Highlights

In-band Network Telemetry Orchestration: Models, Advances, and Challenges

Yonghao Zhang,Lizhuang Tan,Yuyu Zhao,Nguyen Van Tu,Pilar Manzanares-Lopez,Na Li,Huiling Shi,Wei Zhang,Peiying Zhang,Wei Su,James Won-Ki Hong

- This is a comprehensive survey introducing in-band network telemetry orchestration (INTO) technology.
- This survey casts INTO methods as optimization models, offering a unified lens for analyzing orchestration mechanisms.
- This survey proposes a taxonomy that categorizes existing INTO approaches into active, passive, and hybrid modes, and delineates their respective implementation pathways.
- The survey outlines critical challenges and six future directions, guiding the evolution of next-generation network telemetry.

In-band Network Telemetry Orchestration: Models, Advances, and Challenges

Yonghao Zhang[a,b], Lizhuang Tan[a,b,*], Yuyu Zhao[c], Nguyen Van Tu[d], Pilar Manzanares-Lopez[e], Na Li[f,g], Huiling Shi[a,b], Wei Zhang[a,b], Peiying Zhang[a,h], Wei Su[i] and James Won-Ki Hong[j]

[a]*Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Ji'nan 250014, P. R. China*

[b]*Shandong Provincial Key Laboratory of Computing Power Internet and Service Computing, Shandong Fundamental Research Center for Computer Science, Ji'nan 250014, P. R. China*

[c]*School of Cyber Science and Engineering, Southeast University, Nanjing 211189, P. R. China*

[d]*MangoBoost Inc., Seoul 08806, Korea*

[e]*Department of Information Technologies and Communications, Universidad Politécnica de Cartagena, Cartagena 30202, Spain*

[f]*School of Cyber Science and Technology, Shandong University, Qingdao 266237, P. R. China*

[g]*Shandong Branch of National Computer network Emergency Response technical Team/Coordination Center (CNCERT/SD), Ji'nan 250002, P. R. China*

[h]*Qingdao Institute of Software, College of Computer Science and Technology, China University of Petroleum (East China), Qingdao 266580, P. R. China*

[i]*School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing 100044, P. R. China*

[j]*Department of Computer Science and Engineering, Pohang University of Science and Technology, Pohang 37673, Korea*

ARTICLE INFO

Keywords:
Network Management
Network Measurement
Network Telemetry
In-band Network Telemetry
In-band Network Telemetry Orchestration

ABSTRACT

As network scale and complexity escalate, fine-grained monitoring has become critical for operation, administration, and maintenance. In-band network telemetry (INT) offers a leading solution for end-to-end visibility by embedding device states directly into packets. However, its widespread deployment faces a fundamental challenge, the inherent conflict between comprehensive visibility and limited network resources. To address this, the orchestration of INT is crucial. In this paper, we define in-band network telemetry orchestration (INTO) as a control mechanism that performs the unified modeling, planning, and scheduling of telemetry tasks to determine their triggering modes, coverage, and execution paths under resource constraints. Building on this definition, we provide a comprehensive and structured survey of INTO. With the introduction of a system-level constraint abstraction and a structured taxonomy, we present a thorough analysis of the current INTO research. Specifically, we analyze active INTO from both task-driven and method-based perspectives, and examine passive INTO focusing on flow selection and data reduction mechanisms. Furthermore, we conceptualize hybrid INTO as a joint optimization problem that coordinates multi-modal telemetry. Finally, we outline open challenges and future directions to guide the evolution of network telemetry.

1. Introduction

With the emergence and development of software-defined networking (SDN) [1] and programmable data planes (PDPs) [2], network measurement is evolving from being host and control plane-driven to being data plane-driven. SDN enables the decoupling and re-architecting of the control and data planes, making network measurement more flexible. Meanwhile, PDPs enhances the performance of switches in handling network measurements. In this context, network telemetry is recognized as a superior alternative, offering enhanced accuracy, scalability, and performance. At the forefront of this approach is in-band network telemetry (INT) [3]. Originally driven by the capabilities of PDPs and now formally standardized by the P4.org applications working group [4], INT represents a fundamental shift in network monitoring. Unlike legacy approaches that rely on control plane intervention, INT empowers network devices to execute customized packet-processing logic directly in the data plane [5], enabling fine-grained state collection at line rate. Specifically, INT-capable devices such as programmable switches process telemetry instructions within the forwarding pipeline [6] and, as packets traverse the network, each INT-enabled device sequentially collects and appends its local network state to the packet's metadata stack. In operational practice, INT can be realized in two common approaches. In the passive approach, existing user flows are utilized to piggyback telemetry metadata, generating little to no additional telemetry flows, but this approach is constrained by the inherent paths of user flows. In contrast, the active approach injects dedicated probe packets to collect network state information. Although this approach introduces additional overhead, it provides greater flexibility and controllability, enabling on-demand probing of selected network paths. Overall, INT solutions offer accurate, real-time, hop-by-hop measurements that significantly enhance network operation, administration, and maintenance (OAM) by enabling fine-grained network visualization.

However, deploying INT at network scale requires more than adding telemetry capabilities to the data plane, mandates a control-plane mechanism that decides when and where

*Corresponding author

10431250094@stu.qlu.edu.cn (Y. Zhang); tanlzh@sdas.org (L. Tan); yyzhao@seu.edu.cn (Y. Zhao); nguyen.tu@mangoboost.io (N.V. Tu); pilar.manzanares@upct.es (P. Manzanares-Lopez); lina@cert.org.cn (N. Li); shihl@sdas.org (H. Shi); wzhang@sdas.org (W. Zhang); zhangpeiying@upc.edu.cn (P. Zhang); wsu@bjtu.edu.cn (W. Su); jwkhong@postech.ac.kr (J.W. Hong)

ORCID(s): 0000-0001-6826-4596 (L. Tan)

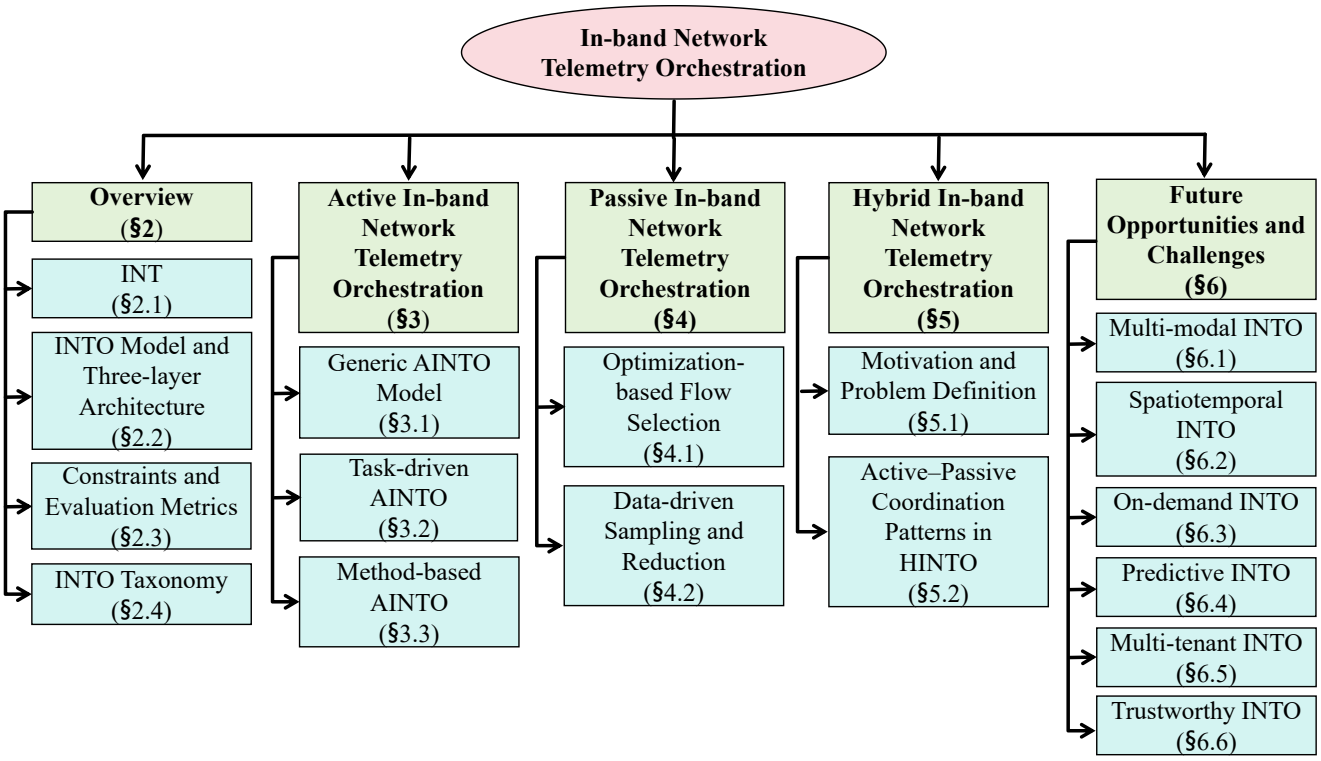


Figure 1: The structure of this survey.

telemetry is triggered, what is collected, and how it is scheduled and reported under stringent resource and operational constraints. We define this approach as in-band network telemetry orchestration (INTO), which is a control mechanism that unifies the modeling, planning, scheduling, and coordination of telemetry tasks. Specifically, INTO determines triggering strategies, coverage scopes, collection granularities, execution paths, and dynamic adaptation policies, thereby explicitly governing the trade-offs among efficiency, accuracy, and scalability. In this survey, INTO is defined and discussed primarily with respect to INT-based orchestration, while some of the underlying orchestration issues may also be relevant, at a conceptual level, to other in-packet telemetry approaches such as IOAM. Although there are several research reviews focused on INT [7], to the best of our knowledge, there is still a lack of comprehensive surveys focused on orchestration challenges and solutions. This work addresses this gap from both a system-level and an algorithmic perspective. The main contributions of this survey are summarized as follows.

- We define INTO and establish a unified system model, and abstract practical constraints into a generic constraint set $\mathcal{C} = \{\mathcal{C}_{\mathrm{MTU}}, \mathcal{C}_{\mathrm{BW}}, \mathcal{C}_{\mathrm{HW}}, \mathcal{C}_{\mathrm{AoI}}\}$, capturing packet-size headroom, bandwidth budget, device resource bounds, and telemetry freshness requirements for orchestration analysis.
- We first classify INTO into three modes: Active INTO (AINTO), Passive INTO (PINTO) and Hybrid INTO (HINTO). For AINTO, we model probe-path planning and scheduling as a generic optimization problem, which subsumes coverage-aware, performance-aware, overhead-optimized, and application-aware schemes as specific instantiations. For PINTO, we abstract passive orchestration as selecting a subset of user flows F and an in-packet sampling configuration S under the same system constraints, unifying optimization-based flow selection and data-driven sampling mechanisms within a common opportunity-management perspective. We further combine active and passive orchestration into a HINTO model with joint decision variables, providing a unified framework for analyzing coordination and cross-modal trade-offs. This taxonomy-driven abstraction facilitates a structured comparison of the three families.
- We identify open challenges and outline future research directions for INTO, including (i) multi-modal INTO, (ii) spatiotemporal INTO, (iii) on-demand INTO, (iv) predictive INTO, (v) multi-tenant INTO, and (vi) trustworthy INTO.

The organizational structure of this survey is illustrated in Fig. 1. Specifically, the remainder of the paper is organized as follows. In Section 2, we provide an overview of INT and INTO, key constraints, evaluation metrics, and the taxonomy adopted in this survey. In Section 3, we review AINTO, presenting a generic AINTO and offering a structured synthesis of representative approaches from both task-driven and method-based perspectives. In Section 4, we survey PINTO, covering optimization-based flow selection

and data-driven sampling and reduction, and summarizes the main insights and limitations of existing PINTO designs. In Section 5, we analyze hybrid INTO by formulating its problem setting and discussing typical active–passive coordination patterns in HINTO systems. In Section 6, we outline future opportunities and challenges for telemetry orchestration, including multi-modal, spatiotemporal, on-demand, predictive, multi-tenant isolation and security, and trustworthy directions.

2. Overview

This section provides a comprehensive overview of the technical foundations and orchestration requirements of INT. We first introduce the framework of INT, detailing device roles and the two operational modes. Next, we present the model of INTO, summarize the key constraints and evaluation metrics, and conclude by outlining the taxonomy of INTO that guides the subsequent discussion.

