Peer reviewed Journals

Sci Chronice

RESEARCH

Energy- and Latency-Aware Multicast Routing with Localized Maintenance for Mobile Information Networks

Zubair Khalid,[1] Imran Junaid,[2] and Ariful Islam[3]

[1]Balochistan University of Information Technology, Engineering and Management Sciences, Sariab Road 87, Department of Information Systems Engineering, Quetta, Balochistan, Pakistan
[2]Khushal Khan Khattak University, Karak Bypass Road 22, Department of Software Technology and Analytics, Karak, Khyber Pakhtunkhwa, Pakistan
[3]Dhaka, Bangladesh

Abstract

Mobile information networks increasingly rely on group communication primitives to disseminate sensor updates, situational awareness messages, and content streams to dynamically changing sets of receivers. Multicast routing in these settings must contend with two constraints that often conflict in practice: limited node energy budgets and stringent end-to-end latency requirements. Mobility further complicates the problem by forcing frequent topology changes, which can make globally maintained multicast structures expensive to repair and prone to transient performance degradation. This paper studies energy- and latency-aware multicast routing under mobility with an emphasis on localized maintenance, where repair actions are constrained to a small neighborhood around a disruption to reduce control overhead and preserve data-plane continuity. We formalize a time-varying network model that captures link availability, transmission power adaptation, queuing delay, and receiver dynamics, and we define a multiobjective optimization target that balances energy expenditure against latency while penalizing excessive reconfiguration. Building on this model, we develop a distributed multicast construction procedure that uses locally measurable link metrics and delay-feasible forwarding choices, and we integrate a localized maintenance mechanism that reacts to link failures using bounded-scope rerouting and soft-state stabilization. We provide analytical characterizations of feasibility and cost, including conditions under which localized repair preserves delay constraints and bounds on additional energy induced by mobility. Finally, we discuss evaluation methodology and expected operating regimes, highlighting trade-offs among energy, latency, and maintenance locality in representative mobile information network scenarios.

1. Introduction

Mobile information networks encompass a broad class of systems in which nodes communicate over wireless links while changing relative positions, operating states, and traffic demands [1]. Examples include teams of mobile devices exchanging alerts, vehicular nodes disseminating cooperative perception, autonomous platforms coordinating tasks, and intermittently connected sensors and relays supporting users on the move. In these environments, multicast communication is often a natural abstraction because a single source may need to deliver the same information to multiple receivers with similar timeliness requirements. The engineering challenge is that multicast efficiency depends on the ability to exploit shared forwarding structure, yet mobility degrades the persistence of any such structure and introduces a tension between fast adaptation and low overhead [2].

Energy constraints arise from the limited battery capacity of many participating devices and from the need to prolong network lifetime or mission duration. Even when some nodes can recharge, energy-aware operation remains important because instantaneous power draw can affect thermal limits, duty cycling, and coexistence with other workloads. At the same time, latency constraints are central for interactive and safety-relevant applications in which stale information reduces utility [3]. Latency includes not only propagation time but also medium access delays, retransmission delays, queuing, and processing. Minimizing energy alone can bias routing toward longer, low-power hops or congested relays, whereas minimizing latency alone can bias routing toward high-power transmissions or unstable links, increasing energy and reconfiguration cost.

Multicast routing in mobile settings is traditionally approached through either tree-based structures, which are energy-efficient when stable but costly to maintain, or mesh-based structures, which are robust but can waste energy through redundant forwarding [4]. Localized maintenance occupies an intermediate point by keeping a tree-like structure but confining repairs to small regions when links break or member-

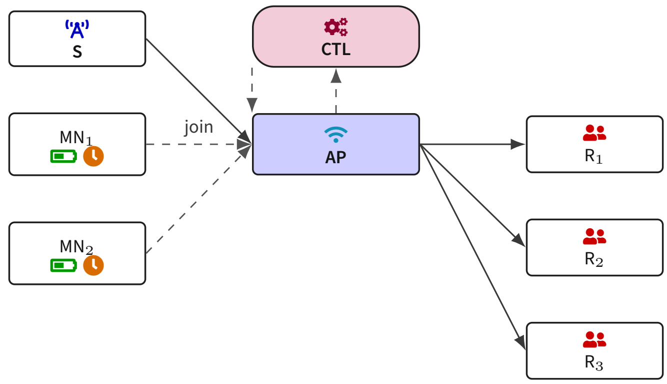

Figure 1. Edge-gateway multicast with energy/latency signals and localized maintenance control.

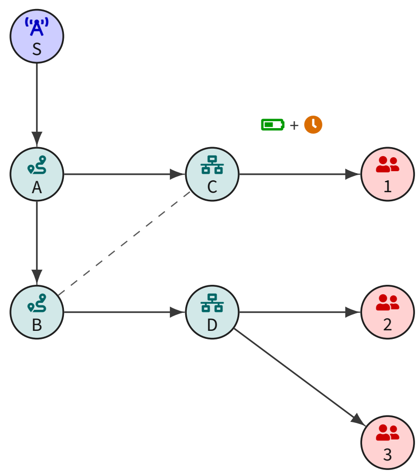

Figure 2. Weighted multicast tree: choose a small set of links that meets delay while favoring higher-energy relays.

ship changes. Localized repair can reduce control-plane overhead and mitigate global disruptions, but it raises nontrivial questions about constraint preservation: a locally optimal patch may violate end-to-end delay limits or shift energy burden onto a subset of nodes, accelerating their depletion.

This paper treats energy- and latency-aware multicast routing as a constrained optimization problem over a time-varying graph and develops a distributed solution framework that emphasizes locality of both construction and maintenance [5]. The contribution is not a claim that any single heuristic dominates in all cases, but a systematic model and design that clarifies which quantities must be tracked locally to balance energy and latency, how delay feasibility can be preserved under bounded-scope repair, and how mobility statistics influence the expected repair rate and overhead. The technical development proceeds by defining link-level metrics that combine transmission energy, expected retransmissions, and delay, and by using those metrics to guide a multicast structure that is delay-feasible by construction. We then design localized maintenance that uses neighborhood-limited detours and stabilization rules to prevent oscillations, while quantifying the additional energy and latency induced by repair operations. Throughout, the design assumes only local measurements and neighbor exchanges rather than global topology knowledge, aligning with operational constraints in mobile information networks [6].

2. Network model and problem formulation

We model the mobile information network over a time horizon as a sequence of directed graphs whose edge sets vary with time due to mobility, wireless channel variation, and node state changes. Let $V(t)$ denote the set of active nodes at time t, and let $E(t) \subseteq V(t) \times V(t)$ denote the set of directed links that are currently feasible under the physical and medium access constraints. For each directed link $(i, j) \in E(t)$, the transmitter i can select a transmit power $p_{ij}(t)$ from an admissible interval determined by hardware limits and regulatory constraints. The received signal-to-interference-plus-noise ratio depends on $p_{ij}(t)$, the pathloss $g_{ij}(t)$, interference conditions, and noise floor, which we abstract into a link success probability $\pi_{ij}(t)$ and an effective service rate $c_{ij}(t)$ in bits per second when scheduled.

Energy consumption is modeled at the granularity of bits transmitted and received, including power am-

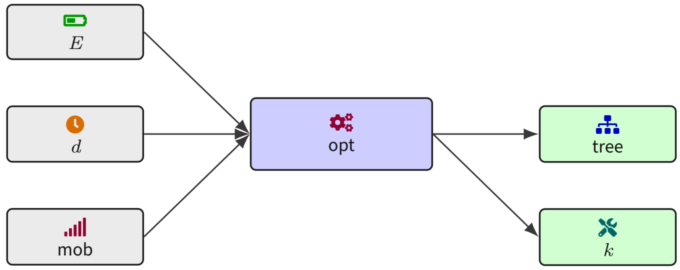

Figure 3. Minimal joint logic: energy, delay, and mobility signals drive tree selection and a local repair radius.

