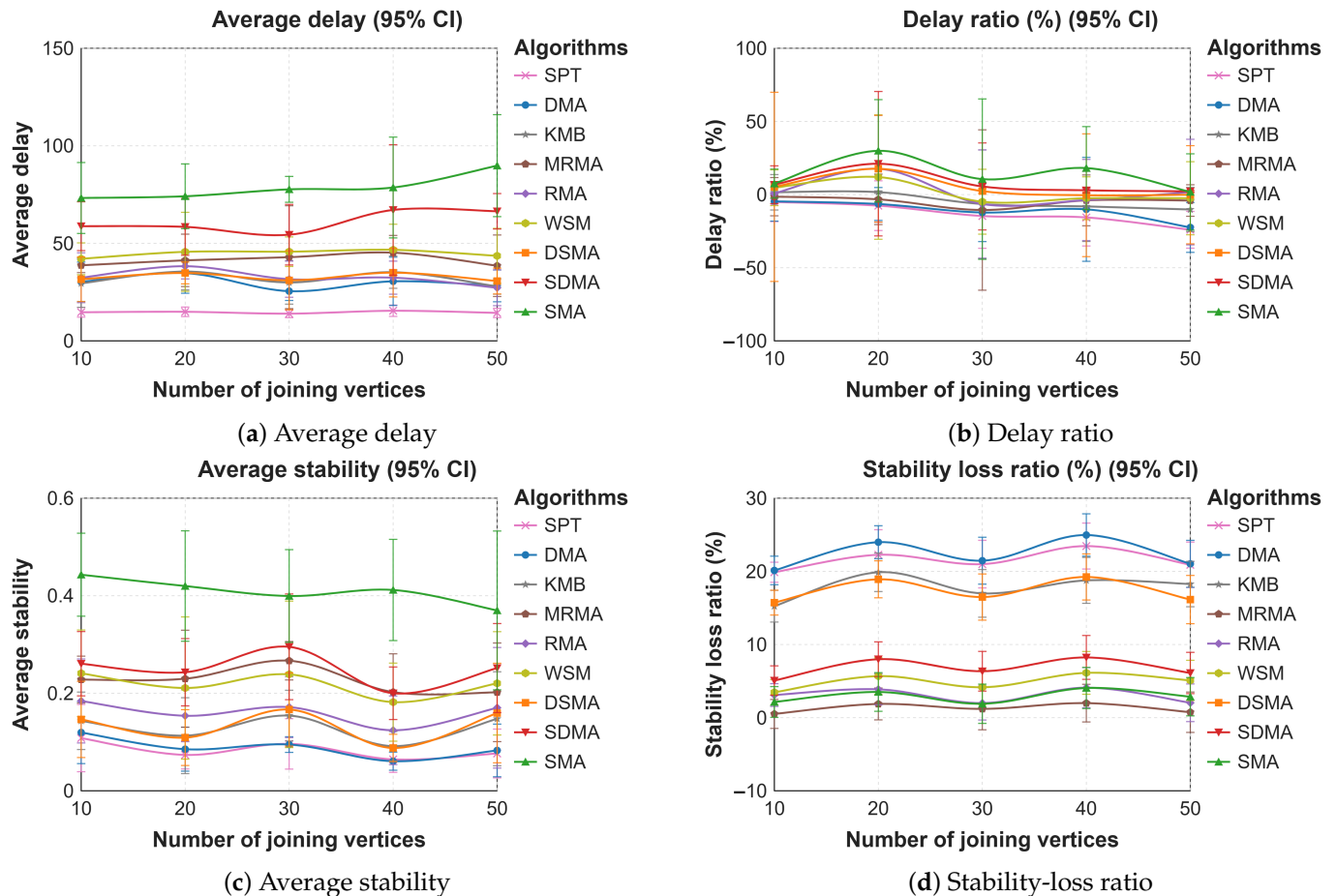


Figure 4. Delay and stability performance under vertex joins for different algorithms. Points and error bars represent mean values and 95% confidence intervals over 50 independent trials.