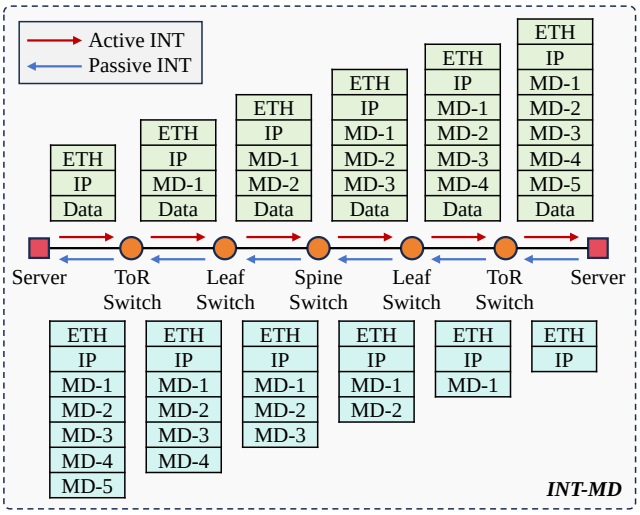
2.1. INT

In-band network telemetry (INT) [3] is a programmable network monitoring framework that collects and reports network state directly from the data plane, without requiring explicit intervention from the control plane. Unlike traditional polling or sampling mechanisms, INT enables network devices to execute customized telemetry instructions at line rate, providing nanosecond-level timing accuracy and fine-grained, flow-level visibility. Within a typical INT domain, devices are logically divided into three roles: the INT Source embeds telemetry instructions into packets; the INT Transit nodes execute these instructions and insert locally measured metadata; and the INT Sink extracts the accumulated metadata and exports it to a monitoring or control system for further processing.

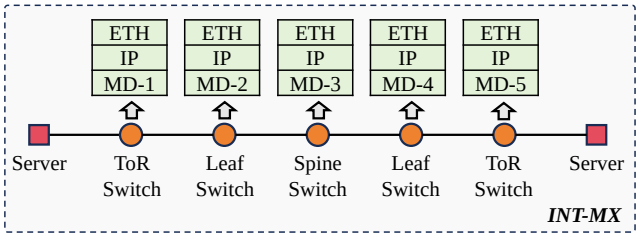
From an operational viewpoint, current INT deployments mainly adopt two modes, as illustrated in Fig. 2 (a) and Fig. 2 (b).

- **INT-MD:** Fig. 2 (a) depicts the classic Metadata-in-Data mode [8]. In INT-MD, both telemetry instructions and collected metadata are written directly into the packet header. As the packet traverses the network, each transit node pushes its locally measured state onto a metadata stack carried by the packet, causing the packet size to increase hop by hop. This design allows the sink to obtain a complete hop-by-hop trace for each instrumented packet directly from the packet header. However the cumulative header growth makes INT-MD particularly sensitive to maximum transmission unit (MTU) constraints.

- **INT-MX:** Fig. 2 (b) illustrates the Metadata Export mode [9]. In INT-MX, the packet header contains only telemetry instructions, so the packet size remains constant or incurs negligible overhead along the path. Transit nodes follow the in-band instructions to collect locally measured state, but instead of embedding metadata into the packet, they export it in separate telemetry reports to a monitoring server. This decoupling prevents MTU violations but introduces additional processing and correlation overhead at the telemetry server, which must reconstruct end-to-end paths from fragmented reports generated by multiple devices.



(a) INT-MD



(b) INT-MX

Figure 2: Schematic view of the two primary INT modes: INT-MD and INT-MX.

Regardless of whether telemetry metadata are embedded in packets or exported as separate reports, INT aims to provide timely and accurate network visibility for the control/management plane for operation, administration, and maintenance. To this end, INT-capable devices collect a set of basic telemetry metrics at multiple granularities, from device-wide states to per-packet metadata. As summarized in Table 1, we categorize these metrics by their measurement scope, including device, interface, queue, and packet.

Beyond these basic metrics, representative systems have shown that INT can support a broad suite of application-layer modules in realistic networks, as summarized in Fig. 3. Figure 3 links these application modules to the telemetry layer and the underlying data-layer interfaces, and highlights where INTO operates to bridge application needs and telemetry configurations. These modules range from general network measurement [10] and SLA verification [11], to anomaly detection [12] and micro-burst detection [13], as well as failure detection [14]. Concretely, INT has been used to measure one-way delay [15], tail latency [16], available bandwidth [17], queue depth [18], flow statistics [19, 20], switch

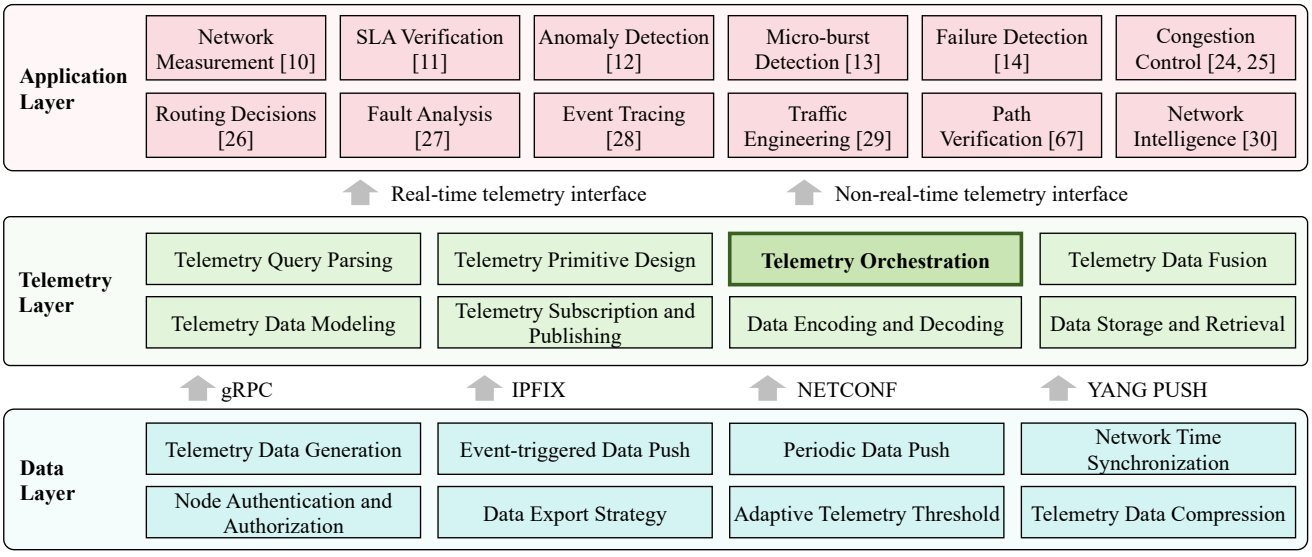


Figure 3: INT system stack and modules.

Table 1
Taxonomy of INT Statistics by Granularity

Granularity Layer	Telemetry Metric
Device & Flow Table	Switch ID
	Flow Table Capacity
	Table Lookup/Match Count
	Timestamp
Port / Interface	Port ID
	Link Utilization
	Packet/Byte Count
	Error Count
	Link Up/Down Status
Queue & Buffer	Queue ID
	Queue Depth / Occupancy
	Drop Count
Packet & Flow	Sequence Number
	Hop Latency
	Flow Age
	Inter-Arrival Time
	5-Tuple

processing delay [21], QoS compliance [22], and SFC performance [23].

Beyond pure measurement, INT has also been integrated into control and troubleshooting workflows, including congestion control [24, 25], routing decisions [26], post-failure diagnosis and recovery [27], event tracing [28], traffic engineering [29], and learning-assisted network intelligence (NetworkAI) [30]. Taken together, these experiences confirm that INT is a practical substrate for both high-resolution visibility and telemetry-driven control.

2.2. INTO

As shown in Section 2.1, INT provides rich data plane capabilities: operators can choose different operational modes and collect multi-granularity telemetry metrics. The remaining challenge is to orchestrate these capabilities at network scale under practical resource and operational constraints, especially in modern multi-application networks where heterogeneous services demand different monitoring targets, granularities, and timeliness. To meet such demands, the control plane must coordinate coupled decisions on what to instrument, what to collect, and how to schedule collection and reporting, so that application intents can be translated into feasible, efficient, and dynamically adjustable telemetry configurations [31].

Building on this, INTO provides the mechanism to translate intents into telemetry configurations. Operators and network applications act as telemetry consumers. They specify what to monitor, at what temporal granularity, and with what overhead and timeliness requirements. INTO translates these requirements into executable telemetry configurations under limited packet headroom, bandwidth, and device resources. INT-capable devices then execute the configuration and deliver per-hop states via INT-MD or INT-MX to the collector, which feeds results back to update orchestration decisions.

2.3. Constraints and Evaluation Metrics

The system model in Section 2.2 highlights that the INTO module acts as the decision-making bridge between application requirements and the underlying INT/IOAM data plane. Unlike an idealized monitoring system, practical orchestration cannot allocate telemetry tasks arbitrarily. Instead, it must identify feasible solutions within a bounded region defined by a tuple of system-level constraints $C = \{C_{\mathrm{MTU}}, C_{\mathrm{BW}}, C_{\mathrm{HW}}, C_{\mathrm{AoI}}\}$, which are imposed by packet formats, network capacity, device pipelines, and timeliness requirements. For readability, Table 2 summarizes the common notations used in this paper.

Table 2
Key Notations Used in This Survey

Notation	Meaning
$G = (V, E)$	Directed network graph, V is the node set and E is the directed link set.
$\mathcal{P} = \{p_1, \ldots, p_K\}$	Selected probe-path set.
p	A probe path.
Edges(p)	The set of links traversed by path p.
E_{target}	Target link set that must be covered.
$J(P)$	Weighted orchestration objective over the selected path set.
$b_p,\ \ell_p,\ r_p$	Bandwidth, latency, and rule-installation cost components for path p.
$w_b,\ w_\ell,\ w_r$	Nonnegative weights balancing b_p, ℓ_p, and r_p in $J(P)$.
$C_{MTU}(P)$	MTU feasibility predicate for a path set P.
$C_{BW}(P)$	Bandwidth feasibility constraint for the path set P.
$C_{HW}(P)$	Hardware feasibility constraint for the path set P.
$C_{AoI}(P)$	Freshness feasibility constraint for the path set P.
$\lvert p\rvert,\ H_{\max}$	Hop count of a probe path and an upper hop limit.

2.3.1. Problem Formulation and Constraints

Packet-size (C_{MTU}). The MTU imposes a strict upper bound on packet length along a forwarding path. Let p denote a forwarding path, let L denote the original payload size, and let L_0 denote the fixed INT header size. If ℓ_k is the metadata size inserted by the k-th device on p, the instrumented packet length must satisfy:

$$L + L_0 + \sum_{k \in p} \ell_k \le \min_{e \in p} \mu_e, \tag{1}$$

where μ_e is the MTU of link e on path p. This constraint implies that orchestration must control either the path length or the total metadata volume to prevent fragmentation.

Bandwidth consumption (C_{BW}). Telemetry traffic shares link capacity with user traffic, and excessive probe injection or per-packet metadata may saturate bottleneck links. Consider a link e with physical capacity C_e. Let $\mathcal{F}_e$ be the set of flows traversing e. For each flow $f \in \mathcal{F}_e$, let $s_f \in [0, 1]$ denote the sampling ratio, let λ_f denote the estimated average packet rate over a control interval, and let δ denote the average telemetry overhead per sampled packet. The average telemetry bandwidth induced on link e over the same control interval is bounded by:

$$\sum_{f \in \mathcal{F}_e} \left(s_f \cdot \lambda_f \cdot \delta\right) \le \alpha \left(C_e - B_e^u\right), \tag{2}$$

where B_e^u is the average bandwidth already consumed by non-telemetry traffic on e over that control interval, and $\alpha \in (0, 1)$ reserves headroom to reduce telemetry-induced congestion. In practical deployments, Equation (2) is more appropriately interpreted as a planning-level or epoch-level feasibility condition, rather than as implying continuous real-time shaping of telemetry traffic according to instantaneous residual bandwidth in the data plane. Existing studies typically approximate such a constraint through budgeted headroom assumptions, coarse-grained sampling or rate configurations, and event-driven or other adaptive reconfiguration mechanisms [32, 33], so that telemetry overhead remains bounded over a control interval. Although recent designs have improved online responsiveness through faster or more adaptive decision mechanisms [34, 35], these adaptations still operate at a timescale substantially coarser than microsecond-level traffic bursts. Accordingly, Equation (2) serves to characterize average or budgeted telemetry overhead under practical operating assumptions, whereas burst-level regulation remains difficult to guarantee on current programmable switches.