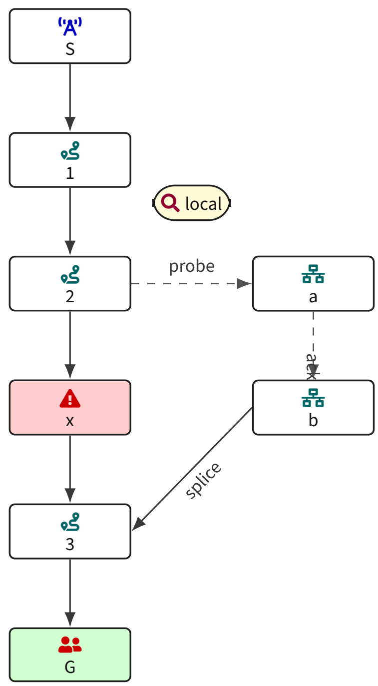

Figure 4. Localized repair: a nearby alternate hop is probed and spliced to restore delivery after a local failure.

plifier energy and circuitry overhead [7]. Let $e^{\mathrm{tx}}_{ij}(t)$ denote the expected energy in joules per delivered information bit on link (i, j) at time t, and let $e^{\mathrm{rx}}_{j}(t)$ denote the receiver-side energy per bit at node j. To capture the impact of retransmissions under unreliable links, we incorporate the expected number of transmission attempts. If a link-layer mechanism repeats until success with independent trials and success probability $\pi_{ij}(t)$, then the expected attempts are $1/\pi_{ij}(t)$ when $\pi_{ij}(t) > 0$. A convenient expression for the expected transmit energy per delivered bit is then

$$e^{\mathrm{tx}}_{ij}(t) = \frac{\alpha_i p_{ij}(t) + \beta_i}{\pi_{ij}(t)\, r_{ij}(t)} \quad \text{and} \quad e_{ij}(t) = e^{\mathrm{tx}}_{ij}(t) + e^{\mathrm{rx}}_{j}(t), \tag{1}$$

where $r_{ij}(t)$ is the physical-layer information rate under the selected modulation and coding, α_i captures amplifier efficiency scaling, and β_i captures per-symbol or per-packet circuitry costs normalized per bit. This abstraction allows energy to increase under higher power, lower rate, or lower reliability, reflecting typical operating behavior without committing to a specific PHY [8].

Latency for a packet traversing a link includes service time and queueing. Let $d_{ij}(t)$ denote the expected per-packet delay on link (i, j) at time t, which can be modeled as

$$d_{ij}(t) = d^{\mathrm{mac}}_{ij}(t) + d^{\mathrm{ser}}_{ij}(t) + d^{\mathrm{q}}_{ij}(t), \tag{2}$$

where $d^{\mathrm{mac}}_{ij}(t)$ captures medium access contention and scheduling delay, $d^{\mathrm{ser}}_{ij}(t)$ captures transmission and possible retransmission time, and $d^{\mathrm{q}}_{ij}(t)$ captures queueing delay. In practice, nodes can estimate $d_{ij}(t)$ using local statistics such as channel access times, measured throughput, and queue occupancy, and can smooth these estimates to reduce volatility.

A multicast session is defined by a source node $s(t) \in V(t)$ and a receiver set $R(t) \subseteq V(t) \setminus \{s(t)\}$ that may change over time due to mobility and application membership. The forwarding structure at time t is represented by a subgraph $H(t) = (V_H(t), E_H(t))$ with

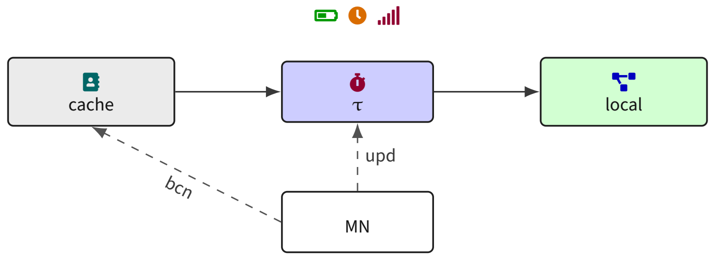

Figure 5. Localized maintenance state: neighbor cache and adaptive timer drive lightweight local decisions under mobility.

Table 1. Network configuration parameters

Parameter	Symbol	Value	Unit
Number of nodes	N	100	–
Simulation area	A	1000×1000	m^2
Radio transmission range	R	250	m
Channel capacity	C	2	Mbps
MAC protocol	–	IEEE 802.11	–
Simulation time	T_{sim}	900	s

$s(t) \in V_H(t)$ and $R(t) \subseteq V_H(t)$, where edges are oriented away from the source. We emphasize tree-like structures for energy efficiency, but allow transient deviations during repair [9]. For each receiver $r \in R(t)$, the end-to-end latency is the path delay from $s(t)$ to r along $H(t)$, denoted $D_{sr}(t)$, and computed as the sum of link delays on the chosen path. A delay requirement is expressed as $D_{sr}(t) \leq \Delta_r(t)$, where $\Delta_r(t)$ is a receiver-specific bound that can encode priority classes.

The optimization objective balances energy, latency, and maintenance cost. Let $f_{ij}(t)$ denote the expected multicast flow rate on edge (i, j) at time t, in bits per second, reflecting the fact that multicast forwarding replicates traffic at branch points. The instantaneous energy rate spent by the forwarding structure is

$$P_E(t) = \sum_{(i,j)\in E_H(t)} e_{ij}(t)\, f_{ij}(t). \tag{3}$$

Latency is constrained rather than minimized directly, but it can also be included as a soft penalty when constraints are slack or infeasible [10]. Maintenance cost reflects control overhead and structural churn. Let $\mathcal{C}(t)$ denote the set of control messages exchanged for construction and repair over a small interval around t, and let $u(t)$ denote a scalar measure of structural change such as the number of parent changes or edge additions and removals per unit time. A generic time-averaged objective over horizon $[0, T]$ can be expressed as

$$J = \frac{1}{T}\int_0^T \Big(P_E(t) + \lambda\, P_M(t)\Big)\, dt, \tag{4}$$

where $P_M(t)$ aggregates maintenance cost and $\lambda \geq 0$ tunes the trade-off. A simple representation is $P_M(t) = \eta_1|\mathcal{C}(t)| + \eta_2 u(t)$ for nonnegative weights η_1, η_2, though the analysis below does not require a specific form as long as $P_M(t)$ grows with message volume and reconfiguration.

The feasible set is constrained by connectivity, delay bounds, and node energy budgets [11]. Node i has residual energy $B_i(t)$, which decreases with transmit and receive work. A safety constraint can be expressed as keeping $B_i(t)$ above an operational threshold or distributing load to reduce imbalance. In mobile information networks, perfect feasibility at all times may be impossible, so an important modeling aspect is the behavior under temporary constraint violation [12]. The design goal becomes to minimize energy subject to delay feasibility whenever feasible, while minimizing the magnitude and duration of violations during disruptions.

3. Energy- and latency-aware multicast construction

Constructing a multicast forwarding structure in a time-varying network resembles a Steiner tree problem with additional delay constraints and time-dependent link costs. Exact solutions are generally intractable and incompatible with decentralized operation [13]. The approach developed here uses a distributed, locally informed construction procedure that targets a delay-feasible tree while implicitly balancing energy and expected stability. The key is to define a link metric that captures energy per delivered bit and a conservative estimate of delay, and then to enforce delay feasibility through bounded path accumulation rather than global optimization.