Overall, Experiment 2 shows that receiver joins affect delay and stability differently across algorithms. *SPT* has the most favorable relative-delay outcome, while *DMA* shows the strongest delay preservation among the proposed *ESMP* algorithms. In contrast, *MRMA* is least affected in stability and achieves slightly higher mean stability than the evaluated 50/50 *WSM* baseline. These results indicate that delay-oriented methods are more suitable when insertion delay is the primary concern, whereas *MRMA* provides a more stability-preserving local adjustment strategy under receiver joins.

6.2.3. Experiment 3

Experiment 3 removes 10–50 vertices from a 120-member group. Departing vertices and incident edges are excluded from both local repair and direct reconstruction. Common graphs and exit sets are used across algorithms.

In terms of delay, Figure 5a shows that *SPT* achieves the lowest mean delay, 20.086, because it directly follows shortest paths. Among the proposed methods, *DMA* also keeps a low mean delay of 30.006. For the ratio-based trade-off methods, *RMA* and *MRMA* obtain mean delays of 31.819 and 32.784, respectively, which are 19.4% and 17.0% lower than the evaluated 50/50 *WSM* baseline. The delay-ratio results in Figure 5b show that *DMA* has the lowest mean delay ratio among the proposed *ESMP* algorithms, 3.19%. Among the ratio-based methods, *MRMA* is less affected than *RMA*, with mean delay ratios of 5.51% and 10.23%, respectively.