Hardware resource constraints (C_{HW}). PDPs impose rigid resource limits, especially in the packet header vector (PHV) [36] and on-chip memory. Let h denote the baseline header width that must be parsed and processed, and let ℓ_{int} denote the additional width introduced by INT metadata. The telemetry configuration must satisfy

$$h + \ell_{\text{int}} \le \phi, \tag{3}$$

where ϕ is the PHV capacity of the target device. Equation (3) characterizes the header-width feasibility of telemetry processing, but it captures only one aspect of the hardware feasibility condition C_{HW}. In practical P4-programmable switches, telemetry deployment is further constrained by the available SRAM/TCAM resources for match-action tables, which limit the number and complexity of telemetry rules that can be realized, as well as by pipeline-stage and ALU dependency constraints, which govern the placement of telemetry parsing, matching, state update, and reporting logic along the forwarding pipeline. Accordingly, C_{HW} should be interpreted more broadly as the joint feasibility of PHV capacity, table memory capacity, and pipeline-stage execution. In addition, pipeline execution is constrained by per-stage instruction budgets, stage-local resource availability, and operation dependencies, motivating sparse and selective activation of telemetry logic in practice.

Timeliness and Age of Information (C_{AoI}). For closed-loop control, telemetry must remain fresh at the controller. We define the age of information (AoI) [37] for a telemetry report as

$$\Delta = \tau_a - \tau_g, \tag{4}$$

where τ_g is the generation time at the data plane and τ_a is the time when the report becomes available for analysis at the sink or controller. We model the report age by the end-to-end delivery latency along the reporting path and the processing latency at the sink,

$$\widehat{\Delta} = \sum_{e \in P} \left(d_e + q_e\right) + \rho, \tag{5}$$

where d_e and q_e denote propagation and queuing delays on link e, and ρ denotes parsing and analytics delay at the sink.

Table 3
Key evaluation indicators, quantitative optimization metrics, and orchestration decision dimensions for INTO.

Category	Metric	Definition	Goal
Evaluation metrics	Telemetry accuracy	Closeness of the reported values to the actual network state.	MAX
	Telemetry granularity	Spatial and temporal resolution of telemetry data.	MAX
	Telemetry freshness	Delay between network events and their observation by applications.	MIN
	Telemetry intrusion	Negative impact of telemetry on bandwidth usage and forwarding performance.	MIN
	Telemetry scope	Fraction of network elements covered by telemetry.	MAX
Optimizatic metrics	Telemetry overlap	Degree to which nodes or links are repeatedly monitored by multiple probe paths.	MIN
	Telemetry path length	Hop count or end-to-end delay of probe paths; highly unbalanced lengths create bottlenecks.	MIN
	Telemetry probe set size	Number of probe flows instantiated by orchestration.	MIN
	Telemetry per-packet metadata overhead	Telemetry bytes added per instrumented packet.	MIN
	Telemetry bandwidth overhead	Extra bandwidth consumed by probe packets under a budget.	MIN

Freshness is satisfied when the realized age stays within a validity budget θ, which encourages INTO to jointly control measurement coverage and end-to-end reporting latency.

Taken together, the above constraints define the feasible deployment region of INTO on real programmable data planes. Rather than being merely abstract terms in the orchestration model, they delimit how telemetry can be deployed under packet headroom, hardware-resource, and timeliness constraints. In practice, these deployment boundaries favor path segmentation, selective telemetry activation, compact encoding, and control-interval-based resource management, so that mathematically feasible telemetry strategies often still require simplified probe formats, sparse activation, or scenario-specific tailoring to remain deployable on real P4 targets.

2.3.2. Optimization Metrics

Under these constraints, the goal of INTO is not simply to maximize the amount of collected data, but to optimize a set of key performance indicators that capture the balance between measurement quality and operational cost. Table 3 summarizes the main evaluation metrics used throughout this survey, together with representative optimization metrics that are commonly used in the literature.

2.4. INTO Taxonomy

Building upon the system model and constraints discussed above, the fundamental design decision for an INTO system is the selection of the telemetry flows. At a higher level, this corresponds to choosing the measurement carrier flows, i.e. whether telemetry is obtained via dedicated probe flows, existing user flows, or a combination of both. This dimension aligns with the classical taxonomy defined in RFC 7799 [38], which categorizes traditional network measurements based on whether they rely on injected probes or piggyback on user flows. Consequently, strictly adhering to the nature of the carrier, we classify existing INTO mechanisms into three primary families: active INTO (AINTO), passive INTO (PINTO), and hybrid INTO (HINTO).

AINTO injects dedicated probe packets or instantiates dedicated measurement flows that are steered along selected paths to collect telemetry. The orchestration plane explicitly chooses probe sources, destinations, candidate paths, and sending rates, adapting these decisions to monitoring objectives and network conditions under practical bandwidth, packet-size, and device-resource constraints. As a result, AINTO offers high flexibility and fine-grained control over where and when measurements are taken. However, it introduces additional flows and processing overhead and must be engineered to avoid interfering with user flows. Moreover, probe flows may not always be fully representative of user flows, which can affect measurement fidelity. Typical design questions include probe path planning, rate control and scheduling, coverage optimization, and overhead-aware resource allocation.

PINTO leverages user flows as the telemetry carrier by piggybacking telemetry on existing packets and orchestrating where and how to sample and encode measurements at switches. Since no dedicated probes are injected, PINTO avoids additional measurement flows, but the overhead manifests as per-packet header cost, which is constrained by packet-size headroom and device resources. On the other hand, the orchestration plane has less control over when and where measurements occur, and coverage and freshness are bounded by the spatiotemporal availability of user flows, making it difficult to provide timely visibility for quiet links or rare flows. Consequently, PINTO designs focus on flow selection and sampling policies, in-packet reduction, and mechanisms to mitigate intrusiveness and side effects on production traffic.

HINTO combines AINTO and PINTO within the network to exploit their complementary strengths.

Fig. 4 visualizes the qualitative trade-offs of the three INTO modes along four key dimensions, including cost, flexibility, intrusiveness, and freshness. These axes do not represent unified quantitative metrics, but instead summarize the relative tendencies commonly reported in representative studies and discussed throughout this survey.

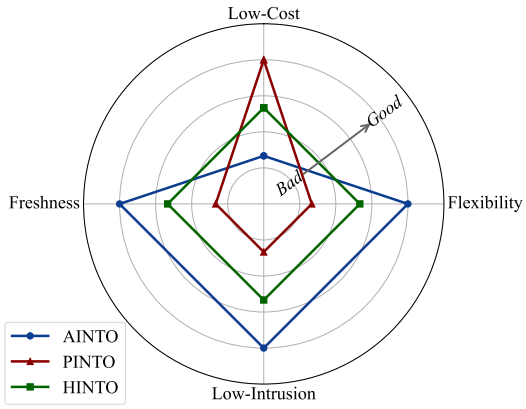


Figure 4: Qualitative comparison of active, passive, and hybrid INTO modes in terms of cost, flexibility, intrusiveness, and freshness. The four dimensions summarize the typical characteristics of the three modes as reported in representative studies.

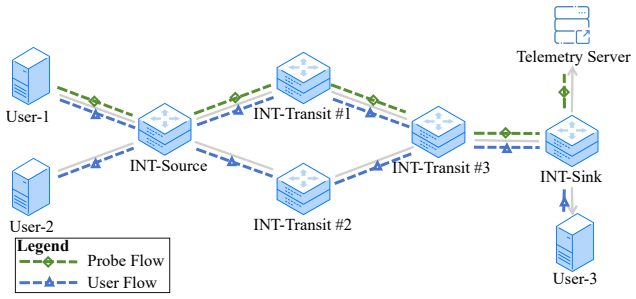


Figure 5: Workflow of AINTO.

3. Active In-band Network Telemetry Orchestration

The intrinsic flexibility and programmability of active INT pose a set of nontrivial orchestration challenges for AINTO systems. These challenges center on how to efficiently design, schedule, and route dedicated probes to maximize telemetry benefits while minimizing overhead and system costs.

Fig. 5 illustrates the operation of AINTO. Unlike passive approaches that depend on user traffic, AINTO explicitly orchestrates probing flows and their schedules, including probing rates and timing. INT-capable devices along each selected probing flow append telemetry metadata, and the sink exports the resulting reports to the telemetry server for analysis and closed-loop control.

This section is organized around two complementary perspectives. We first adopt the task-driven AINTO taxonomy (Section 3.2), categorizing prior works based on its primary objectives. We then review the literature from a method-based AINTO perspective (Section 3.3), organizing representative systems by their core technical approaches.

3.1. Generic AINTO Model

To systematize the diverse landscape of AINTO mechanisms, we formulate active probing within a generic optimization framework that acts as a unifying abstraction for the subsequent analysis. We model the network as a directed graph $G = (V, E)$, within this topology, the orchestration plane seeks to select a set of probe paths $P = \{p_1, \dots, p_K\}$, where each path p_i is a sequence of directed edges. In operational networks, the feasibility of P is strictly constrained by the underlying routing protocols and available path-steering capabilities, thus the optimization is confined to the space of realizable paths.

To quantify the overhead of each selected probe path p, we consider three cost components: bandwidth overhead, latency overhead, and rule-installation overhead. Let b_p, ℓ_p, and r_p denote these three costs for path p, respectively. We then define the weighted objective $J(P)$ as:

$$\min_{P} \; J(P) = \sum_{p \in P} \Big(w_b \, b_p \; + \; w_\ell \, \ell_p \; + \; w_r \, r_p \Big), \tag{6}$$

where w_b, w_ℓ, and w_r are nonnegative weights that balance the trade-off among bandwidth, latency, and rule-installation costs according to operators' monitoring goals and resource bottlenecks. Because these cost terms are heterogeneous in their physical units, Equation (6) is intended as a generic linear abstraction of the trade-off among these objectives, rather than a directly deployable objective with a fixed aggregation procedure. In concrete instantiations, meaningful aggregation would generally require additional normalization or scaling according to the target system and optimization setting. We also acknowledge that this linear formulation does not fully capture non-linear hardware effects in practice, especially when device resources approach saturation.

The probing paths must satisfy a coverage constraint

$$\bigcup_{p \in P} \mathrm{Edges}(p) \supseteq E_{\text{target}}, \tag{7}$$

where $\mathrm{Edges}(p)$ denotes the set of links traversed by path p. This constraint requires that every link in E_{target} is covered by at least one probe path.

In addition, the solution must satisfy the system-level constraints introduced in Section 2.3.1, including MTU, bandwidth, hardware, and freshness limits:

$$C = C_{\text{MTU}}(P) \wedge C_{\text{BW}}(P) \wedge C_{\text{HW}}(P) \wedge C_{\text{AoI}}(P). \tag{8}$$

Under this abstraction, AINTO schemes can be viewed as specific instantiations of the generic problem in Equation (6)–Equation (8). Coverage-aware schemes typically set $E_{\text{target}} = E$ and aim to maintain full topological coverage while minimizing $J(P)$ under the system constraints. Performance-aware schemes emphasize the latency-related term ℓ_p and the freshness constraint C_{AoI}. Overhead-optimized schemes aim to reduce $|P|$ or the bandwidth cost b_p subject to C_{BW}. Finally, an application-aware schemes tailor E_{target}, the cost components, and the weights (w_b, w_ℓ, w_r) to specific service requirements.

3.2. Task-driven AINTO

We categorize task-driven AINTO schemes into four main types: coverage-aware, performance-aware, overhead-optimized,

and application-aware. In the following subsections, we revisit representative schemes in each category and explicitly interpret them as concrete instantiations of the generic model in Equation (6)–Equation (8). We highlight how each scheme selects the target edge set E_{target}, formulates the objective $J(P)$, and enforces different subsets of system-level constraints.