We define an augmented cost for a directed link

Table 2. Mobility model configuration

Parameter	Symbol	Value	Unit / Notes
Mobility model	–	Random waypoint	–
Min node speed	$v_{\min}$	0	m/s
Max node speed	$v_{\max}$	20	m/s
Pause time	T_p	0–30	s
Node density	ρ	100	nodes/km^2

Table 3. Multicast session configuration

Setting	Value	Description
Number of multicast groups	5–10	Concurrent multicast sessions
Group size	5–30 nodes	Members per multicast group
Sources per group	1–3	Multicast senders per group
Traffic model	CBR/UDP	Constant bit rate over UDP
Packet size	512 bytes	Application payload size
Sending rate	4 pkts/s	Per multicast source

(i, j) at time t: [14]

$$w_{ij}(t) \; = \; e_{ij}(t) + \mu \, d_{ij}(t) \quad \text{with} \quad \mu \geq 0, \tag{5}$$

where μ acts as a conversion factor between delay and energy units. This scalarization does not replace explicit delay bounds; rather, it provides a tie-breaking mechanism among delay-feasible options. A node can compute $w_{ij}(t)$ using local estimates of $e_{ij}(t)$ and $d_{ij}(t)$ derived from measured channel statistics and queueing. Since mobility makes these estimates noisy, the node maintains exponentially weighted averages, and it also tracks a variability indicator, such as the empirical variance of inter-contact time or link success probability [15]. To incorporate stability without overfitting, a mild stability regularizer can be added:

$$\tilde{w}_{ij}(t) \; = \; w_{ij}(t) + \rho \, \sigma_{ij}(t), \tag{6}$$

where $\sigma_{ij}(t)$ is a nonnegative volatility score and $\rho \geq 0$ is small. This term discourages selecting links that appear energy-cheap but are frequently disrupted, reducing repair rates.

The multicast construction proceeds by building receiver paths from the source into a shared structure. Each node maintains a parent pointer toward the source and an accumulated path delay estimate [16]. Let $\hat{D}_i(t)$ denote the estimated delay from $s(t)$ to node i along the currently selected parent pointers, and let $\hat{W}_i(t)$ denote the accumulated scalarized cost. Nodes exchange periodic advertisements containing their current $\hat{D}_i(t)$ and $\hat{W}_i(t)$, along with a freshness timestamp to prevent stale routes from persisting. When a node j hears from a neighbor i that could serve as a parent, it evaluates whether adopting i as its parent would satisfy its delay feasibility and reduce its cost. The update rule is guided by the inequalities

$$\hat{D}_j(t) \leftarrow \hat{D}_i(t) + d_{ij}(t), \tag{7}$$

$$\hat{W}_j(t) \leftarrow \hat{W}_i(t) + \tilde{w}_{ij}(t), \tag{8}$$

subject to $\hat{D}_j(t) \leq \Delta_j(t)$ when j is a receiver or when j is a relay that must keep downstream receivers feasible. Because receivers can appear anywhere, the structure must allow non-receiver nodes to join as Steiner relays [17]. A practical rule is that any node that has at least one downstream member becomes part of the forwarding set and must respect an internal delay budget derived from the tightest downstream requirement. This can be captured by each node maintaining a downstream slack value $\hat{S}_j(t)$, interpreted as the remaining delay margin available for paths passing through j. If j has a downstream receiver r with requirement $\Delta_r(t)$, then j enforces $\hat{D}_j(t) \leq \Delta_r(t) - \hat{D}_{jr}(t)$ where $\hat{D}_{jr}(t)$ is the estimated delay from j to r within the current subtree. In decentralized form, children can report their required slack upward, and the parent can compute the minimum slack across children. This reporting is local and can be piggybacked on periodic keepalive messages to reduce overhead [18].

To avoid excessive branching and to increase sharing, the construction favors attaching receivers to existing tree nodes when possible. When a receiver r joins, it initiates a join solicitation that propagates within a limited radius in terms of delay or hop count. Nodes already in the tree respond with their $\hat{D}_i(t)$ and cost state, enabling r to choose an attachment point that minimizes incremental cost while satisfying $\hat{D}_i(t) + D_{i \to r}(t) \leq \Delta_r(t)$, where $D_{i \to r}(t)$ is the delay of the chosen path from i to r. The path from i to r is computed using the same local cost metric but with a strict delay-feasibility check at each step [19]. Conceptually, the process resembles constrained shortest paths from mul-

Table 4. Energy model parameters for wireless interfaces

Radio state	Notation	Power level	Description
Transmit	P_{tx}	1.4 W	Sending data or control frames
Receive	P_{rx}	1.0 W	Receiving data or control frames
Idle	P_{idle}	0.83 W	Radio on, no packets processed
Sleep	P_{sleep}	0.013 W	Low-power standby mode
Battery cap.	E_{bat}	1000 J	Initial energy per node

Table 5. Localized maintenance and repair parameters

Parameter	Symbol	Value	Description
Local repair radius	H_{repair}	2 hops	Max hops for localized repair
Max repair attempts	N_{retry}	3	Retries before route error
Repair timeout	T_{repair}	200 ms	Wait time for local fix
Link failure threshold	N_{fail}	2	Missed HELLOs before failure
Cache entry lifetime	T_{cache}	30 s	Validity of local route cache

tiple candidate attachment points, but implemented by localized message exchanges.

A central difficulty in delay-constrained multicast is that delay feasibility is not purely additive when queueing depends on load, and load depends on the multicast tree structure. The model addresses this by treating $d_{ij}(t)$ as a measured quantity that already reflects current contention and queueing, making feasibility a function of present conditions rather than a priori capacity planning. However, measured delays can change after the tree is built because added traffic increases contention [20]. To mitigate this feedback loop, nodes include an admission control mechanism based on incremental load estimates. If the source transmits at rate $\gamma(t)$ and a relay forwards to k children, then its outgoing load is approximately $k\gamma(t)$ when no coding is used. Nodes can estimate the marginal increase in queueing delay under an additional child using local queue models. A simple approximation uses an $M/M/1$-style sensitivity: [21]

$$d^{q}_{ij}(t) \approx \frac{1}{c_{ij}(t) - \ell_{ij}(t)} \quad \text{when} \quad \ell_{ij}(t) < c_{ij}(t), \tag{9}$$

where $\ell_{ij}(t)$ is the current offered load on the link in bits per second. When a new child would increase $\ell_{ij}(t)$ beyond a safety fraction of $c_{ij}(t)$, the relay declines the attachment or offers an alternative neighbor. While the $M/M/1$ approximation is not exact for wireless contention, it provides a locally computable guardrail that reduces severe queueing spikes.

The resulting multicast structure is a directed arborescence rooted at $s(t)$ under stable conditions, with each node having a single parent. The tree is energy-aware because $\tilde{w}_{ij}(t)$ includes expected energy per delivered bit and retransmissions, and it is latency-aware because candidates are screened by delay feasibility with measured delay components. The design is compatible with localized information because all quantities required for parent selection are available from neighbor advertisements and local measurements [22]. The next section addresses how this structure is maintained under mobility, when edges in $E(t)$ disappear or degrade and when receivers join or leave.

4. Localized maintenance under mobility

Mobility induces link breakage, fluctuating quality, and intermittent connectivity. A multicast tree built at one instant may become invalid shortly thereafter, causing packet loss and delay violations if repairs are not performed [23]. Global reconstructions can restore optimality but typically incur significant control overhead, transient loops, and convergence delays, particularly when the network is large or when mobility is frequent. Localized maintenance aims to repair the structure near the disruption while preserving most of the tree, thereby reducing both overhead and instability. The core question is how to design local repair rules that are sufficiently conservative to preserve delay constraints and sufficiently agile to react to topology changes [24].

We model disruptions as events that invalidate one or more parent-child edges in the current forwarding structure. Let (u, v) be an edge in the tree at time t^- that becomes unusable at time t^+ due to mobility or quality degradation. Node v, which relied on u as its parent, detects the failure through missing keepalives, repeated transmission failures, or explicit link-layer notifications [25]. Upon detection, v enters a repair state in which it must quickly find an alternative parent to reconnect to the source while keeping downstream receivers feasible. Localized repair restricts the search to a bounded neighborhood, defined either by hop radius $h_{\max}$ or by a delay budget $\delta_{\max}$. This bound is chosen so that the repair overhead remains small and the repaired path does not significantly inflate end-to-end

Table 6. Decomposition of end-to-end latency

Component	Symbol	Expression	Description
Queuing delay	D_q	–	Time in interface queues
MAC access delay	D_{mac}	–	Backoff and channel access
Transmission delay	D_t	L/C	Frame length over data rate
Propagation delay	D_p	d/v	Distance over signal speed
Route discovery delay	D_r	–	Path setup / repair time

Table 7. Performance metrics used for evaluation

Metric	Symbol	Definition	Unit
Packet delivery ratio	PDR	$\frac{\text{pkts received}}{\text{pkts sent}}$	%
Avg. end-to-end delay	$\overline{D}$	Mean latency over delivered packets	ms
Energy per delivered pkt	E_d	Total energy / delivered packets	J/packet
Normalized control overhead	NCO	Control bytes / data bytes	–
Average tree depth	H	Mean hop count from sources to group members	hops

delay.