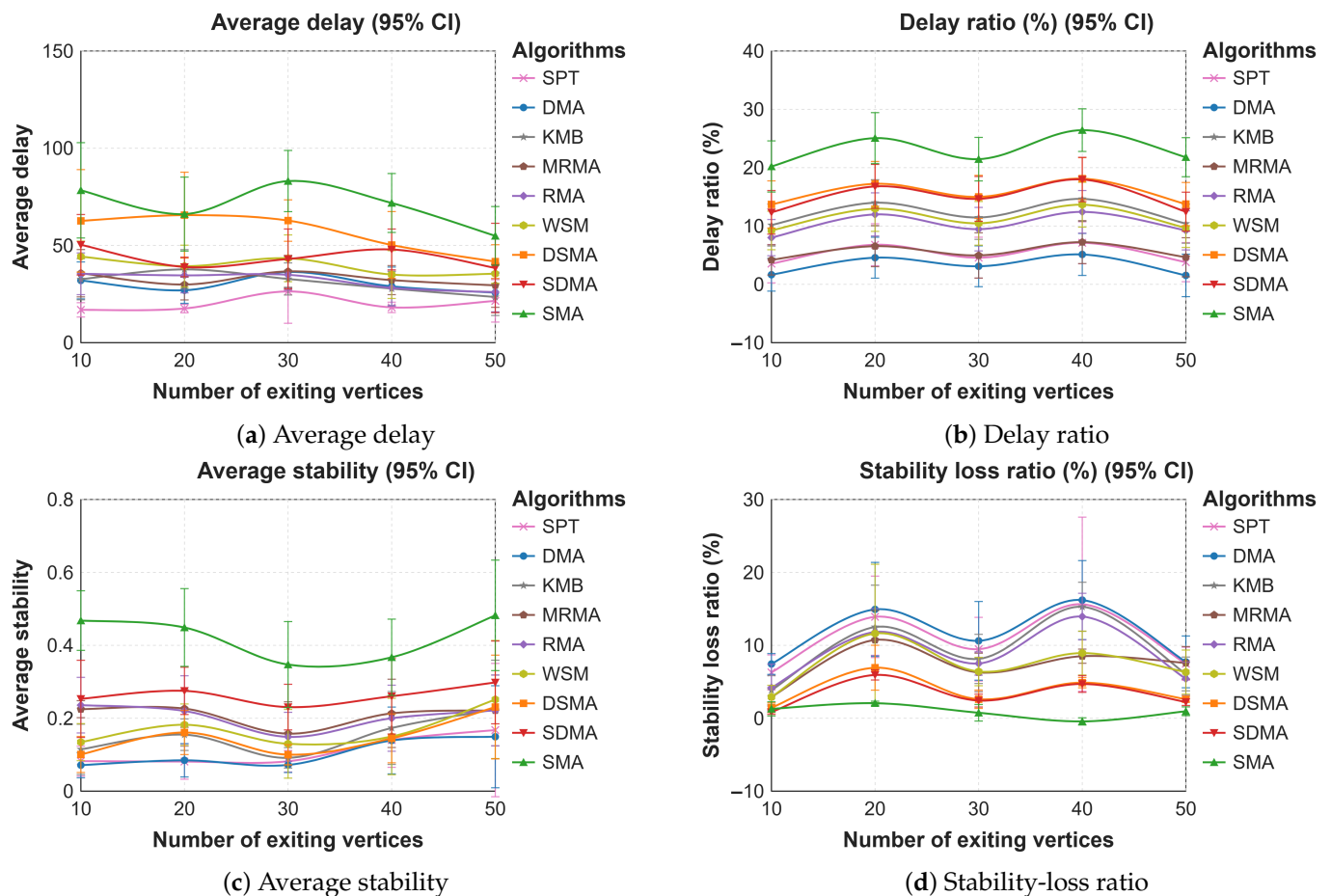


Figure 5. Delay and stability performance under vertex exits for different algorithms. Points and error bars represent mean values and 95% confidence intervals over 50 independent trials.

In terms of stability, Figure 5c shows that SMA provides the highest mean stability, 0.423, while SDMA reaches 0.263. Among the ratio-based trade-off methods, MRMA and RMA achieve mean stabilities of 0.209 and 0.205, respectively, both higher than the evaluated 50/50 WSM baseline's 0.170. Compared with the evaluated 50/50 WSM baseline, MRMA improves mean stability by 23.1%, and RMA improves it by 21.0%. The stability-loss results in Figure 5d show that SMA is least affected overall, with a mean stability-loss ratio of 0.92%. Among the trade-off methods, MRMA has a mean stability-loss ratio of 7.21%, slightly lower than WSM's 7.24% and lower than RMA's 8.58%.

Overall, Experiment 3 shows that vertex-exit repair preserves the intended metric preferences of the algorithms. SPT and DMA remain delay-oriented, while SMA and SDMA retain stronger stability. Among the ratio-based methods, RMA provides slightly lower absolute delay, whereas MRMA is less affected than RMA in both delay ratio and stability-loss ratio. These results indicate that MRMA provides a comparatively stable trade-off among the ratio-based methods under vertex exits, although the conclusions remain limited to the evaluated exit distributions and the selected 50/50 WSM configuration.

6.2.4. Experiment 4

Experiment 4 evaluates conditional repair of 120-member trees after 10–50 sampled tree-edge failures. Sampling weights are proportional to instability $1 - p_{ijk}$, so less stable tree edges are more likely to be selected as failed edges. Delay and stability ratios characterize the repair quality after these failures occur.

In terms of delay, Figure 6a shows that *SPT* has the lowest mean delay, 21.458, because it remains strongly shortest-path-oriented. However, Figure 6b shows that *SPT* is also the most affected by edge failures, with a mean delay ratio of 43.88%. Among the proposed methods, *RMA* has a lower absolute mean delay, 33.940, which is 25.1% lower than *WSM*. *MRMA* has a mean delay of 39.961, 11.8% lower than *WSM*. More importantly, *MRMA* is least affected in delay after repair, with the smallest mean delay ratio, 1.44%, compared with 2.68% for *RMA* and 4.31% for *WSM*.