3.2.1. Coverage-aware AINTO

Coverage-aware orchestration adopts the generic active probing model in a straightforward way. It sets $E_{\text{target}} = E$ in Equation (7), requiring every link in the network to be traversed by at least one probe path, and then seeks a probe path set P that covers all links while minimizing the objective $J(P)$ under the system-level constraints in Equation (8). In other words, the primary design goal is to ensure complete topological visibility, while cost terms and constraints are used to refine this baseline towards practical deployability. The theoretical foundation of this category lies in the Chinese Postman Problem (CPP) or Eulerian circuit construction: finding the shortest closed walk that traverses every edge at least once. Evolution in this field can be viewed as the progressive incorporation of real constraints into this idealized graph model.

Idealized Eulerian Construction. Early works focused on topological efficiency under relaxed constraints, effectively minimizing a simplified version of $J(P)$ in which negligible rule and MTU costs. NetVision [15] pioneered this direction by mapping network traversal to the classical Eulerian circuit problem [39]. Leveraging Segment Routing (SR) [40] for source-routed steering, it employs Hierholzer algorithm to generate a single probe path with linear complexity $O(|E|)$. In terms of our model, this corresponds to finding a single Eulerian trail $P = \{p\}$ that satisfies the coverage constraint with $E_{\text{target}} = E$ and approximately minimizes ℓ_p under weak system constraints. However, this approach relies on the idealized assumption that the network graph is balanced and strongly connected, that is, each node has equal in-degree and out-degree.

Real-world topologies rarely satisfy the Eulerian condition naturally. To address this, INT-Path [18] introduces a formalized graph augmentation phase. It proves that a graph with $2k$ odd-degree vertices necessitates at least k disjoint probe paths for full coverage. By employing minimum-weight perfect matching (MWPM), INT-Path optimally pairs these odd-degree vertices and adds virtual edges to Eulerize the graph, thereby achieving the theoretical lower bound on the number of probe paths $|P|$. From the perspective of Equation (6), MWPM effectively constructs a path set P that minimizes the bandwidth and latency components of $J(P)$ while still enforcing $E_{\text{target}} = E$. This strategy decouples path generation from the underlying routing protocol, significantly reducing controller overhead compared to naive traversal schemes.

Deployment Constraints. Although graph-theoretical optimality is elegant, practical deployment introduces hard location constraints that tighten $C_{\text{HW}}(P)$ and related feasibility conditions. In practice, probes cannot originate from arbitrary nodes, they must start and end at specific vantage points. INT-probe [41] addresses this constraint by restricting valid sources and sinks to a pre-defined anchor set $S \subset V$. This restriction transforms the standard CPP into a multi-depot k-CPP, where each path $p \in P$ must begin and terminate at nodes in S. The orchestration algorithm applies a constrained MWPM to eliminate odd-degree nodes outside the anchor set S, ensuring valid probe endpoints while minimizing path overlap. Under our model, the feasible region of P is shrunk by deployment-aware constraints, while the objective $J(P)$ still balances bandwidth and latency costs.

Physical Resource Constraints. A critical limitation of ideal Eulerian formulations is the MTU constraint $C_{\text{MTU}}(P)$. A network-wide Eulerian trail is typically too long to fit within a single packet metadata stack. To resolve this, INT-Segment [42] introduces a construct-and-segment methodology. It first generates a global Eulerian trail via an optimal MWPM or a greedy heuristic and then decomposes it into multiple probe paths that respect MTU constraints using dynamic programming (DP). In terms of Equation (8), the DP-based segmentation is explicitly designed to guarantee that the accumulated metadata size $\sum l_{\text{meta}}$ of each segment satisfies $C_{\text{MTU}}(P)$ while preserving $E_{\text{target}} = E$.

Addressing the heterogeneity of production networks, MTU-Adaptive [43] extends this segmentation logic. Instead of enforcing a static global MTU, it dynamically calculates the bottleneck MTU for each candidate segment and incorporates a delay constraint related to $C_{\text{AoI}}(P)$. By sorting adjacency lists based on MTU values during path construction, it jointly optimizes telemetry intrusion and data freshness, thereby refining both the objective $J(P)$ and the constraint set in a topology-aware manner.

Scalability and Resilience. Centralized matching algorithms scales poorly in mega-scale networks, which in our abstraction manifests as the increasing computational cost of solving Equation (6) over large P. INT-react [34] proposes a linear-time graph reconstruction method by introducing virtual vertices to connect odd-degree pairs, transforming the problem into a single Eulerian circuit computation that can be split in $O(|E|)$ time. This enables millisecond-level recomputation in response to network dynamics while still maintaining full coverage under the same constraint set.

Furthermore, to mitigate the single point of failure in centralized orchestration and relax strict hardware constraints $C_{\text{HW}}(P)$, INT-Partition [44] proposes hierarchical orchestration. It utilizes an edge-clustering algorithm to decompose the network into weakly coupled regions, each managed by a local controller, while a global controller coordinates inter-region connectivity. This divide-and-conquer strategy preserves the coverage requirement $E_{\text{target}} = E$ while distributing the optimization of $J(P)$ across multiple controllers, significantly enhancing system robustness.

Table 4 summarizes representative coverage-aware AINTO

Table 4
Comparison of Representative Coverage-aware AINTO Schemes

Scheme	Year	Key Algorithm / Technique	Exec. Mode		Aspects Considered			
			Stat	Dyn	MTU	BW	AoI	HW
NetVision [15]	2018	Hierholzer Algorithm	✓		-	-	✓	-
INT-Path [18]	2019	Optimized Euler Trail	✓		-	✓	✓	-
INT-probe [41]	2021	MWPM	✓		-	✓	✓	✓
INT-react [34]	2022	Linear-time Graph Reconstruction		✓	-	-	-	-
INT-Segment [42]	2022	Dynamic Programming	✓		✓	-	-	-
MTU-Adaptive [43]	2024	Heterogeneous MTU-aware Segmentation	✓		✓	✓	-	-
INT-Partition [44]	2025	Graph Partitioning	✓		-	-	-	✓

Stat/Dyn: Static Planning / Dynamic Execution.

Table 5
Preconditions, Evaluation Settings, and Deployment Conditions of Representative Coverage-aware AINTO Schemes

Scheme	Preconditions	Evaluation Setting	Deployment Condition
NetVision [15]	Active probes; SR-based steering; designated vantage points	Mininet; Fat-Tree with application cases	Centrally managed network; SR support
INT-Path [18]	Global topology knowledge; source routing	Prototype/simulation; random graphs and DCNs	Centralized control; structurally regular DCNs
INT-probe [41]	Anchor-constrained deployment; centrally planned paths	P4 prototypes; WAN/DCN topologies	Fixed probe anchors; controller support
INT-react [34]	Global topology awareness; fast replanning	Megascale DCNs; planning efficiency and path balance	Large dynamic DCNs; fast centralized replanning
INT-Segment [42]	Single-path planning; DP-based segmentation	Random graphs/DCNs; latency and path length	MTU-aware telemetry; flexible INT Source placement
MTU-Adaptive [43]	Per-link MTU awareness; latency-aware adjustment	Hardware/Mininet/BMv2; large-scale DCNs	Heterogeneous MTUs; dynamic telemetry demand
INT-Partition [44]	Hierarchical partitioning; per-partition telemetry	BMv2 simulations; small-scale Tofino validation	Hyperscale networks; hierarchical orchestration

schemes and compares their architectures, execution modes, and the constraints they explicitly consider. Table 5 further summarizes their key assumptions, evaluation settings, and deployment conditions. Together, these two tables show that the evolution of coverage-aware orchestration reflects a transition from theoretical graph traversal to constraint-driven engineering under the unified model of Equation (6)–Equation (8). Early Eulerian-based approaches focused on establishing the theoretical lower bounds on probing efficiency, thereby achieving strong topological coverage with relatively simple graph-theoretic constructions, but this efficiency was often obtained under more idealized assumptions and with limited consideration of deployment realism [15, 18]. Subsequent works progressively incorporated practical considerations such as anchor placement, MTU fragmentation, and controller scalability, thereby improving deployability and adaptability, but also making the orchestration process more constrained and complex [41, 42, 43, 34, 44]. Therefore, the design of coverage-aware AINTO is fundamentally a trade-off between coverage completeness and practical feasibility: stronger coverage guarantees are desirable, but they must be balanced against packet-size limits, control-plane scalability, and execution constraints in real networks. Accordingly, coverage-aware AINTO is particularly suitable for structured and centrally managed environments, such as data center fabrics or other programmable networks, where broad topological visibility is a primary requirement. However, achieving full topological coverage is merely the baseline and does not guarantee the quality of the collected telemetry. Coverage-centric designs often result in probe paths with substantial length variance, which leads to asynchronous data reporting and stale information, i.e. limited control over $C_{\mathrm{AoI}}(P)$. This further highlights another trade-off between completeness and timeliness: a probe set that maximizes coverage may still provide poorly synchronized or stale telemetry. This limitation motivates a shift from optimizing completeness to optimizing quality, giving rise to the performance-aware strategies discussed in the subsequent section.

3.2.2. Performance-aware AINTO

While coverage-aware orchestration ensures visibility, it often overlooks the temporal quality of the collected data and the significant variance in probe path lengths. In the generic model of Equation (6)–Equation (8), coverage-aware schemes primarily enforce the coverage constraint with $E_{\text{target}} = E$, but exert limited control over the latency-related term ℓ_p and the freshness constraint $C_{\text{AoI}}(P)$. In complex operational environments, telemetry data must be delivered with high synchronization, minimal latency, and robust fault tolerance. This section reviews performance-aware strategies that refine the objective $J(P)$ and the constraint set toward these dimensions, evolving from static path balancing to dynamic timeliness assurance and resilient orchestration.

Path Synchronization. A primary challenge in static path planning is the straggler effect, where uneven probe path lengths lead to asynchronous data arrival. To address this within a single domain, CFINT [45] employs a greedy clustering algorithm based on the minimum dominating set. By decomposing the network into balanced micro-domains and restricting probes to intra-cluster paths, it enforces a strict upper bound on hop counts, significantly reducing latency variance. Under the unified model, CFINT can be interpreted as constraining the feasible path set P so that the deviation in ℓ_p across paths is bounded, thereby implicitly tightening $C_{\text{AoI}}(P)$.

For digital twin networks, GP-INT [46] aims to generate a set of balanced INT probing paths rather than a single network-wide trail. It first partitions the topology into dense subgraphs via community detection, and then applies a depth-limited search (DLS) within each partition to construct probing paths under MTU-induced hop limits. Assuming link delays are comparable, GP-INT uses the deviation of path lengths to reduce probing-time skew and thus improve synchronization across paths. Formally, given a network graph $G = (V, E)$, it plans a probing path set P by minimizing a composite objective:

$$\min_{P} \ \Phi(P) = \sigma_L(P) + |P| + \sum_{p\in P}\sum_{v\in V} x_p^v, \tag{9}$$

where $\sigma_L(P)$ is the standard deviation of path lengths in P, and $x_p^v \in \{0,1\}$ indicates whether path p covers node v. The probing paths must jointly cover all nodes:

$$\sum_{p\in P} x_p^v \geq 1, \forall v \in V, \tag{10}$$

and each probing path is bounded by a hop limit to avoid packet fragmentation:

$$|p| < H_{\max}, \forall p \in P, \tag{11}$$

$$x_p^v \in \{0,1\}, \forall v \in V, \ \forall p \in P. \tag{12}$$

Focusing on fine-grained structural optimization of Euler trails, INT-balance [47] introduces a heuristic break-and-reconnect strategy. Unlike traditional traversals, it decomposes the graph into short segments and iteratively splices them to prioritize length uniformity, thereby minimizing the maximum telemetry cycle delay. In terms of the generic formulation, this reduces the worst-case ℓ_p across $p \in P$, directly improving the temporal consistency of telemetry snapshots.

Addressing performance consistency across cross-domain boundaries, SRv6-ALINT [48] extends this balancing logic to large-scale interconnected networks. It utilizes a centralized controller that executes a DFS-stitch heuristic to iteratively merge shorter paths with longer ones. This approach minimizes the variance between maximum and minimum path lengths while enabling privacy-preserving telemetry via segment routing over IPv6 (SRv6) header manipulation. Additionally, for wide area networks (WANs) with heterogeneous links, WAN-INT [49] proposes a cost-aware planner that filters low-quality links and incrementally extends probing paths using constrained depth-first search until a cumulative delay threshold is reached. This explicitly bounds end-to-end delay within $C_{\text{AoI}}(P)$, balancing coverage and synchronization in latency-sensitive environments.