A repair mechanism needs a local view of feasible attachment points. Nodes in the vicinity of v can advertise whether they are currently connected to the source within acceptable delay, along with their $\hat{D}_i(t)$ and a remaining slack estimate. Node v then explores candidate parents i among its neighbors and possibly neighbors-of-neighbors, using a limited-scope query that carries the required slack for v's subtree [26]. If v has downstream receivers with tight requirements, it computes a required parent condition of the form $\hat{D}_i(t) + d_{iv}(t) + \hat{D}_{v\to r}(t) \le \Delta_r(t)$ for all receivers r in its subtree, where $\hat{D}_{v\to r}(t)$ is the estimated delay from v to r within the subtree. In practice, enforcing this for all downstream receivers is achieved by tracking only the tightest requirement, because satisfying the minimum slack across receivers suffices. Let $\Delta_v^{\min}(t)$ denote the minimum remaining budget for the subtree rooted at v, computed as

$$\Delta_v^{\min}(t) \;=\; \min_{r\in\mathcal{R}_v(t)} \Big(\Delta_r(t) - \hat{D}_{v\to r}(t)\Big), \qquad (10)$$

where $\mathcal{R}_v(t)$ is the set of receivers in v's subtree. Then v needs a parent i such that $\hat{D}_i(t) + d_{iv}(t) \le \Delta_v^{\min}(t)$. This scalar constraint is locally checkable given neighbor advertisements of $\hat{D}_i(t)$ and measured $d_{iv}(t)$.

Localized maintenance also must address the risk of oscillation, where nodes repeatedly change parents as conditions fluctuate, causing churn and wasted energy. To stabilize, the repair logic introduces hysteresis and soft-state timers [27]. Each node maintains a minimum residence time with a parent unless a hard failure occurs, and it changes parent on soft degradations only when the cost improvement exceeds a threshold. If a link remains usable but its delay increases, the node evaluates whether the delay slack remains nonnegative. If slack is still sufficient, the node may defer switching to avoid churn [28]. If slack becomes negative, the node must switch promptly. This behavior can be expressed as a rule in which a node computes a margin

$$m_v(t) \;=\; \Delta_v^{\min}(t) - \Big(\hat{D}_{\text{parent}(v)}(t) + d_{\text{parent}(v),v}(t)\Big), \qquad (11)$$

and triggers proactive repair when $m_v(t)$ falls below a small safety buffer, for example when $m_v(t)$ approaches zero [29]. The buffer is not a fixed constant in all deployments; it may be adapted to observed delay variability, becoming larger when delays fluctuate widely.

Repair locality interacts with energy because local detours can increase hop count, and local repair may attach to a nearby node with higher energy cost compared to a globally optimal alternative. To bound this effect, the maintenance mechanism prioritizes candidate parents that minimize incremental energy subject to delay feasibility, using the same link cost $\tilde{w}_{ij}(t)$ as in construction. However, to prevent concentrating load on a single stable relay, the mechanism also incorporates an energy-aware fairness term that depends on residual energy [30]. Let $B_i(t)$ be residual energy at node i. A relay with low $B_i(t)$ should be less likely to attract additional children. This can be captured by an adjusted parent-selection cost

$$\bar{w}_{iv}(t) \;=\; \tilde{w}_{iv}(t) + \kappa\,\phi\big(B_i(t)\big), \qquad (12)$$

where $\phi(\cdot)$ is a nonincreasing function such as $\phi(B) = 1/(B+\epsilon)$ and $\kappa \ge 0$ tunes the bias [31]. The use of ϕ does not guarantee max-min fairness, but it locally discourages overuse of low-energy relays.

Localized repair can still create temporary forwarding gaps. To reduce packet loss during repair, the mechanism allows a short-lived buffering and alternative forwarding mode at the subtree root experiencing disruption [32]. Node v can temporarily forward packets via any available neighbor that claims connectivity

Table 8. Multicast routing protocols considered

Protocol	Type	Maintenance style	Brief description
ELMR	Tree-based	Localized repair	Energy- and latency-aware multicast routing
MAODV	Tree-based	Global repair	Multicast extension of AODV
ODMRP	Mesh-based	Periodic refresh	On-demand multicast routing protocol
CAMP	Mesh-based	Core-based	Core-assisted multicast protocol

Table 9. Qualitative comparison of protocol behavior

Protocol	Energy cost	End-to-end delay	Control overhead
ELMR	Low	Low	Medium
MAODV	Medium	Medium	Medium–High
ODMRP	High	Low	High
CAMP	Medium	Medium	Medium

to the source, even if the path is not yet fully validated for optimality, as long as a strict delay feasibility check passes. Once a stable parent is chosen, forwarding converges back to tree mode. This transient behavior can be controlled by a time-to-live on repair forwarding state, ensuring that the network does not remain in an unstable hybrid mode for long durations [33].

Mobility also affects receiver membership. When a receiver leaves, its parent can prune that branch if no other receivers remain downstream, reducing energy consumption. Pruning is localized and uses a simple rule: a node that is not a receiver and has no downstream members stops forwarding and informs its parent [34]. Conversely, when a receiver joins, it attaches through the localized join procedure described earlier. These join and leave operations can be handled with the same soft-state approach as link maintenance: membership is refreshed periodically, and absence of refresh triggers timeout-based removal. Soft-state reduces the need for reliable control messaging, which is beneficial under mobility [35].

An important aspect of localized maintenance is the scope selection. If the scope is too small, repair may fail even when a feasible global route exists. If the scope is too large, repair becomes similar to global reconstruction, losing the advantages of locality. The design thus adapts scope based on observed conditions [36]. When node density is high and links are short-lived, a small scope may suffice because alternative parents are nearby. When density is low or mobility is high, the scope may expand temporarily upon repair failure, but still remain bounded to avoid flooding. A bounded expanding ring search in terms of delay budget can be implemented without explicit enumeration by progressively relaxing the acceptable parent delay bound while keeping message propagation limited by a hop cap [37]. The analysis section below clarifies how scope and mobility statistics influence expected repair success.

5. Analytical characterization and optimization structure

The distributed procedures described above rely on local metrics and bounded repair, so a natural analytical goal is to understand when delay feasibility is preserved, how energy cost compares to a baseline, and how mobility affects maintenance overhead. The results here are characterizations rather than universal guarantees, because they depend on stochastic processes governing mobility and wireless performance [38]. Nonetheless, it is possible to derive bounds and expected-value relations that inform parameter choices.