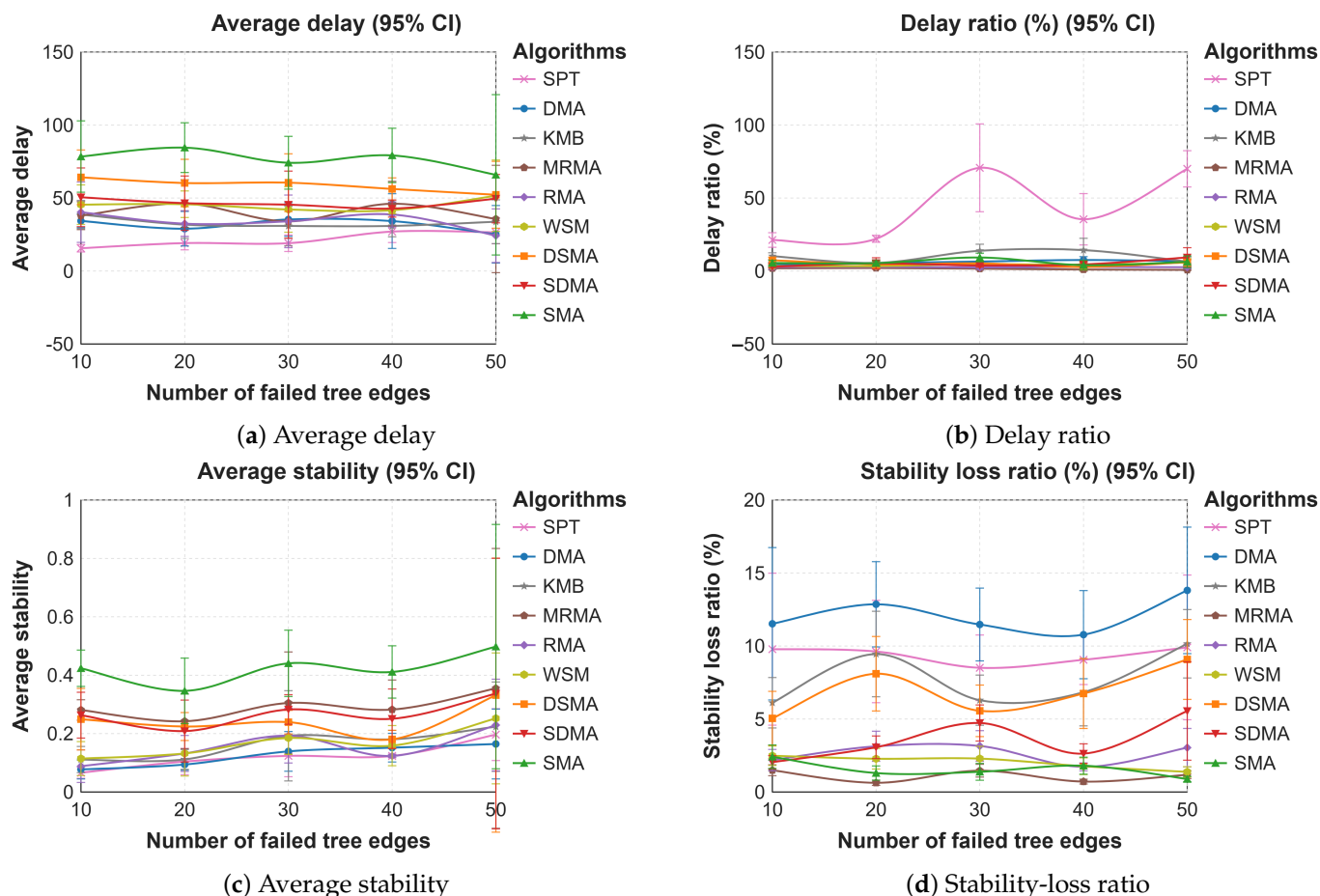


Figure 6. Delay and stability performance under sampled tree-edge failures for different algorithms. Points and error bars represent mean values and 95% confidence intervals over 50 independent trials.

In terms of stability, Figure 6c shows that *SMA* provides the highest mean stability, 0.424, but it also has the largest mean delay among all methods. Among the trade-off methods, *MRMA* reaches a mean stability of 0.293, which is 73.9% higher than *WSM*'s 0.169 and 91.2% higher than *RMA*'s 0.153. The stability-loss results in Figure 6d further show that *MRMA* is least affected in stability, with the smallest mean stability-loss ratio, 1.09%. This value is lower than *SMA*'s 1.56%, *WSM*'s 2.04%, and *RMA*'s 2.65%.