Timeliness Optimization. Static graph algorithms often overlook real-time link-state fluctuations. To address this, INTView [50] models the topology as a directed graph whose edge weights are the instantaneous link latencies, and constructs an Euler circuit E_c over the directed topology. The circuit is then partitioned into K non-overlapping probing paths so as to minimize the straggler path, defined as the probing path with the maximum accumulated latency. Let E_c denote an Euler circuit indexed by $c \in C$, let $d_i(t)$ be the real-time latency (edge weight) of link $i \in E_c$ at planning time t, and let $I_{i,j,c} \in \{0,1\}$ indicate whether link i is assigned to the j-th probing path ($j \in \{1,\dots,K\}$) on circuit c. INTView formulates the planning objective as:

$$\min_{c\in C} \ \max_{1\leq j\leq K} \sum_{i\in E_c} I_{i,j,c}\, d_i(t), \tag{13}$$

subject to the each directed link is covered exactly once on the chosen circuit:

$$\sum_{j=1}^{K} I_{i,j,c} = 1, \forall i \in E_c, \ \forall c \in C, \tag{14}$$

and a controller bandwidth budget that upper-bounds the number of probing paths:

$$K\,T \leq \beta, \tag{15}$$

where T is the per-probe bandwidth overhead and β is the controller-side bandwidth budget. Under this formulation, time-varying link latencies enter the orchestration through $d_i(t)$, and minimizing the worst-case path latency directly reduces the INT job completion time under dynamic network conditions.

Taking adaptivity a step further, AdapINT [35] formulates the dynamic probe-path deployment as a multi-objective optimization problem over a set of dynamic probe paths $\{d_q\}_{q=1}^{Q}$. Let $S \subseteq E$ denote the service network, and let L'_q be the set

of links traversed by the q-th dynamic probe path d_q. AdapINT aggregates multiple telemetry performance indicators $F = \{f_1, \ldots, f_m\}$ using a weighted objective:

$$C = \sum_{i=1}^{m} w_i f_i, \tag{16}$$

and selects the dynamic probe paths to minimize this objective:

$$\min_{\{d_q\}_{q=1}^{Q}} C. \tag{17}$$

Under our generic model, S plays the role of E_{target} and $\{d_q\}$ corresponds to the selected probing path set. Moreover, the weight vector $\{w_i\}$ in Equation (16) follows the same trade-off principle as the generic weighted objective in Equation (6), balancing bandwidth, latency, and rule-installation costs via (w_b, w_ℓ, w_r) over (b_p, ℓ_p, r_p).

In closed-loop control scenarios where data value decays rapidly over time, Freshness-INT [51] explicitly integrates the AoI metric into orchestration. It formulates the problem as a multi-objective optimization task minimizing the weighted sum of deployment cost and average AoI. By employing a variable neighborhood search heuristic, the system dynamically adjusts probe frequencies to ensure decision-making is based on the freshest possible states. From the perspective of the generic formulation, this approach jointly optimizes $J(P)$ while tightening the freshness constraint $C_{\text{AoI}}(P)$ according to service timeliness requirements.

Hierarchical Robustness. Centralized orchestration suffers from scalability and fault-tolerance bottlenecks. To address this at different granularities, FANT [33] proposes a hierarchical framework that decouples telemetry into two complementary layers: coarse-grained baseline monitoring, which performs low-frequency measurements to track global network health, and on-demand fine-grained monitoring, which triggers high-precision probing for fault localization. This dual-layer design enables the system to dynamically switch strategies upon detecting anomalies, balancing resource consumption and fault sensitivity while maintaining feasible solutions to Equation (6) under varying failure scenarios.

To handle rapid link failures, Patcher [52] introduces a reactive fault-tolerance mechanism. Rather than recomputing probing paths globally, it employs a localized shortest-path repair algorithm to reconnect disrupted segments, maximizing the similarity between pre- and post-failure configurations. Within the generic formulation, this approach can be viewed as a local update of the probing path set P to restore feasibility with respect to the coverage constraint, without fully re-solving the global optimization problem. To achieve autonomous resilience, INT-Source [53] pushes the orchestration logic entirely to the data plane. It utilizes a unique data plane deduplication mechanism where switches forward probes based on telemetry identifiers. This allows probes to implicitly perform a loop-free Breadth-First Search (BFS), adapting to topology changes at line rate and effectively maintaining the constraints in Equation (8) without centralized coordination.

Complementing these reactive approaches, ATINT [54] advocates proactive robustness by embedding anomaly tolerance into the initial probing plan. Let P denote the set of probe cycles and let $x_{e,p} \in \{0, 1\}$ indicate whether directed link $e \in E$ is traversed by cycle $p \in P$. ATINT enforces a k-redundancy requirement (with $k = d + 1$) so that every link is covered by at least $d+1$ cycles:

$$\sum_{p \in P} x_{e,p} \geq d + 1, \forall e \in E. \tag{18}$$

Moreover, to guarantee uninterrupted visibility for regular links under up to d concurrent anomalies, ATINT requires the cycles covering the same link e to be disjoint on all other links. In terms of the generic model, these resilience constraints expand the feasible set by ensuring that even if probe packets on up to d anomaly links are disrupted, at least one unaffected cycle remains to report the state of any other link.

Performance-aware AINTO has evolved from static path synchronization to dynamic, adaptive, and resilient orchestration. Early schemes mainly reduced straggler effects through path balancing and synchronization-oriented graph decomposition, whereas later designs progressively incorporated delay-sensitive planning, online adaptation, and fault-tolerant coordination. Methodologies have progressed from graph clustering for synchronization to delay-sensitive adaptive planning, and finally to fault-tolerant architectures. Table 6 summarizes representative performance-aware AINTO schemes in terms of their architecture, execution modes, and the emphasized constraints. Table 7 further summarizes their key assumptions, evaluation settings, and deployment conditions. Taken together, these representative schemes indicate that the design of performance-aware AINTO is fundamentally shaped by trade-offs among synchronization quality, timeliness, adaptability, and robustness: stronger guarantees on telemetry freshness and consistency are desirable, but they usually require more dynamic control, richer state awareness, and more complex coordination mechanisms. Improving synchronization and freshness usually requires tighter control over path length, delay, or probing frequency, but this also increases orchestration complexity and dependence on real-time network information [45, 46, 47, 48, 49]. Likewise, enhancing robustness and responsiveness under failures improves adaptability, but often at the cost of additional redundancy, hierarchical coordination, or more sophisticated repair mechanisms [33, 52, 53, 54]. This progression reflects a broader shift from treating the network as a static graph to treating it as a dynamic system that requires real-time, quality-assured telemetry delivery under the unified optimization framework of Equation (6)–Equation (8). These characteristics make performance-aware AINTO particularly suitable for latency-sensitive and dynamically changing environments, such as 5G/6G-oriented networks, digital twin systems, and large-scale wide-area deployments that require timely and consistent telemetry feedback.

Table 6
Comparison of Representative Performance-aware AINTO Schemes

Scheme	Year	Key Algorithm / Technique	Exec. Mode		Aspects Considered			
			Stat	Dyn	MTU	BW	AoI	HW
Patcher [52]	2020	Localized Path Repair		✓	-	-	-	✓
CFINT [45]	2022	Greedy Clustering	✓		-	-	✓	-
INT-balance [47]	2023	Break-and-Reconnect	✓		-	-	✓	-
WAN-INT [49]	2023	Constrained DFS with Delay Threshold	✓		-	-	✓	-
INTView [50]	2023	Real-time Weighted Links and Segment Partitioning		✓	-	-	✓	-
GP-INT [46]	2024	Community Detection and DLS Partitioning	✓		-	-	✓	-
AdapINT [35]	2024	DRL-based Dual-timescale Probe Dispatch		✓	-	-	✓	-
Freshness-INT [51]	2025	Variable Neighborhood Search		✓	-	-	✓	✓
SRv6-ALINT [48]	2025	DFS-stitch Heuristic	✓		✓	-	✓	-
INT-Source [53]	2025	Data plane Deduplication		✓	-	-	✓	✓
FANT [33]	2025	Hierarchical Baseline and On-demand Triggering		✓	-	✓	✓	-
ATINT [54]	2025	Enhanced Dijkstra with k-redundancy	✓		-	-	-	✓

Stat/Dyn: Static Planning / Dynamic Execution.

3.2.3. Overhead-optimized AINTO

In active telemetry systems, the injection and forwarding of probes inevitably introduce non-negligible overheads, including header expansion, bandwidth consumption, and additional processing load. If left unmanaged, these factors can degrade network performance. From the perspective of the generic model in Equation (6), overhead-optimized AINTO aims to explicitly minimize the bandwidth-related component b_p and the size of the probe set P under the system-level constraints of Equation (8). This section reviews overhead-optimization strategies that address these challenges across three progressive dimensions: header space reuse, global flows planning, and redundancy data elimination.

Header Overhead Reduction. The most fundamental challenge in INT is the header explosion problem, where metadata accumulation leads to MTU violations. To address this at the protocol level, SR-INT [55] proposes a header time-multiplexing mechanism. Instead of appending new metadata at every hop, it reuses the SR label stack to store telemetry data. As packets traverse the network, transit nodes dynamically overwrite the used SR labels with collected measurements. This design keeps packet length constant end-to-end, effectively eliminating MTU risks while reducing per-packet bandwidth consumption. Under the unified formulation, SR-INT can be seen as a protocol-level refinement that relaxes the MTU constraint and shrinks bandwidth cost for each path.

Global Bandwidth Minimization. While per-packet optimization is necessary, minimizing the overall bandwidth footprint requires intelligent coordination across multiple probe flows. Building upon SR-INT, SR-INT Orchestration [56] formulates the joint scheduling of multiple service flows as a Mixed-Integer Linear Program (MILP), which is approximately solved using column generation. Let $\mathcal{F}$ denote the set of service flows, and let C_k be the candidate SR path set (columns) for flow $k \in \mathcal{F}$. Binary variable $\lambda_{k,c}$ indicates whether flow k selects path $c \in C_k$, and $\rho_{u,v}^{k,c} \in \{0,1\}$ indicates whether link $(u,v) \in E$ is traversed by that path. With b_k being the bandwidth demand of flow k and $\tilde{y}_v$ representing the monitoring benefit obtained at node $v \in V$, the master problem minimizes a bandwidth–coverage trade-off:

$$\min \frac{w_b}{W} \sum_{k\in\mathcal{F}} \sum_{c\in C_k} \lambda_{k,c} \Big(\sum_{(u,v)\in E} \rho_{u,v}^{k,c}\, b_k \Big) - w_c \sum_{v\in V} \tilde{y}_v, \quad (19)$$

where W is a normalization constant and (w_b, w_c) balance bandwidth minimization against monitoring benefit. Each flow selects exactly one routing column:

$$\sum_{c\in C_k} \lambda_{k,c} = 1, \forall k \in \mathcal{F}, \quad (20)$$

and the aggregate bandwidth on each link is bounded by its capacity $B_{u,v}$:

$$\sum_{k\in\mathcal{F}} \sum_{c\in C_k} \lambda_{k,c}\, \rho_{u,v}^{k,c}\, b_k \le B_{u,v}, \forall (u,v) \in E. \quad (21)$$

In terms of Equation (6), this formulation is dominated by the bandwidth component, while $\tilde{y}_v$ captures system-level coverage utility.