Consider a snapshot in which the network graph is connected enough to support at least one delay-feasible path from the source to each receiver. Let $D_r^*(t)$ be the minimum achievable delay from $s(t)$ to receiver r at time t under the measured per-link delays $d_{ij}(t)$. Delay feasibility requires $D_r^*(t) \leq \Delta_r(t)$ [39]. The construction procedure selects a parent tree that ensures each receiver's path satisfies the bound under current measurements. Under perfect measurement and no subsequent load change, this is immediate. Under load feedback, feasibility can be violated if the additional multicast traffic increases $d_{ij}(t)$ after attachment. A conservative sufficient condition for feasibility is that each link's delay estimate includes an upper confidence bound that accounts for worst-case additional contention induced by one more child [40]. If $\hat{d}_{ij}(t)$ denotes the measured delay and $\delta_{ij}^{\text{inc}}(t)$ is an upper bound on delay increase under an incremental load $\Delta\ell_{ij}$, then using

$$d_{ij}^{\text{cons}}(t) = \hat{d}_{ij}(t) + \delta_{ij}^{\text{inc}}(t) \tag{13}$$

in the feasibility checks yields a margin against post-attachment increase. The tightness of this bound depends on how $\delta_{ij}^{\text{inc}}(t)$ is computed. Under a simplified service model where the effective capacity $c_{ij}(t)$ remains constant over the short term and queueing dominates variability, one can relate delay increase to

Table 10. Impact of multicast group size on performance trends

Group size	Avg. path length	Delivery ratio trend	Energy per node
5 members	Short	High	Low
10 members	Moderate	High	Low–Medium
20 members	Long	Medium	Medium
30 members	Longer	Medium–Low	High
40 members	Longest	Low	Highest

utilization. Using the approximation $d^{q}_{ij}(t) \approx 1/(c_{ij}(t) - \ell_{ij}(t))$, the marginal delay sensitivity is

$$\frac{\partial d^{q}_{ij}(t)}{\partial \ell_{ij}(t)} \approx \frac{1}{\big(c_{ij}(t) - \ell_{ij}(t)\big)^2}, \tag{14}$$

so an upper bound for a small increment $\Delta\ell_{ij}$ is $\delta^{\mathrm{inc}}_{ij}(t) \approx \Delta\ell_{ij}/(c_{ij}(t) - \ell_{ij}(t))^2$ when $c_{ij}(t) > \ell_{ij}(t)$. This does not fully capture wireless contention coupling across links, but it provides a tractable local bound that scales rapidly as utilization approaches capacity, which is consistent with empirical observations in many MAC regimes.

For localized repair, the key feasibility question is whether a subtree root v can find an alternative parent within its repair scope that preserves the delay constraints for all downstream receivers. Let $\mathcal{N}_h(v,t)$ denote the set of nodes within h hops of v at time t in the underlying connectivity graph. Suppose repair searches for a parent in $\mathcal{N}_h(v,t)$ that is already connected to the source with advertised delay $\hat{D}_i(t)$. A sufficient condition for repair success is existence of some $i \in \mathcal{N}_h(v,t)$ such that

$$\hat{D}_i(t) + D^{\min}_{i \to v}(t) \;\leq\; \Delta^{\min}_v(t), \tag{15}$$

where $D^{\min}_{i \to v}(t)$ is the minimum delay from i to v within the local subgraph induced by the scope. Because repair uses local exploration rather than exact minimization, success further depends on whether the exploration finds such a path [41]. If the exploration is a constrained shortest-path search that expands neighbors in order of increasing accumulated delay and prunes when accumulated delay exceeds $\Delta^{\min}_v(t) - \hat{D}_i(t)$, then it will find a feasible path whenever one exists within the scope and the link delays used are conservative. Thus, feasibility reduces largely to a geometric and stochastic question: how often does such an i exist within the chosen scope given mobility and density.

A common way to model node positions is as a spatial point process at each time, with temporal correlation induced by mobility. While a full stochastic geometry treatment may be complex, an instructive approximation is to consider that at the moment of repair, potential attachment points are distributed with an effective density $\lambda_{\mathrm{eff}}(t)$ representing the density of nodes that are both within scope and currently connected to the source with sufficient slack. If the repair region is approximated as an area A_h (for a hop-bounded scope under roughly uniform neighbor distance), then the probability that at least one candidate exists is approximately [42]

$$P_{\mathrm{cand}} \;\approx\; 1 - \exp\big(-\lambda_{\mathrm{eff}}(t)\, A_h\big). \tag{16}$$

This expression highlights that localized repair is more reliable when there are many connected nodes nearby and when the effective region size is large. However, increasing scope increases control overhead and may increase delay. The design therefore benefits from adjusting h based on $\lambda_{\mathrm{eff}}(t)$, which can be inferred from local neighbor counts and observed success rates.

Energy overhead due to maintenance can be decomposed into control-plane energy and data-plane inflation [43]. Control-plane energy depends on the number of messages and their sizes, and data-plane inflation depends on the additional hops or higher-cost links introduced by repair compared to a globally optimal reconstruction. Let E_{ctrl} denote the energy per control message transmission and reception aggregated across the local neighborhood, and let $M(t)$ be the message count per unit time. Then the control energy rate is approximately $P_{\mathrm{ctrl}}(t) \approx E_{\mathrm{ctrl}} M(t)$. Data-plane inflation can be expressed by comparing the total energy rate of the maintained structure $P_E(t)$ to a hypothetical baseline $P^{\mathrm{glob}}_E(t)$ that would be obtained by instantaneous global reconstruction minimizing energy subject to delay. Define the inflation ratio

$$\Gamma_E(t) \;=\; \frac{P_E(t)}{P^{\mathrm{glob}}_E(t)}. \tag{17}$$

Localized maintenance tends to keep $\Gamma_E(t)$ close to 1 when disruptions are sparse and when alternative parents exist nearby with similar link costs [44]. In contrast, when connectivity is sparse, local repair may create longer detours, increasing $\Gamma_E(t)$. One can bound $\Gamma_E(t)$ in terms of the maximum relative stretch of the chosen local detour compared to the best feasible detour. If the local search finds a path from some attachment point i to v with accumulated cost at most σ times the minimum possible cost within the scope, and if the scope contains a globally optimal attachment, then the repair cost is within σ of the best [45].

In practice, σ is influenced by the search strategy; expanding candidates by increasing $\tilde{w}$ under a delay constraint tends to yield small empirical stretch, though this is scenario-dependent.

Mobility determines how often repairs are triggered. Let T_{ij} be the random lifetime of link (i,j) while it is used in the forwarding tree, and let Λ be the aggregate failure rate across all tree edges. If link failures are approximately independent and memoryless over short intervals, a simplifying assumption is exponential link lifetimes with parameter ν_{ij}, so $\mathbb{E}[T_{ij}] = 1/\nu_{ij}$. Then

$$\Lambda(t) \;\approx \sum_{(i,j)\in E_H(t)} \nu_{ij}(t), \qquad (18)$$

and the expected number of repairs per unit time is proportional to $\Lambda(t)$. Although exponential lifetimes are not always accurate, the linear aggregation emphasizes that larger trees and less stable links increase repair frequency. The stability regularizer $\sigma_{ij}(t)$ in the construction metric can reduce $\Lambda(t)$ by biasing away from volatile links, at the expense of potentially higher energy or delay.

Latency impact under maintenance is similarly decomposed into baseline path delay and repair-induced transient delay [46]. Let $D_{sr}(t)$ be the steady-state delay along the maintained tree and let $\tau_r(t)$ be the time receiver r experiences disruption during repair. The effective application-level latency may include buffering and retransmission delays during $\tau_r(t)$. A rough characterization of the expected additional delay due to disruptions is $\mathbb{E}[\Delta D_r] \approx \mathbb{E}[\tau_r] \cdot \gamma$ under a constant rate model, where γ is the source packet rate in packets per second, because disruption causes packets to queue or be resent. Reducing $\mathbb{E}[\tau_r]$ is thus important for latency-sensitive applications, and localized repair aims to reduce τ_r by limiting scope and accelerating convergence. However, too small a scope may increase τ_r if repair fails and must expand, so the optimal scope is not necessarily minimal.

These relations indicate a structured tuning problem [47]. Parameters μ, ρ, κ, and the repair scope bounds trade energy, latency slack, fairness, and maintenance cost. A pragmatic optimization view is to minimize a weighted sum of estimated steady-state energy and expected repair overhead subject to a target probability of delay feasibility. While an exact solution is not necessary for operation, the model clarifies that the system should adapt to observed mobility and contention, increasing stability bias and repair scope when failure rates rise, and relaxing them when the network is stable to improve energy efficiency [48].