Overall, Experiment 4 shows that local repair remains effective under sampled tree-edge failures, but different algorithms are affected in different ways. *RMA* maintains a stronger advantage in absolute delay, whereas *MRMA* is less affected in both delay ratio and stability-loss ratio. These results indicate that *MRMA* provides the most stable repair behavior under the evaluated failure setting, while *RMA* remains preferable when lower delay is the primary objective. The conclusions characterize conditional repair quality after sampled tree-edge failures and should not be interpreted as universal resilience under all possible network-failure patterns.

6.3. Overall Analysis of Experimental Results

The experimental results show that the algorithms exhibit clear metric-specific preferences rather than a single uniform ranking. *SPT* consistently provides the strongest delay reference because it directly follows shortest paths, but its stability and tree structure are not always favorable. *DMA* also remains delay-oriented among the proposed *ESMP* algorithms. In contrast, *SMA* and *SDMA* give stronger priority to stability, but this usually comes with larger delay. These results confirm the inherent conflict between delay minimization and stability preservation in the multi-graph multicast setting.

Among the trade-off methods, *RMA* and *MRMA* show different behaviors. *RMA* favors high stability per unit delay and therefore stays closer to delay-oriented methods. *MRMA* uses pair-level average stability-delay information and provides a more balanced trade-off. In Experiment 1, *MRMA* reduces mean delay by 14.9% and improves mean stability by 67.7% compared with the evaluated 50/50 *WSM* baseline. It also uses the fewest tree vertices on average, indicating a compact forwarding structure.

The dynamic adjustment experiments further show that local repair preserves the intended metric preferences. In Experiment 2, *SPT* has the most favorable relative-delay outcome after receiver joins, while *DMA* has the most favorable delay ratio among the proposed *ESMP* algorithms. However, *MRMA* is least affected in stability, with the smallest mean stability-loss ratio. In Experiment 3, vertex exits again preserve the delay-oriented behavior of *SPT* and *DMA*, while *MRMA* is less affected than *RMA* in both delay ratio and stability-loss ratio among the ratio-based methods. In Experiment 4, under sampled tree-edge failures, *MRMA* has the smallest mean delay ratio and the smallest mean stability-loss ratio, showing the most stable repair behavior in that failure setting.

The equal-weight *WSM* baseline provides a useful scalarized comparison, but its performance depends on the selected normalization and 50/50 weighting. Therefore, the results should not be interpreted as proving universal superiority over all weighted-sum or Pareto-based methods. Instead, they show that retaining parallel edges and applying metric-specific simplification rules can produce different and controllable delay-stability behaviors.

Overall, no method is best on every metric. Delay-oriented methods are preferable when low delay is the main objective, while stability-oriented methods are suitable when higher path availability is required. *MRMA* provides a comparatively consistent trade-off among the ratio-based methods across the evaluated static and dynamic scenarios. These findings remain limited to connected synthetic multi-graphs, the selected baselines, the evaluated 50/50 *WSM* configuration, and the tested event models.

7. Conclusions

ESMP formulates multicast construction on a multi-graph whose parallel edges have delay and stability attributes. An aggregate-edge-delay-constrained decision variant is NP-hard. We therefore developed six polynomial-time heuristics with delay, stability, or stability-delay-ratio preferences, together with local repair for vertex and link dynamics.

Experiments on connected synthetic multi-graphs confirm these metric preferences. *SPT* provides the low-delay reference, *SMA* favors stability, and *RMA*/*MRMA* provide stability-aware trade-offs. *MRMA* additionally uses pair-level average information. These findings are limited to the selected synthetic settings, baselines, equal-weight *WSM*, and event designs. In particular, Experiment 4 measures conditional repair after sampled tree-edge failures rather than resilience under a common network event.

Future work will evaluate real or trace-driven Clos/Fat-Tree, 5G transport, and heterogeneous wireless topologies. It will also examine physical channels, switching over-

head, additional baselines, larger-scale optimized implementations, and receiver-average-delay complexity.