Adopting a swarm intelligence perspective, AINTO [57] focuses on minimizing hop-by-hop overhead accumulation by employing a Dragonfly Algorithm, a nature-inspired metaheuristic, combined with preprocessing pruning to efficiently solve the bandwidth minimization model. This approach identifies optimal probe paths that satisfy connectivity constraints while minimizing the global bandwidth budget. Similarly, INT-LLPP [58] proposes a unified probe architecture

Table 7
Preconditions, Evaluation Settings, and Deployment Conditions of Representative Performance-aware AINTO Schemes

Scheme	Preconditions	Evaluation Setting	Deployment Condition
Patcher [52]	Local repair after node failures	Recovery efficiency; collector load	Fault-tolerant telemetry; programmable networks
CFINT [45]	Cluster-based planning; SR under MTU limits	BMv2; latency and bandwidth overhead	SDN-enabled network; synchronized telemetry
INT-balance [47]	Global topology knowledge; balanced path generation	Multiple topologies; path balance and execution time	Centralized control; timely telemetry collection
WAN-INT [49]	Performance-aware orchestration; quality-cost balancing	Commercial WAN; cost and quality	Managed WANs; heterogeneous links
INTView [50]	Latency-aware planning; no-detour probing	Real devices; large-scale settings	Low-latency telemetry; programmable networks
GP-INT [46]	Graph partitioning; depth-limited planning	Simulation/P4 implementation; balanced collection	Programmable network; MTU-constrained telemetry
AdapINT [35]	Dual-timescale probes; DRL-based adaptation	Diverse scenarios; latency, overhead, adaptability	Dynamic network; changing telemetry demands
Freshness-INT [51]	AoI-guided planning; constrained resources	Resource-constrained settings; freshness and cost	Large-scale network; freshness-aware telemetry
SRv6-ALINT [48]	SRv6-guided telemetry; boundary-node uploading	Cross-LAN settings; path balance and compression	Multi-LAN deployment; SRv6 support
INT-Source [53]	Topology-adaptive forwarding; no centralized planning	BMv2/Tofino; coverage and robustness	Dynamic DCNs; low controller dependence
FANT [33]	Centralized probe control; dynamic granularity switching	WAN/DCN topologies; different scales	Flexible granularity; programmable networks
ATINT [54]	Redundant staggered probes; bounded anomalies	Interruption probability; monitoring continuity	Production network; anomaly-tolerant telemetry

that consolidates the monitoring requirements of multiple services into a single probe set, and adopts a heuristic path-planning algorithm to jointly minimize telemetry delay and signaling or forwarding overheads. Within the generic model, these schemes reduce both b_p and ℓ_p while satisfying $C_{\mathrm{BW}}(P)$ and other system-level constraints.

Redundant Data Elimination. As the network scale increases, periodic probing can generate massive amounts of redundant telemetry data. To mitigate this inefficiency, Cache-INT [59] incorporates in-network caching to enable data reuse. It employs an incremental reporting mechanism where switches telemetry updates only when the deviation from the real-time state and the cached snapshot exceeds a threshold. Combined with offline path optimization via deep reinforcement learning (DRL), Cache-INT significantly reduces invalid flows, effectively shrinking the realized bandwidth cost term in $J(P)$.

Further efficiency can be achieved by spatially pruning monitoring targets. Probe-Optimizer [60] focused on identifying structurally significant nodes using centrality metrics weighted by flow volume and latency sensitivity. By generating probing paths exclusively among critical nodes rather than sweeping the entire topology, it maximizes information gain per unit of overhead. This approach can be viewed as redefining E_{target} in Equation (7) to a smaller, application-aware subset.

Finally, OpenINT [61] addresses rigidity in telemetry collection through a lightweight and dynamically reconfigurable framework. Its standout feature, dynamic metadata attachment, allows data plane devices to flexibly insert only task-relevant telemetry fields on demands. Coordinated by a heuristic path search algorithm with $O(N^2)$ complexity, OpenINT enables elastic task updates with minimal resource usage, pushing the orchestration closer to an on-demand realization of the generic optimization model.

Overhead optimization in AINTO has evolved from static header compression to intelligent orchestration. Early designs mainly reduced per-packet overhead through protocol-level header reuse, whereas later schemes progressively optimized the bandwidth footprint of multiple probe flows and further improved efficiency by suppressing redundant data or focusing only on high-value monitoring targets. Methodologies have progressed from reducing the size of single packets to optimizing global flows via meta-heuristics, and finally to eliminating semantic redundancy through caching and critical node identification. This evolution reveals several clear trade-offs. Reducing per-packet telemetry overhead improves feasibility under MTU and bandwidth constraints, but often limits the flexibility of metadata representation [55]. Global flow planning improves network-wide bandwidth efficiency, but it requires more complex optimiza-

Table 8
Comparison of Representative Overhead-optimized AINTO Schemes

Scheme	Year	Key Algorithm / Technique	Exec. Mode		Aspects Considered			
			Stat	Dyn	MTU	BW	AoI	HW
SR-INT [55]	2021	Header Time-Mux	✓		✓	✓	-	✓
SR-INT Orchestration [56]	2023	MILP and Path Ranking Heuristic	✓		-	✓	-	-
AINTO [57]	2023	Dragonfly Algorithm	✓		-	✓	-	-
OpenINT [61]	2024	Heuristic Path Search		✓	-	✓	-	✓
INT-LLPP [58]	2025	Heuristic Path Planning		✓	-	✓	✓	-
Cache-INT [59]	2025	In-network Caching	✓		-	✓	-	✓
Probe-Optimizer [60]	2025	Centrality Analysis		✓	-	✓	-	-

Stat/Dyn: Static Planning / Dynamic Execution.

Table 9
Preconditions, Evaluation Settings, and Deployment Conditions of Representative Overhead-optimized AINTO Schemes

Scheme	Preconditions	Evaluation Setting	Deployment Condition
SR-INT [55]	Time-multiplexed SR/INT headers; constant packet length	POF-based SDN testbed; bandwidth overhead	Programmable SDN; SR and INT support
SR-INT Orchestration [56]	Service-flow-aware joint SR/INT planning	Simulation; coverage and bandwidth trade-offs	Centralized control; active service flows
AINTO [57]	Source-routed probes; joint path-item planning	Real-world topologies; bandwidth and freshness	Large-scale network; centralized orchestration
OpenINT [61]	Decoupled telemetry modules; online reconfiguration	Tofino prototype; flexibility and overhead reduction	Programmable DCNs; non-disruptive updates
INT-LLPP [58]	Shared probe set; latency-constrained planning	Simulation/testbed; overhead and latency	Low-latency telemetry; programmable networks
Cache-INT [59]	Cache-enabled routers; incremental transmission	Fixed/flexible caching scenarios; transmission overhead	Cache-enabled network; lightweight updates
Probe-Optimizer [60]	Node-importance-aware planning; differentiated frequencies	Random/FatTree topologies; overhead and CPU usage	Proactive INT; overhead-aware orchestration

tion and coordination across services [56, 57, 58]. Redundancy elimination and target pruning further reduce realized telemetry cost, but they may rely more heavily on accurate thresholding, application-specific target selection, or dynamic control support [59, 60, 61]. This evolution indicates a future trend towards on-demand, lightweight, and value-centric telemetry systems in which $J(P)$ is dominated by information efficiency rather than raw coverage. Table 8 summarizes representative overhead-optimized AINTO schemes and contrasts their key techniques and addressed constraints. Table 9 further summarizes their key assumptions, evaluation settings, and deployment conditions. Together, these two tables show that the design of overhead-optimized AINTO is fundamentally a trade-off between telemetry cost and information value: lower bandwidth and lighter probe overhead are desirable, but they often come at the cost of reduced coverage generality, stronger dependence on workload characteristics, or more sophisticated orchestration logic. Accordingly, overhead-optimized AINTO is particularly suitable for large-scale production networks and other cost-sensitive deployment settings, where continuous telemetry is needed but bandwidth, packet headroom, and processing resources must be tightly controlled.

3.2.4. Application-aware AINTO

In practical operational environments, indiscriminate network-wide monitoring is often inefficient, as upper-layer applications typically require targeted insights rather than raw global data. As a result, application-aware orchestration has emerged as a critical approach, shifting from resource-driven comprehensive sensing to demand-driven precise monitoring. Within the unified model of Equation (6)–Equation (8), application-aware AINTO customizes the target set E_{target}, the cost terms in $J(P)$, and even the constraint set to align the telemetry orchestration with specific application objectives. This section reviews strategies that bridge this gap, ranging from generic logic orchestration to deep protocol integration and scenario-

specific adaptation.

Generic Application Logic Orchestration. The foundational challenge in application-aware INTO is translating abstract monitoring intents into concrete probe configurations. P^{2}INT [62] addresses this challenge by formalizing requirement driven probing as a MILP. Given a network $G = (V, E)$ and a set of required telemetry tasks $\mathcal{R}$, P^{2}INT selects a minimum-size set of probing cycles that jointly ensures task coverage and link visibility under packet-capacity constraints. Let $\mathcal{P} = \{1, \dots, |\mathcal{P}|\}$ be an upper-bounded index set of candidate cycles. Binary variable y_p indicates whether cycle $p \in \mathcal{P}$ is activated, $x_{p,i,j}$ indicates whether directed link $(i, j) \in E$ is used by cycle p, and $z_{p,t,i}$ indicates whether cycle p collects telemetry item t at node i. The MILP can be written as:

$$\min \sum_{p\in\mathcal{P}} y_p \tag{22a}$$

$$\text{s.t.} \sum_{p\in\mathcal{P}} z_{p,t,i} = 1, \forall (i,t) \in \mathcal{R}, \tag{22b}$$

$$z_{p,t,i} \le \sum_{j\in V} x_{p,j,i}, \forall p \in \mathcal{P},\ (i,t) \in \mathcal{R}, \tag{22c}$$

$$z_{p,t,i} + x_{p,i,j} \le 2y_p, \forall p \in \mathcal{P},\ (i,j) \in E,\ t \in \mathcal{T}_i, \tag{22d}$$

$$\sum_{j\in V} x_{p,i,j} - \sum_{j\in V} x_{p,j,i} = 0, \forall p \in \mathcal{P},\ i \in V, \tag{22e}$$

$$\sum_{p\in\mathcal{P}} \left(x_{p,i,j} + x_{p,j,i}\right) \ge 1, \forall (i,j) \in E_{\text{target}}, \tag{22f}$$

$$\sum_{i\in V}\sum_{t\in\mathcal{T}_i} z_{p,t,i}\, s_t + \sum_{(i,j)\in E} x_{p,i,j} \le U_p, \forall p \in \mathcal{P}, \tag{22g}$$

$$\sum_{i\in S}\sum_{j\in S} x_{p,i,j} \le |S| - 1, \forall p \in \mathcal{P},\ S \subseteq V \setminus \{o_p\}, \tag{22h}$$

$$|S| \ge 2.$$

$$x_{p,i,j} \in \{0,1\},\ z_{p,t,i} \in \{0,1\},\ y_p \in \{0,1\}. \tag{22i}$$

Here, $\mathcal{T}_i$ denotes the set of telemetry items available at node i, s_t is the size of item t, and U_p is the packet-space budget of cycle p (bounded by MTU and encapsulation overhead). Constraint (22b) enforces requirement satisfaction, (22c) ensures items can only be collected if the cycle visits the node, (22e) imposes flow conservation to form valid cycles, (22f) implements link-visibility over a task-specific target set E_{target}, and (22g) captures the capacity trade-off between collected telemetry and cycle length. The subtour-elimination constraint (22h) further enforces strong connectivity. Within the generic formulation, P^{2}INT instantiates E_{target} according to application requirements and sets $J(P)$ to prioritize minimizing the number of activated cycles under capacity constraints.