6. Evaluation methodology and practical considerations

A credible evaluation of energy- and latency-aware multicast with localized maintenance must reflect the coupled nature of wireless contention, mobility, and control overhead. Because the design relies on measured link delay and reliability, the evaluation should include a medium access model that captures contention and retransmissions, rather than assuming fixed link delays. At the same time, evaluation must explore a range of mobility regimes and node densities to determine when localized maintenance is beneficial and when its limitations emerge [49].

A representative evaluation framework models nodes moving in a bounded area with reflective or wrap-around boundaries to avoid density artifacts. Mobility can be parameterized by average speed and pause time, but the choice of mobility model influences link lifetime distributions. Random waypoint mobility tends to produce central clustering, whereas group mobility produces correlated movement that can stabilize certain links while destabilizing inter-group links [50]. For mobile information networks such as cooperative vehicular systems, mobility constrained to road graphs produces different contact patterns, often with directional flows and intermittent connectivity. The maintenance algorithm's performance should be tested across multiple mobility models because the volatility score $\sigma_{ij}(t)$ and the link lifetime parameters $\nu_{ij}(t)$ depend strongly on mobility correlation. In all cases, it is useful to calibrate models so that the network experiences a range of link durations, from long-lived to short-lived, because localized repair behavior changes qualitatively across these regimes.

Traffic models should include a multicast source rate $\gamma(t)$ and packet sizes consistent with the application. For situational awareness, packets may be small but frequent, whereas for content dissemination they may be larger or bursty [51]. Since queueing delay depends on offered load, the evaluation should vary $\gamma(t)$ and include background unicast traffic that competes for the medium. Without background traffic, the delay component may be dominated by retransmissions, and the benefits of delay-aware admission control may not be visible. With background traffic, queueing and MAC contention become major contributors, and the delay feasibility mechanism becomes more meaningful [52]. An important practical detail is whether multicast is implemented as replicated unicast at the MAC layer or as true wireless multicast. Many commodity systems treat multicast as unreliable broadcast without acknowledgments, which changes the reliability and energy model. The framework here is compatible with either approach by adjusting $\pi_{ij}(t)$ and retransmission assumptions, but evaluation should make the chosen model explicit.

Metrics should be computed at both network and receiver levels [53]. Energy consumption is most informative when separated into transmit, receive, idle listening, and control components, because localized maintenance can reduce control energy but may increase data forwarding energy due to detours. Latency should be measured as end-to-end delay distribution, not only average, because applications often care about tail behavior. Delay feasibility can be reported as the fraction of delivered packets that meet receiver bounds, which is sensitive to disruptions and repair time [54]. Structural churn can be measured by parent changes per unit time and by the number of edges added or removed during repairs. Control overhead can be measured in bytes and in channel time consumed by control packets, because energy and contention effects depend on airtime.

A typical comparison set includes a baseline energy-minimizing multicast tree without delay awareness, a delay-minimizing tree without energy awareness, a mesh-based approach that maintains multiple parents or redundant paths, and a global-rebuild approach that recomputes the tree periodically [55]. In an energy-only baseline, one often observes low energy under stable conditions but frequent delay violations when the tree routes through congested relays or long hop chains. In a delay-only baseline, one often observes improved latency but increased energy due to higher power choices and reduced sharing, and also potentially increased churn because the minimum-delay routes may involve volatile links. A mesh-based approach can reduce disruption impact because alternate paths exist, but it may increase energy due to redundant forwarding and can increase contention, which may paradoxically worsen latency under high load. Periodic global rebuild can keep the structure near-optimal at rebuild times but may cause oscillations and high control overhead, and it introduces a rebuild period that trades responsiveness against overhead [56].

When evaluating localized maintenance, it is important to attribute improvements or regressions to specific mechanisms. The use of conservative delay estimates and admission control typically increases the success rate of meeting delay bounds, but it may reduce sharing if relays decline new children under load, leading to higher overall energy. The fairness-adjusted parent cost can spread load and delay node depletion, but it may increase hop count and thus increase energy per delivered bit [57]. The stability regularizer can reduce repair frequency, but if it biases too strongly away from cheap links, it may increase energy in low-mobility regimes where repairs would have been rare anyway. These effects should be explored by varying ρ and κ across regimes.

Localized repair scope is another central factor [58]. With a small hop cap, repairs are fast and overhead is low, but repair failure probability can rise in sparse networks. With a larger hop cap, repair is more likely to succeed without escalation, but overhead rises and repairs may select longer detours that consume more energy and reduce latency slack. A nuanced evaluation varies scope and reports not only average overhead but also the distribution of repair times and the fraction of repairs that required scope expansion [59]. A useful way to summarize is to report the empirical relationship between observed neighbor density and required scope for a target repair success probability, which can motivate adaptive scope selection based on local neighbor counts.

Practical implementation requires attention to measurement noise and clocking. Delay estimates $d_{ij}(t)$ derived from queueing and MAC contention can fluctuate rapidly, and if parent selection reacts too strongly, churn can increase. The hysteresis mechanism and residence times mitigate this, but the chosen timescales matter [60]. If advertisements are too frequent, control overhead increases and contention grows. If they are too infrequent, repair detection slows and stale routes persist. A reasonable operational design uses a base advertisement interval with random jitter to avoid synchronization and an event-driven fast advertisement when a hard failure is detected. Soft-state timeouts should be set relative to the advertisement interval and expected link lifetimes; timeouts that are too short cause false failure detections under temporary interference, while timeouts that are too long prolong disruptions [61].

Security and trust are also relevant because localized maintenance relies on neighbor advertisements of delay and connectivity. In adversarial environments, nodes could misreport metrics to attract traffic or disrupt routing. While a full security design is beyond scope, practical deployments can include sanity checks such as validating advertised delays against observed forwarding performance and limiting the influence of any single neighbor through bounded trust [62]. In non-adversarial but heterogeneous environments, nodes may have different capabilities, and metric normalization matters. A device with higher processing capacity may sustain lower queueing delay and thus attract more children, which can be acceptable if it has sufficient energy or power supply, but problematic if it is battery-limited. The fairness term based on residual energy partially addresses this by discouraging overuse of low-energy nodes [63].

Another practical consideration is interaction with application-layer reliability. If the application requires reliable delivery, then packet losses during repair must be recovered. Local buffering at subtree roots and selective retransmissions can reduce effective loss, but they may increase latency [64]. The design should therefore align with the application's reliability-latency trade-off. In some mobile information networks, it is acceptable to drop outdated updates, in which case the repair mechanism can prioritize quick reconnection over perfect reliability. In others, such as mission-critical logs, reliability is more important, and repair might involve temporary redundant forwarding [65]. The framework can accommodate this by allowing a short-lived redundant mode during repair, but evaluation should reflect the specific reliability semantics.

Finally, it is useful to interpret performance through the lens of operating regimes. In dense networks with moderate mobility, localized maintenance tends to find nearby alternative parents quickly, keeping disruptions short and overhead low, and the energy cost is often close to a globally rebuilt tree because local alternatives are plentiful. In sparse networks or under very high mobility, the neighborhood may not contain feasible attachment points, causing repair escalation and potentially longer detours, and the benefits of localization may diminish [66]. Under high offered load, the delay component becomes sensitive to contention, making conservative delay estimation and admission control more important, but also making sharing harder because relays saturate quickly. Under low load, energy dominates and the algorithm may behave similarly to energy-aware routing with occasional repairs. These qualitative patterns do not guarantee specific numerical outcomes, but they clarify what factors most strongly influence performance and which parameters should adapt online [67].

7. Conclusion

This paper developed a model and design framework for energy- and latency-aware multicast routing with localized maintenance in mobile information networks. The core problem is balancing energy expenditure against end-to-end delay constraints under time-varying connectivity, while limiting the control overhead and structural churn that mobility induces. The proposed approach uses locally measurable link metrics that combine expected energy per delivered bit, retransmission effects, and measured delay, and it enforces delay feasibility through conservative path accumulation and slack propagation [68]. Localized maintenance repairs disruptions by searching for alternative attachment points within bounded scope, using subtree delay budgets to preserve downstream feasibility, and employing hysteresis and soft-state stabilization to reduce oscillations. Analytical characterizations related repair success probability and overhead to effective candidate density, link lifetime processes, and scope selection, and they clarified how stability bias and fairness adjustments trade energy for reduced repair rate and better load distribution.