Author Contributions: Conceptualization, X.D. and D.L.; data curation, X.D.; software, X.D.; funding acquisition, Q.Y. and D.L.; methodology, X.D. and D.L.; project administration, Q.Y.; resources, D.L.; writing—original draft, X.D. and D.L.; writing—review and editing, X.D. and D.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under Grants 62462028, 62162021, and 62362026; the specific research fund of the Innovation Platform for Academicians of Hainan Province under Grant YSPTZX202314; and the Key Project of Hainan Province under Grant ZDYF2023GXJS158.

Data Availability Statement: The experiments use synthetic data generated by the procedures described in Section 6.1; no external dataset was used. The fixed random-seed rules are reported in the manuscript. The topology-generation code, algorithm and baseline implementations, dynamic adjustment routines, metric calculation code, runtime measurement code, statistical aggregation scripts, figure-generation scripts, processed result tables, and raw trial-level outputs are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Li, Y.; Wang, N.; Zhang, W.; Liu, Q.; Liu, F. Discrete Artificial Fish Swarm Algorithm-Based One-Off Optimization Method for Multiple Co-Existing Application Layer Multicast Routing Trees. *Electronics* **2024**, *13*, 894. <https://doi.org/10.3390/electronics13050894>.
2. Duan, Y.; Ni, H.; Zhu, X. Reliable Multicast Based on Congestion-Aware Cache in ICN. *Electronics* **2021**, *10*, 1579. <https://doi.org/10.3390/electronics10131579>.
3. Hu, M.; Xiao, M.; Hu, Y.; Cai, C.; Deng, T.; Peng, K. Software Defined Multicast Using Segment Routing in LEO Satellite Networks. *IEEE Trans. Mob. Comput.* **2022**, *23*, 835–849.
4. Tan, X.; Li, S.; Wang, S.; Liu, Y.; Zheng, Q.; Yang, J. Cooperative Bargaining Game Based Adaptive Video Multicast Over Mobile Edge Networks. *IEEE Trans. Multimed.* **2024**, *26*, 2380–2394.
5. Zhang, D.; Zhao, D.; Ma, H. Robust Load-Balanced Backbone-Based Multicast Routing in Mobile Opportunistic Networks. *Front. Comput. Sci.* **2023**, *17*, 174502.
6. Yang, W.; Chen, Y.; Huang, Z.; Zhang, H.; Gu, H. Routing and Wavelength Assignment for Multiple Multicasts in Optical Network-on-Chip (ONoC). *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2023**, *42*, 4934–4947.
7. Wang, C.-H.; Chiang, S.-H.; Shen, S.-H.; Yang, D.-N.; Chen, W.-T. Multicast Traffic Engineering with Segment Trees in Software-Defined Networks. In Proceedings of the IEEE INFOCOM 2020—IEEE Conference on Computer Communications, Virtual, 6–9 July 2020; pp. 1808–1817.
8. Ford, A.; Raiciu, C.; Handley, M.; Bonaventure, O.; Paasch, C. *TCP Extensions for Multipath Operation with Multiple Addresses*; RFC 8684; Internet Engineering Task Force (IETF): Fremont, CA, USA, 2020. <https://doi.org/10.17487/RFC8684>.
9. Iyengar, J.; Thomson, M. *QUIC: A UDP-Based Multiplexed and Secure Transport*; RFC 9000; Internet Engineering Task Force (IETF): Fremont, CA, USA, 2021. <https://doi.org/10.17487/RFC9000>.
10. Han, B.; Xu, C.; Li, Y.; Wang, X.; Xun, P. MPR-QUIC: Multi-path Partially Reliable Transmission for Priority and Deadline-Aware Video Streaming. *J. Syst. Archit.* **2024**, *153*, 103195.
11. Zheng, Z.; Ma, Y.; Liu, Y.; Yang, F.; Li, Z.; Zhang, Y.; Zhang, J.; Shi, W.; Chen, W.; Li, D.; et al. XLINK: QoE-Driven Multi-Path QUIC Transport in Large-Scale Video Services. In Proceedings of the ACM SIGCOMM 2021 Conference, Virtual Event, 23–27 August 2021; pp. 418–432.
12. Ni, Y.; Zheng, Z.; Lin, X.; Gao, F.; Zeng, X.; Liu, Y.; Xu, T.; Wang, H.; Zhang, Z.; Du, S.; et al. CellFusion: Multipath Vehicle-to-Cloud Video Streaming with Network Coding in the Wild. In Proceedings of the ACM SIGCOMM 2023 Conference, New York, NY, USA, 10–14 September 2023; pp. 668–683.
13. Chiariotti, F.; Zanella, A.; Kucera, S.; Fahmi, K.; Claussen, H. The HOP Protocol: Reliable Latency-Bounded End-to-End Multipath Communication. *IEEE/ACM Trans. Netw.* **2021**, *29*, 2281–2295.
14. Yang, F.; Li, Z.; Zhou, J.; Zhang, X.; Wu, Q.; Pau, G.; Xie, G. Disco: A Framework for Dynamic Selection of Multipath Congestion Control Algorithms. In Proceedings of the IEEE International Conference on Network Protocols (ICNP), Reykjavik, Iceland, 10–13 October 2023; pp. 1–12.