Building upon the requirement-driven MILP of P^{2}INT, DyPro [32] explicitly models correlated telemetry demands by introducing spatial dependencies, namely groups of telemetry items that must be collected atomically from the same forwarding device within the same measurement round. Let $\mathcal{M}$ denote the set of monitoring applications, and let $\mathcal{R}^s_m$ be the collection of dependency groups required by application $m \in \mathcal{M}$, where each group $g \in \mathcal{R}^s_m$ is a set of telemetry items that must be jointly collected. Using the same decision variable $z_{p,t,i} \in \{0,1\}$, DyPro introduces an auxiliary counter $s_{m,i,p,g}$ to count how many items of group g are collected at device i by probing cycle p:

$$s_{m,i,p,g} = \sum_{t\in g} z_{p,t,i}, \forall m \in \mathcal{M}, \forall g \in \mathcal{R}^s_m,\ \forall i,\ \forall p. \tag{23}$$

And enforces all-or-nothing satisfaction by requiring:

$$\frac{s_{m,i,p,g}}{|g|} = 1, \forall m \in \mathcal{M}, \forall g \in \mathcal{R}^s_m,\ \forall i. \tag{24}$$

Moreover, to cope with evolving requirements, DyPro optimizes reconfiguration cost by minimizing deviations from a previous orchestration plan. It introduces absolute-difference variables:

$$z^{\oplus}_{p,t,i} = \left|z_{p,t,i} - z^{\bullet}_{p,t,i}\right|, \forall p,\ \forall t,\ \forall i, \tag{25a}$$

$$x^{\oplus}_{p,i,j} = \left|x_{p,i,j} - x^{\bullet}_{p,i,j}\right|, \forall p,\ \forall (i,j), \tag{25b}$$

$$y^{\oplus}_{p} = \left|y_{p} - y^{\bullet}_{p}\right|, \forall p. \tag{25c}$$

Accordingly, the static objective is replaced by a reconfiguration aware objective that minimizes the total amount of changes:

$$\min \sum_{p}\sum_{t}\sum_{i} z^{\oplus}_{p,t,i} + \sum_{p}\sum_{(i,j)} x^{\oplus}_{p,i,j} + \sum_{p} y^{\oplus}_{p}. \tag{26}$$

Targeting NFV service function chains (SFCs), IntOpt [63, 11] optimizes the deployment of active INT monitoring flows (MFs) over the subset of physical links $\tilde{E}$ that are actually traversed by SFCs. It maps SFC SLA requirements to per-link telemetry demands, including (i) the telemetry item demand δ_{ij} and (ii) a strict upper bound on the probing interval τ_{ij}. The controller then selects and routes a set of candidate MFs $\mathcal{F}$ to minimize overall monitoring overhead while satisfying SLA-driven constraints:

$$\min\ \alpha_{\text{mf}} \sum_{f\in\mathcal{F}} a_f + \beta_{\text{util}} \sum_{f\in\mathcal{F}}\sum_{(i,j)\in\tilde{E}} \phi^f_{ij}\, \delta_{ij}\, a_f, \tag{27}$$

where $a_f \in \{0,1\}$ indicates whether MF f is activated, and $\phi^f_{ij} \in \{0,1\}$ indicates whether MF f traverses physical link (i, j). The first term captures the fixed per-MF overhead, while the second term approximates data plane processing and utilization overhead that scales with telemetry demand.

$$\sum_{(i,j)\in\tilde{E}} \phi^f_{ij}\, \delta_{ij} \le \lambda_f, \quad \forall f \in \mathcal{F}, \tag{28}$$

Table 10
Comparison of Representative Application-aware AINTO Schemes

Scheme	Year	Key Algorithm / Technique	Exec. Mode		Aspects Considered			
			Stat	Dyn	MTU	BW	AoI	HW
P²INT [62]	2021	MILP and Fix-and-Optimize Heuristic	✓		-	✓	-	-
DyPro [32]	2022	Dependency-aware Heuristic		✓	✓	-	✓	-
IntOpt [63]	2022	Hybrid Meta- Heuristic	✓		-	✓	✓	-
SFANT [64]	2023	P4-INT and SRv6 Integrated Framework	✓		-	-	✓	-
INToSR [65]	2024	SRv6 Header Embedding	✓		✓	-	-	✓
FDSR-INT [66]	2025	Tabu Search	✓		✓	✓	✓	-

Stat/Dyn: Static Planning / Dynamic Execution.

Table 11
Preconditions, Evaluation Settings, and Deployment Conditions of Representative Application-aware AINTO Schemes

Scheme	Preconditions	Evaluation Setting	Deployment Condition
P²INT [62]	Capacity-bounded probes; joint link-item coverage	Prior probing baselines; probe count and resource usage	Packet-capacity constraints; active telemetry
DyPro [32]	Application-aware planning; spatiotemporal dependency	Changing requirements; application satisfaction rate	Dynamic orchestration; application-driven telemetry
IntOpt [63]	Active monitoring flows; joint path-item-frequency optimization	P4-FPGA and numerical evaluation; overhead and delay	SDN/NFV environments; service-chain monitoring
SFANT [64]	INT/SRv6 probes; flexible path-item specification	Server-based simulation; accuracy, path cost, bandwidth	SRv6-based control; flexible active telemetry
INToSR [65]	SRv6 programmability; critical-node measurement	Tofino testbed; processing delay and localization accuracy	SRv6 network; lightweight fine-grained telemetry
FDSR-INT [66]	SRv6-guided probing; dual-bitmap under MTU limits	Space–air–ground simulation; bandwidth and path overhead	Heterogeneous dynamic network; on-demand telemetry

which bounds the total telemetry items carried by each MF to prevent oversized probes, where λ_f is a configurable MF telemetry budget.

$$\omega_{ij} \geq R^s_{ij}, \quad \forall s \in \mathcal{S},\ \forall (i,j) \in \tilde{E}, \tag{29}$$

ensuring that every physical link used by an SFC is activated for monitoring, so that SFC-relevant resources are always covered by at least one MF.

$$\tau_f \leq \tau_{ij} + M\,(1 - \phi^f_{ij}), \quad \forall f \in \mathcal{F},\ \forall (i,j) \in \tilde{E}, \tag{30}$$

enforces SLA-driven probing frequency consistency: if MF f traverses link (i,j), then its probing interval τ_f must be no larger than the link's required bound τ_{ij}. M is a sufficiently large constant.

Deep Protocol Fusion and Scenario Customization. Beyond generic logic, recent advancements strive for deep integration with underlying network protocols and specific physical scenarios. SFANT [64] targets the Industrial Internet by fusing P4-based INT with SRv6. It adopts a three-layer architecture where the control plane plans minimum-cost paths using BFS or DFS, while the data plane executes fine-grained collection along SRv6 tunnels, ensuring low latency and high reliability without interfering with user flows. This effectively instantiates $J(P)$ with industrial-grade delay and reliability constraints.

To further eliminate the encapsulation overhead in SRv6 environments, INToSR [65] proposes a protocol-native monitoring approach. Telemetry instructions are embedded directly into the arguments fields of SRv6 segment identifiers and collected in the segment routing header (SRH) tail. This design leverages native endpoint behaviors to achieve zero-additional-header monitoring, relaxing $C_{\mathrm{MTU}}(P)$ while preserving telemetry granularity.

Finally, for highly dynamic environments like aeronautical computing networks, FDSR-INT [66] introduces a scenario-adaptive mechanism. It models the probe path planning as a Traveling Salesman Problem to optimize the traversal order of aerial nodes based on delay and bandwidth costs. To comply with the strict MTU constraints in air-to-ground links, it combines tabu search with a greedy path cutting strategy, ensuring packet feasibility while maximizing information gain. From the unified perspective, FDSR-INT tailors both the objective and constraint set to the aeronautical context, balanc-

ing ℓ_p, b_p, and $C_{\mathrm{MTU}}(P)$.

Application-aware AINTO has evolved from satisfying static coverage to managing complex logical dependencies and finally to deep architectural integration. Early schemes mainly translated application intents into task-specific probe configurations, whereas later designs progressively incorporated dependency-aware reconfiguration, protocol-native integration, and scenario-specific adaptation. This evolution reflects a trend towards intent-based telemetry, where orchestration logic is not only aware of what to measure, but also of how measurement interacts with underlying protocols and physical constraints, all within the optimization framework of Equation (6)–Equation (8). This evolution also reveals several clear trade-offs. Requirement-driven orchestration improves monitoring relevance and avoids indiscriminate network-wide probing, but it usually relies on stronger application assumptions and more complex intent translation logic [62, 32]. Deep integration with service chains, SRv6, or scenario-specific architectures improves protocol awareness and deployment fit, but it also increases coupling between telemetry design and the underlying system or environment [63, 11, 64]. Table 10 compares representative application-aware AINTO schemes, highlighting how monitoring intents are translated into probe configurations and how each approach integrates with protocols and scenarios. Table 11 further summarizes their key assumptions, evaluation settings, and deployment conditions. Together, these two tables show that the design of application-aware AINTO is fundamentally a trade-off between monitoring specificity and orchestration generality: telemetry becomes more efficient and relevant when it is tailored to application needs, but this often comes at the cost of stronger workload dependence, tighter protocol coupling, or reduced portability across scenarios. Accordingly, application-aware AINTO is particularly suitable for service- and scenario-specific environments, such as SFC/NFV deployments, industrial networks, SRv6-based programmable settings, and other specialized systems where monitoring objectives are tightly coupled with application logic and protocol behavior.

3.3. Method-based AINTO

Section 3.2 adopts a task-driven perspective and reviews existing AINTO research according to different telemetry objectives. Complementing this view, this section examines AINTO from a methodological perspective, focusing on how different approaches solve the generic optimization problem defined in Equation (6) under the system-level constraints in Equation (8).

Based on the classification of AINTO schemes introduced in Section 3.2, AINTO methodologies can be grouped into three major categories: graph-based approaches, optimization-based approaches, and machine learning-based approaches. Table 12 presents a high-level comparison of these three categories, summarizing their typical problems, strengths, and limitations. The following subsections focus on the algorithmic characteristics rather than re-describing individual systems in detail.

3.3.1. Graph-based Approaches

Graph theory provides the fundamental abstraction for AINTO, where the network is modeled as a graph $G = (V, E)$, and orchestration is reduced to classical traversal and covering problems. As outlined in Section 3.2, many coverage-aware and performance-aware schemes first construct a feasible path set P on this graph and then refine it according to Equation (6). From a methodological viewpoint, graph-based approaches mainly contribute topological theoretical foundations, such as how to achieve full coverage with minimal overlap or how to decompose a large graph into tractable subproblems.

A first line of work relies on Eulerian abstractions. Representative schemes such as NetVision [15] and INT-Path [18] reduce probe planning to the CPP, using Hierholzer-style traversals and graph augmentation to construct Eulerian trails. In non-Eulerian topologies, techniques such as MWPM are used to pair odd-degree vertices, guaranteeing minimal additional traversals but introduces $O(N^3)$ complexity. A second line of work focuses on graph decomposition and partitioning. Path decomposition methods combine Eulerian traversal with DP-based segmentation to ensure that each segment satisfies constraints such as $C_{\mathrm{MTU}}(P)$, INT-Segment [42] and MTU-Adaptive [43]. Topology partitioning methods divide G into weakly coupled subgraphs, enabling parallel planning and localized re-optimization, for example CFINT [45], GP-INT [46], and INT-Partition [44]. Recent works, such as INT-react [34] and INT-Source [53], further reduces orchestration complexity by enabling linear-time reconstruction and demonstrating how loop-free BFS-style traversals can be implemented directly in the data plane.

Overall, graph-based approaches are attractive due to their intuitive modeling and provable bounds on path length and probe count. However, they are primarily designed for static graphs with simple edge costs. When Equation (8) involves time-varying, or application-specific constraints, pure graph algorithms often require frequent global recomputation or heavy problem-specific engineering. This limitation motivates more expressive optimization-based techniques.

3.3.2. Optimization-based Approaches

Optimization-based approaches explicitly formulate AINTO as constrained optimization problem in which the orchestration plane seeks a path set P that minimizes $J(P)$ in Equation (6) while respecting the resource tuple in Equation (8). Compared with purely graph-theoretic methods, this family emphasizes formal modeling of trade-offs between coverage, latency, overhead, and resilience, at the cost of higher computational complexity.

At the modeling level, MILP are widely used to capture resource allocation and multi-objective trade-offs. Requirement-driven schemes such as P^{2}INT [62], DyPro [32], IntOpt [63], and SR-INT Orchestration [56] build explicit optimization models over Equation (6), while ATINT [54] further incorporates k-redundancy to guarantee robustness. These formulations provide accurate baselines for the generic AINTO problem but become impractical for online orchestration as

Table 12
Comparison of AINTO Research Methodologies

Methodology	Problems Solved	Key Advantage	Key Limitation
Graph-based	Addresses static path planning and full-network coverage.	Intuitive modeling with solid theoretical foundations.	High computational complexity and difficulty in handling complex constraints.
Optimization-based	Solves resource allocation and multi-objective trade-off problems.	Capable of formally defining and solving complex constrained problems.	NP-hard problem solving is difficult and may easily fall into local optima.
Machine Learning-based	Handles adaptive decision-making in dynamic environments.	Strong adaptability and does not require precise mathematical modeling.	High training cost and limited model interpretability.

network scale and constraint complexity increase. DP is employed in problems with optimal substructure, such as MTU-constrained path segmentation in INT-Segment [42] and MTU-Adaptive [43]. However, DP similarly suffers from exponential growth when applied to global planning.

To improve scalability, many works adopt heuristic and meta-heuristic approximations. Domain-specific heuristics consider particular network properties, for example, INT-balance [47] and SRv6-ALINT [48] rebalance path lengths using break-and-reconnect strategy or DFS-stitching, Patcher [52] and OpenINT [61] perform localized repair and incremental path search to handle failures or dynamic metadata requirements. In parallel, generic meta-heuristics are used to explore complex objective landscapes. Representative examples include bandwidth-oriented planning in AINTO [57], centrality-guided probing in Probe-Optimizer [60], multi-objective heuristics in INT-LLPP [58] and IntOpt [63], and AoI-based or scenario-aware planning in Freshness-INT [51], Cache-INT [59], and FDSR-INT [66].