The overall picture is that localized maintenance can reduce global disruptions and control cost when the local neighborhood often contains feasible alternatives, but its effectiveness depends on mobility, density, and offered load [69]. Conservative delay estimation and admission control can improve feasibility under contention but may reduce sharing and increase energy. Scope adaptation and stability regularization can reduce disruption time and churn but may increase path cost. These trade-offs suggest that practical deployments benefit from lightweight online adaptation driven by observed neighbor availability, repair success rates, and delay variability, rather than fixed parameters. The model and mechanisms presented here provide a basis for such adaptation while remaining compatible with decentralized operation and the measurement realities of mobile wireless systems [70].

References

[1] A. Bartoli, G. Dini, and L. Lopriore, “Single address space implementation in distributed systems,” Concurrency: Practice and Experience, vol. 12, pp. 251–280, 4 2000.

[2] C.-C. Chiang and R. Y. Lee, “Coordination languages and models for open distributed systems,” International Journal of Software Innovation, vol. 1, pp. 1–13, 1 2013.

[3] F. Giraldeau and M. Dagenais, “Wait analysis of distributed systems using kernel tracing,” IEEE Transactions on Parallel and Distributed Systems, vol. 27, pp. 2450–2461, 8 2016.

[4] A. Languasco, A Partially Decentralised Protocol for a Distributed Encrypted Storage System, pp. 195–204. Springer Nature Switzerland, 11 2023.

[5] R. Malik, S. Kim, X. Jin, C. Ramachandran, J. Han, I. Gupta, and K. Nahrstedt, “Mlr-index: An index structure for fast and scalable similarity search in high dimensions,” in International Conference on Scientific and Statistical Database Management, pp. 167–184, Springer, 2009.

[6] B. Huang, Y. Liu, Z. Xia, and J. Wang, “A bi-event-triggered multi-agent system for distributed optimization,” IEEE Transactions on Network Science and Engineering, vol. 10, pp. 1074–1084, 3 2023.

[7] T. Xu, Y. Ye, and Y. Tang, “Experimental evaluation of isac in distributed and centralized multiuser-mimo systems,” in 2024 IEEE Conference on Standards for Communications and Networking (CSCN), pp. 340–345, IEEE, 11 2024.

[8] Z. Lv, K. Huang, Y. Wang, R. Tao, G. Wu, J. Zhang, and Y. Tian, Distributed Differential Privacy Protection System for Personalized Recommendation, pp. 211–222. Germany: Springer Singapore, 8 2020.

[9] H. A. Thompson, P. J. Fleming, and A. J. Chipperfield, “Multi-objective optimisation of systems architectures for distributed aero-engine control systems,” in 34th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, American Institute of Aeronautics and Astronautics, 7 1998.

[10] X. Lin, W. Shi, and Y. Du, “Interactive data encryption strategy for distributed simulation system,” International Journal of Modeling, Simulation, and Scientific Computing, vol. 04, pp. 1342005–, 9 2013.

[11] J. Liu and W. Chen, “Distributed convex optimisation with event-triggered communication in networked systems,” International Journal of Systems Science, vol. 47, pp. 3876–3887, 1 2016.

[12] J. Saia, “Session details: Privacy and distributed systems,” in Proceedings of the 10th ACM international workshop on Foundations of mobile computing, ACM, 8 2014.

[13] T. Srinivasan, R. Chandrasekar, V. Vijaykumar, V. Mahadevan, A. Meyyappan, and A. Manikandan, “Localized tree change multicast protocol for mobile ad hoc networks,” in 2006 International Conference on Wireless and Mobile Communications (ICWMC’06), pp. 44–44, IEEE, 2006.

[14] A. J. Helmicki, C. A. Jacobson, and C. N. Nett, “Control-oriented modeling of distributed parameter systems,” Journal of Dynamic Systems, Measurement, and Control, vol. 114, pp. 339–346, 9 1992.

[15] Z. Li, J. Shen, J. Ren, and Y. Huang, “Modeling method of multi-dimensional feature distributed system,” in 2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT), pp. 1066–1072, IEEE, 10 2021.

[16] H. Huang and S. Yang, ISCTCS - A Method of Evaluating Distributed Storage System Reliability. Germany: Springer Berlin Heidelberg, 6 2015.

[17] W. Yu, G. Wen, G. Chen, and J. Cao, Distributed Cooperative Control of Multi-agent Systems - Consensus in Multi-agent Systems. Wiley, 9 2016.

[18] B. Yang, “Distributed transfer function analysis of complex distributed parameter systems,” Journal of Applied Mechanics, vol. 61, pp. 84–92, 3 1994.

[19] Z. Zhong and X. Chen, “Software design of distributed automatic testing system for uav flight control system,” in SPIE Proceedings, vol. 5253, pp. 598–601, SPIE, 9 2003.

[20] B. Yuan, D. Jankov, J. Zou, Y. Tang, D. C. Bourgeois, and C. Jermaine, “Tensor relational algebra for distributed machine learning system design,” Proceedings of the VLDB Endowment, vol. 14, pp. 1338–1350, 10 2021.

[21] R. Chandrasekar, R. Suresh, and S. Ponnambalam, “Evaluating an obstacle avoidance strategy to ant colony optimization algorithm for classification in event logs,” in 2006 International Conference on Advanced Computing and Communications, pp. 628–629, IEEE, 2006.

[22] S. Valle, N. Prabakar, and H. Upadhyay, “Building a distributed system for live virtual machine introspection,” in EPiC Series in Computing, vol. 89, pp. 72–62, EasyChair, 11 2022.

[23] G. Liu, T. Jiang, T. B. Ollis, X. Li, F. Li, and K. Tomsovic, “Resilient distribution system leveraging distributed generation and microgrids: a review,” IET Energy Systems Integration, vol. 2, pp. 289–304, 12 2020.

[24] S. Berger and R. Henning, “Communication system cues to benefit distributed action teams,” Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 63, pp. 904–908, 11 2019.

[25] H. C. Khatri and R. E. Goodson, “Optimal feedback solutions for a class of distributed systems,” Journal of Basic Engineering, vol. 88, pp. 337–342, 6 1966.

[26] J. C. Knight and J. I. A. Urquhart, “Fault tolerant distributed systems using ada,” in 4th Computers in Aerospace Conference, American Institute of Aeronautics and Astronautics, 8 1983.

[27] P. S. Curtiss and J. F. Kreider, “Recent developments in the control of distributed electrical generation systems,” Journal of Solar Energy Engineering, vol. 125, pp. 352–358, 8 2003.

[28] S. Castano, A. Ferrara, and S. Montanelli, Dynamic Knowledge Discovery in Open, Distributed and Multi-Ontology Systems. IGI Global, 1 2011.

[29] H. Wang, A. Papachristodoulou, and K. Margellos, “Distributed safety verification for multi-agent systems,” in 2023 62nd IEEE Conference on Decision and Control (CDC), pp. 5481–5486, IEEE, 12 2023.

[30] D. T. Dunham, S. S. Blackman, and R. J. Dempster, “Multiple hypothesis tracking for a distributed multiple platform system,” in SPIE Proceedings, vol. 5428, pp. 180–191, SPIE, 8 2004.

[31] L. M. Shekhtman and E. Waisbard, “Engravechain: A blockchain-based tamper-proof distributed log system,” Future Internet, vol. 13, pp. 143–, 5 2021.

[32] A. J. Muñoz-Vázquez, G. Fernández-Anaya, J. D. Sanchez-Torres, and F. Meléndez-Vázquez, “Predefined-time control of distributed-order systems,” Nonlinear Dynamics, vol. 103, pp. 2689–2700, 2 2021.