15. Kirmizioglu, R.A.; Tekalp, A.M. Multi-Party WebRTC Services Using Delay and Bandwidth Aware SDN-Assisted IP Multicasting of Scalable Video Over 5G Networks. *IEEE Trans. Multimed.* **2020**, *22*, 1005–1015.
16. Liu, L.; Tan, S.; Guo, S.; Liu, G.; Feng, H. Joint Dynamic VNF Placement and Delay and Jitter Aware Multicast Routing in NFV-Enabled SDNs. *IEEE Trans. Mob. Comput.* **2025**, *24*, 11764–11778.
17. Chiang, S.-H.; Wang, C.-H.; Yang, D.-N.; Liao, W.; Chen, W.-T. Distributed Multicast Traffic Engineering for Multi-Domain Software-Defined Networks. *IEEE Trans. Parallel Distrib. Syst.* **2023**, *34*, 446–462.
18. Ren, H.; Xu, Z.; Liang, W.; Xia, Q.; Zhou, P.; Rana, O.F.; Galis, A.; Wu, G. Efficient Algorithms for Delay-Aware NFV-Enabled Multicasting in Mobile Edge Clouds With Resource Sharing. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *31*, 2050–2066.
19. Xie, M.; Gong, J.; Jia, X.; Ma, X. Age and Energy Tradeoff for Multicast Networks With Short Packet Transmissions. *IEEE Trans. Commun.* **2021**, *69*, 6106–6119.
20. Jin, Z.; Jia, W.-K. DH-SVRF: A Reconfigurable Unicast/Multicast Forwarding for High-Performance Packet Forwarding Engines. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 1262–1275.
21. Rahman, M.M. Blockchain-Assisted QoS-Aware Routing for Software-Defined Wide Area Network. *Electronics* **2025**, *14*, 1949. <https://doi.org/10.3390/electronics14101949>.
22. Li, P.; Wang, H.; Tian, G.; Fan, Z. Towards Sustainable Cloud Computing: Load Balancing with Nature-Inspired Meta-Heuristic Algorithms. *Electronics* **2024**, *13*, 2578. <https://doi.org/10.3390/electronics13132578>.
23. Jayaraman, R.; Manickam, B.; Annamalai, S.; Kumar, M.; Mishra, A.; Shrestha, R. Effective Resource Allocation Technique to Improve QoS in 5G Wireless Network. *Electronics* **2023**, *12*, 451. <https://doi.org/10.3390/electronics12020451>.
24. Tripathi, V.; Talak, R.; Modiano, E. Information Freshness in Multihop Wireless Networks. *IEEE/ACM Trans. Netw.* **2023**, *31*, 784–799.
25. Lyu, X.; Zhang, C.; Ren, C.; Hou, Y. Distributed Graph-Based Optimization of Multicast Data Dissemination for Internet of Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 3117–3128.
26. Araniti, G.; Rinaldi, F.; Scopelliti, P.; Molinaro, A.; Iera, A. A Dynamic MBSFN Area Formation Algorithm for Multicast Service Delivery in 5G NR Networks. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 808–821.
27. Wang, R.-J.; Wang, C.-H.; Yang, D.-N.; Lee, G.-S.; Chen, W.-T.; Sheu, J.-P. Optimizing Resource Block Allocation for Multicast in Beyond 5G Networks. *IEEE Trans. Mob. Comput.* **2025**, *24*, 1–18.
28. Dijkstra, E.W. A Note on Two Problems in Connexion with Graphs. *Numer. Math.* **1959**, *1*, 269–271.
29. Kou, L.; Markowsky, G.; Berman, L. A Fast Algorithm for Steiner Trees. *Acta Inf.* **1981**, *15*, 141–145.