[33] I. Raicu, I. Foster, and P. Beckman, “Making a case for distributed file systems at exascale,” in Proceedings of the third international workshop on Large-scale system and application performance, pp. 11–18, ACM, 6 2011.

[34] R. Walker and P. Nilsson, “Experience of using the chirp distributed file system in atlas,” Journal of Physics: Conference Series, vol. 396, pp. 052073–, 12 2012.

[35] O. T. Saa, “Some mathematical aspects of dag-based distributed ledger systems,” 7 2020.

[36] R. Chandrasekar and T. Srinivasan, “An improved probabilistic ant based clustering for distributed databases,” in Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI, pp. 2701–2706, 2007.

[37] L. Chen and Y. Guo, “Design of the distributed control system based on can bus,” Computer and Information Science, vol. 4, pp. 83–89, 6 2011.

[38] M. Li, Y. Chen, C. Liu, D. Wang, and H. Qin, “Construction and application of distributed power storage system,” E3S Web of Conferences, vol. 369, pp. 2002–02002, 2 2023.

[39] C. Dajun, Q. Zhangquan, Z. Qi, and L. Li, “Spark distributed system and gpu parallel computing,” in International Conference on Electronic Information Engineering and Computer Science (EIECS 2022), pp. 64–64, SPIE, 4 2023.

[40] B. S. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, “Osdi - an integrated experimental environment for distributed systems and networks,” ACM SIGOPS Operating Systems Review, vol. 36, pp. 255–270, 12 2002.

[41] F. Endress, T. Kipouros, and M. Zimmermann, “Distributing design domains for topology optimization in systems design,” in Volume 2: 43rd Computers and Information in Engineering Conference (CIE), American Society of Mechanical Engineers, 8 2023.

[42] P. Fonseca, K. Zhang, X. Wang, and A. Krishnamurthy, “Eurosys - an empirical study on the correctness of formally verified distributed systems,” in Proceedings of the Twelfth European Conference on Computer Systems, pp. 328–343, ACM, 4 2017.

[43] Q. Zhu, S. Fei, T. Li, and T. Zhang, “Robust adaptive control for uncertain systems with discrete and distributed delays,” Journal of Control Theory and Applications, vol. 6, pp. 287–292, 9 2008.

[44] Y. N. Kyrychko and I. B. Schwartz, “Enhancing noise-induced switching times in systems with distributed delays.,” Chaos (Woodbury, N.Y.), vol. 28, pp. 063106–063106, 6 2018.

[45] N. Arshad, D. Heimbigner, and A. L. Wolf, “Dealing with failures during failure recovery of distributed systems,” ACM SIGSOFT Software Engineering Notes, vol. 30, pp. 1–6, 5 2005.

[46] G. D. Fatta and G. Fortino, “Sac - a customizable multi-agent system for distributed data mining,” in Proceedings of the 2007 ACM symposium on Applied computing, pp. 42–47, ACM, 3 2007.

[47] J. Liu and C. Zhang, “Distributed learning systems with first-order methods,” Foundations and Trends in Databases, vol. 9, pp. 1–100, 6 2020.

[48] A. Squicciarini, G. Petracca, and E. Bertino, “Codaspy - adaptive data protection in distributed systems,” in Proceedings of the third ACM conference on Data and application security and privacy, pp. 365–376, ACM, 2 2013.

[49] D. Wang, F. Xie, Y. Lu, T. Ji, C. Du, and Y. Zheng, “Bioinspired dynamic soaring simulation system with distributed pressure sensors.,” Bioinspiration & biomimetics, vol. 17, pp. 36010–036010, 4 2022.

[50] E. G. Amparore and S. Donatelli, Quantitative Assessments of Distributed Systems: Methodologies and Techniques -

Stochastic Path Properties of Distributed Systems: The CSLTA Approach. Wiley, 4 2015.

[51] C. Ramachandran, S. Misra, and M. Obaidat, "On evaluating some agent-based intrusion detection schemes in mobile ad-hoc networks," Proceedings of the SPECTS 2007, pp. 594–601, 2007.

[52] H. Lin, Agent-Based Architecture of a Distributed Laboratory System. IGI Global, 1 2011.

[53] J. Tufarolo, J. Nielsen, S. Symington, R. Weatherly, A. Wilson, J. Ivers, and T. Hyon, "Automated distributed system testing: designing an rti verification system," in WSC'99. 1999 Winter Simulation Conference Proceedings. 'Simulation - A Bridge to the Future' (Cat. No.99CH37038), vol. 2, pp. 1094–1102, IEEE, 1 2003.

[54] Z. Zhou, Y. Wu, J. Cheng, and Y. Wang, "Distributed filtering for markovian jump systems with packet loss compensation," International Journal of Control, Automation and Systems, vol. 21, pp. 2154–2161, 5 2023.

[55] Z. Gao, L. Shi, and P. Reviriego, "Enhancing data protection with a distributed storage system based on the redundant residue number system," Wireless Networks, vol. 30, pp. 5601–5612, 3 2023.

[56] Y. Yang and X. Wei, "Efficient distributed file system offloading on smartnic," in 2024 IEEE International Conference on High Performance Computing and Communications (HPCC), pp. 34–41, IEEE, 12 2024.

[57] J. Brumfield, "Performance modeling of distributed object-oriented database systems," in Proceedings [1988] International Symposium on Databases in Parallel and Distributed Systems, pp. 22–32, IEEE, 8 2005.

[58] S. M. Koehler, F. Chuang, Y. Ma, A. Daly, and F. Borrelli, Distributed Model Predictive Control for Forced-Air Systems, pp. 167–189. Springer International Publishing, 12 2017.

[59] M. Taube, G. Losa, K. L. McMillan, O. Padon, M. Sagiv, S. Shoham, J. R. Wilcox, and D. Woos, "Modularity for decidability of deductive verification with applications to distributed systems," ACM SIGPLAN Notices, vol. 53, pp. 662–677, 6 2018.

[60] Q. Xue, J. Wang, and J. Sun, "Research on distributed virtual reality system in electronic commerce," in SPIE Proceedings, vol. 5444, pp. 229–232, SPIE, 3 2004.

[61] J. Chen, H. Zhuang, and M. Yang, A Distributed Vehicle-Infrastructure Cooperation System Based on OpenHarmony, pp. 258–271. Germany: Springer Nature Singapore, 2 2024.

[62] H. F. Rashvand and J. M. A. Calero, Distributed Sensor Systems: Practice and Applications - Monitoring Well Being. Wiley, 2 2012.

[63] A. Dubey, R. Mehrotra, S. Abdelwahed, and A. N. Tantawi, "Performance modeling of distributed multi-tier enterprise systems," ACM SIGMETRICS Performance Evaluation Review, vol. 37, pp. 9–11, 10 2009.

[64] A. Macchelli, "Distributed-parameter port-hamiltonian systems in discrete-time," in 2023 62nd IEEE Conference on Decision and Control (CDC), vol. 223, pp. 2931–2936, IEEE, 12 2023.

[65] P. Wei and H.-X. Li, "Multiscale fusion for abnormality detection and localization of distributed parameter systems," 12 2023.

[66] N. Jaber, C. Wagner, S. Jacobs, M. Kulkarni, and R. Samanta, "Quicksilver: modeling and parameterized verification for distributed agreement-based systems," Proceedings of the ACM on Programming Languages, vol. 5, pp. 1–31, 10 2021.

[67] R. Malik, C. Ramachandran, I. Gupta, and K. Nahrstedt, "Samera: a scalable and memory-efficient feature extraction algorithm for short 3d video segments.," in IMMERSCOM, p. 18, 2009.

[68] M. Paz and Y. H. Kim, Dynamic Analysis of Systems with Distributed Properties, pp. 415–436. United States: Springer International Publishing, 11 2018.

[69] W. He, F. Qian, G. Chen, and Q.-L. Han, Distributed Impulsive Control of Leader-Following Multi-agent Systems, pp. 1–54. Springer Nature Singapore, 4 2019.

[70] M. Ally, Intelligent Tutoring Systems for Distributed Learning. IGI Global, 1 2011.