30. Takahashi, H.; Matsuyama, A. An Approximate Solution for the Steiner Problem in Graphs. *Math. Jpn.* **1980**, *24*, 573–577.
31. Mehlhorn, K. A Faster Approximation Algorithm for the Steiner Problem in Graphs. *Inf. Process. Lett.* **1988**, *27*, 125–128.
32. Liu, J.; Chen, G.; Young, E.F.Y. REST: Constructing Rectilinear Steiner Minimum Tree via Reinforcement Learning. In Proceedings of the 2021 58th ACM/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, 5–9 December 2021; pp. 1–6.
33. Li, W.; Zhang, J.; Liu, Y.; Zeng, G.; Wang, Z.; Zeng, C.; Zhou, P.; Wang, Q.; Chen, K. Cepheus: Accelerating Datacenter Applications with High-Performance RoCE-Capable Multicast. In Proceedings of the 2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA), Edinburgh, UK, 2–6 March 2024; pp. 908–921.
34. Wooders, S.; Liu, S.; Jain, P.; Mo, X.; Gonzalez, J.E.; Liu, V.; Stoica, I. Cloudcast: High-Throughput, Cost-Aware Overlay Multicast in the Cloud. In Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation (NSDI), Santa Clara, CA, USA, 16–18 April 2024; pp. 281–296.
35. Xie, J.; Guo, D.; Wu, J.; Ren, B.; Chen, T.; Chen, H. Exploiting Reliable and Scalable Multicast Services in IaaS Datacenters. *IEEE Trans. Serv. Comput.* **2018**, *14*, 1370–1383.
36. Hmila, M.; Fernández-Veiga, M.; Rodríguez-Pérez, M.; Herrería-Alonso, S. Distributed Energy Efficient Channel Allocation in Underlay Multicast D2D Communications. *IEEE Trans. Mob. Comput.* **2020**, *21*, 514–529.
37. Mousavi, M.; Al-Shatri, H.; Klein, A. Cost Sharing Games for Energy-Efficient Multi-Hop Broadcast in Wireless Networks. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 310–324.
38. Cho, I.-S.; Baek, S.J. Optimal Multicast Scheduling for Millimeter Wave Networks Leveraging Directionality and Reflections. In Proceedings of the IEEE INFOCOM 2021—IEEE Conference on Computer Communications, Virtual, 10–13 May 2021; pp. 1–10.
39. Lu, L.; Li, Q.; Zhao, D.; Yang, Y.; Luan, Z.; Zhou, J.; Jiang, Y.; Xu, M. Hawkeye: A Dynamic and Stateless Multicast Mechanism with Deep Reinforcement Learning. In Proceedings of the IEEE INFOCOM 2023—IEEE Conference on Computer Communications, New York, NY, USA, 17–20 May 2023; pp. 1–10.
40. Abolhassani, B.; Tadrous, J.; Eryilmaz, A. Delay Gain Analysis of Wireless Multicasting for Content Distribution. *IEEE/ACM Trans. Netw.* **2020**, *29*, 529–542.

41. Ren, B.; Guo, D.; Shen, Y.; Tang, G.; Lin, X. Embedding Service Function Tree with Minimum Cost for NFV-Enabled Multicast. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1085–1097.
42. Karp, R.M. Reducibility Among Combinatorial Problems. In *50 Years of Integer Programming 1958–2008*; Jünger, M., Liebling, T.M., Naddef, D., Nemhauser, G.L., Pulleyblank, W.R., Reinelt, G., Rinaldi, G., Wolsey, L.A., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 219–241.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.