The key advantage of optimization-based approaches is their expressiveness, by explicitly instantiating the objective in Equation (6) and the constraint set in Equation (8), they can incorporate fine-grained requirements under hard resource budgets. However, this fidelity incurs substantial runtime cost. Solving MILP or performing meta-heuristic search typically involves iterative exploration and repeated feasibility checks, and solutions often need to be recomputed when link conditions or monitoring demands change. As a result, these solvers may struggle to keep pace with rapid, sub-second dynamics in operational networks.

3.3.3. DRL-based Approaches

Graph-based and optimization-based approaches demonstrate strong modeling and solving capabilities on small-scale networks, where the topology and flows dynamics can be captured with manageable complexity. However, in large-scale networks with dynamic and time-varying topologies, their effectiveness is often limited by the exploding decision space and the need for rapid reconfiguration. In this setting, DRL provides a practical avenue for addressing large-scale in-band network telemetry orchestration.

DRL-based AINTO methods treat orchestration as an online control problem. As telemetry streams in, a learned agent updates the probing configuration to track changing network conditions. Concretely, the agent observes a compact state that summarizes recent telemetry signals and remaining headroom of key budgets, and then outputs actions that modify the probing plan, such as selecting probe paths and adjusting probing rates. A key challenge in INT orchestration is that bandwidth, device resources, and freshness targets are often hard operational constraints rather than soft preferences. Accordingly, practical DRL designs typically combine reward design with explicit feasibility handling, for example by restricting the action space to admissible choices or by coupling the policy with lightweight constraint checks. This allows the agent to reduce the orchestration cost $J(P)$ while respecting the system-level constraints in Equation (8). The policy is usually parameterized by a deep neural network and trained with actor–critic style algorithms, enabling near-constant-time inference at runtime. Representative examples include AdapINT [35], which learns probe dispatching policies for dynamic networks using an encoder–decoder actor–critic architecture, and Cache-INT [59], which applies DRL to jointly optimize probing path selection and in-network caching.

Compared with optimization-based solvers, DRL agents can react to network changes with low latency and support continuous adaptation once trained. Despite these advantages, DRL-based approaches introduce practical challenges in operational networks. Generalization is not guaranteed, and policies trained under one topology or flows regime may degrade when conditions shift; training can also be expensive and may incur long cold-start phases before a usable policy emerges. Moreover, the black-box nature of deep models limits interpretability and complicates constraint assurance, motivating hybrid designs that combine learned inference with lightweight feasibility checks or optimization layers.

3.4. Summary and Insights on AINTO

Synthesizing the task-driven review in Section 3.2 and the method-based discussion in Section 3.3, the evolution of AINTO can be summarized along two main trajectories, corresponding to how the objectives have evolved and how

the solution approaches have progressed.

Early studies largely target broad coverage of the monitoring scope under a quasi-static topology assumption, while timeliness or freshness is often treated as secondary. Subsequent performance-aware designs elevate timeliness to a first-class concern by explicitly enforcing freshness constraints and shaping probe paths to control latency. More recent overhead-optimized and application-aware works further move from blind sensing to intent-driven monitoring, incorporating bandwidth and processing costs into the utility objective and translating application intent into a tailored monitoring scope and budget allocation under shared system constraints.

From the other perspective, the key tension in AINTO has shifted from whether the orchestration problem can be modeled and solved to whether decisions can be made continously in real networks. Early studies mainly used graph-based and optimization-based to instantiate the objectives and constraints in Equation (6)–Equation (8), which leads to controllable orchestration results. However, when the topology state, link conditions, or monitoring demands change frequently, these methods often require repeated solving. As a result, decision latency may not match network dynamics. To address this issue, recent work introduces DLR-based approaches, trading offline training for near-constant-time online inference. This enables rapid adaptation of probing paths and rates, but it also brings engineering challenges such as training cost and generalization.

In general, AINTO enables fine-grained, intent-aware, and controllable telemetry but inevitably incurs extra bandwidth overhead and processing overhead. In the next section, we discuss a complementary INTO paradigm that avoid injecting additional flows.

4. Passive In-band Network Telemetry Orchestration

Unlike AINTO, which focuses on probe-path planning and scheduling via dedicated probes, PINTO piggybacks telemetry metadata on existing user flows. This shift introduces a fundamental dependency, as visibility is constrained by the spatiotemporal distribution of user traffic. Fig. 6 illustrates this intuition. The orchestration plane selects a subset of user flows and instruments their packets such that telemetry is collected only when and where these selected flows traverse INT-capable devices and reach the sink.

From the perspective of the generic optimization model in Eq. (6) and the system constraints in Eq. (8), PINTO replaces the probe path set P with a subset of user flows F and an associated sampling or reduction strategy S, while remaining subject to the same bandwidth, MTU, and processing constraints. Consequently, the orchestration challenge shifts from path planning to opportunity management, namely how to exploit available flow opportunities to maximize monitoring utility without violating strict resource budgets and packet-size limits.

This section systematically reviews PINTO strategies from two complementary perspectives. Section 4.1 focuses on macroscopic flow selection, discussing how to identify an effective subset of flows to achieve coverage and timeliness goals using optimization-based formulations. Section 4.2 shifts to microscopic data sampling and reduction, analyzing data-driven techniques such as probabilistic sampling and delta encoding that minimize per-packet overhead to fit within the MTU.

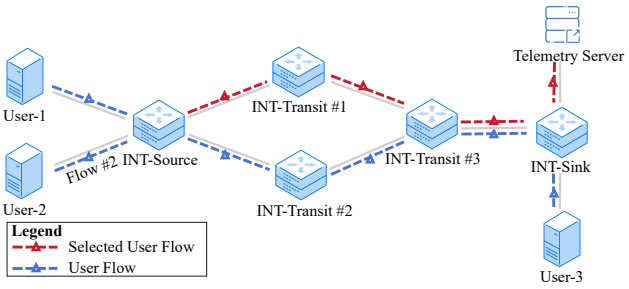


Figure 6: Workflow of PINTO.

4.1. Optimization-based Flow Selection

At the macroscopic level, PINTO can be viewed as a resource allocation problem, namely, given a limited budget of bandwidth and processing capacity, how to select a subset of user flows to maximize information utility. Unlike AINTO, which actively injects dedicated probe packets, PINTO must operate within the constraints imposed by existing user flows patterns. Optimization-based approaches formulate this challenge as a mathematical programming problem, evolving from static capacity management to dynamic and dependency-aware orchestration.

Static Trade-offs. Early PINTO schemes typically treat flow selection as a static optimization problem that balances monitoring coverage against bandwidth cost and freshness. These approaches assume quasi-stationary flows and aim to compute a single configuration that optimally trades off these metrics under a fixed capacity budget. Marques *et al.* [67] present an early work that formalizes PINTO flow selection as integer linear programming (ILP). Let $\mathcal{F}$ denote the set of candidate user flows, and let $\mathcal{I}$ denote the set of interfaces with nonzero telemetry demand. Each flow $f \in \mathcal{F}$ traverses a fixed set of interfaces $\rho(f) \subseteq \mathcal{I}$. For each $i \in \mathcal{I}$, δ_i is the number of telemetry items required at i, and κ_f is the telemetry capacity that flow f can carry per reporting round. Binary variable $x_{i,f}$ indicates whether interface i is assigned to flow f, and y_f indicates whether flow f is telemetry-active. INTO-concentrate minimizes the number of telemetry-active flows:

$$\min \ \phi_c = \sum_{f \in \mathcal{F}} y_f, \tag{31}$$

subject to the exact-coverage constraint:

$$\sum_{f \in \mathcal{F}:\ i \in \rho(f)} x_{i,f} = 1, \forall i \in \mathcal{I} : \delta_i > 0, \tag{32}$$

and the per-flow capacity constrain:

$$\sum_{i\in\rho(f)} x_{i,f}\,\delta_i \le y_f\,\kappa_f, \forall f \in \mathcal{F}. \tag{33}$$

By contrast, INTO-balance minimizes the maximum telemetry load assigned to any single flow to improve freshness. Let k denote this maximum load:

$$\min\ \phi_b = k. \tag{34}$$

It keeps the same coverage constraint (32), while enforcing both per-flow feasibility:

$$\sum_{i\in\rho(f)} x_{i,f}\,\delta_i \le \kappa_f, \forall f \in \mathcal{F}, \tag{35}$$

and load-balancing:

$$\sum_{i\in\rho(f)} x_{i,f}\,\delta_i \le k, \forall f \in \mathcal{F}. \tag{36}$$

In both formulations, $x_{i,f} \in \{0,1\}$ and $y_f \in \{0,1\}$, with $k \ge 0$.

Building on these, Zhang *et al.* [68] extend PINTO flow selection to a multi-objective optimization framework, explicitly modeling the conflict between intrusion and freshness. By employing non-dominated sorting genetic algorithm II (NSGA-II), their approach generates a set of pareto-optimal solutions, allowing operators to flexibly navigate the trade-off curve according to varying network conditions. Similarly, INT-Selection [69] targets port-level full coverage to address spatial imbalance, where hot ports consume all resources. It utilizes DP for small-scale networks and a greedy heuristic for large scales to select the minimal flow subset that traverses all active ports, ensuring zero blind spots with minimal overhead. While these designs clearly expose the fundamental coverage, intrusion, and freshness trade-offs and provide operators with tunable knobs, they are offline and cannot react to rapid flow bursts or long-term workload shifts.

Dynamic Closed-loops. To overcome the limitations of static planning, a second line of work embeds flow selection into closed-loop control, allowing telemetry policies to evolve with flows dynamics rather than being fixed a priori. Sel-INT [70] introduces a dual-layer closed-loop orchestration for time-varying traffic. At the macro level, it solves a global planning problem that balances monitoring gain against INT-induced bandwidth overhead. Using $\pi_{k,v_s,m} \in \{0,1\}$ to denote whether flow f_k collects telemetry item m at switch v_s, the objective can be compactly written as:

$$\begin{aligned}\max_{\{\pi_{k,v_s,m}\}}\ \Phi = w_g \sum_{v_s}\sum_{m}\sum_{k} \sigma_{k,v_s}\,\pi_{k,v_s,m}\,\eta_{v_s,m} \\ -\ w_b \sum_{v_s}\sum_{m}\sum_{k} \sigma_{k,v_s}\,\pi_{k,v_s,m}\,s_m \frac{1}{t^m_{v_s}} h_{k,v_s},\end{aligned} \tag{37}$$

where $\eta_{v_s,m}$ is the information gain, s_m is the metadata size, $t^m_{v_s}$ is the monitoring period, and h_{k,v_s} is the remaining hop count after v_s.

At the micro level, Sel-INT uses a long short-term memory (LSTM) predictor to estimate future bandwidth peaks $\hat{b}_{k,\tau}$ and proactively adjusts the sampling ratio $r_{k,\tau}$ before congestion:

$$r_{k,\tau} = f\left(\frac{\min\left(\kappa\cdot\max(B_{\text{th}} - \hat{b}_{k,\tau}, 0),\ \hat{B}_{\text{INT}}\right)}{B_e},\ p_{k,\tau}\right), \tag{38}$$

where B_{th} is a high-load threshold, $\hat{B}_{\text{INT}}$ caps the INT budget, and $\kappa \in (0,1]$ controls the monitoring aggressiveness.

Taking adaptivity further, D-INTO [71] formulates orchestration as a long-term stochastic optimization and uses Lyapunov optimization to turn the stability requirement into an online control problem. It maintains a virtual queue $Q(t)$ to track the stability debt induced by reconfiguration between consecutive time slots:

$$Q(t+1) = \max\{Q(t) + \Delta(t) - \Delta_{\text{avg}},\ 0\}, \tag{39}$$

where $\Delta(t)$ measures configuration changes between slots and Δ_{avg} is the long-term stability budget. Using the drift-plus-penalty principle, D-INTO decomposes the long-term objective into per-slot decisions that minimizes a stability-accuracy trade-off as

$$\min_{S(t)}\ Q(t)\,\Delta(t) - \nu\, I(t), \tag{40}$$

subject to the per-slot feasibility constraints. Here, $S(t)$ denotes the orchestration decision at time slot t, $I(t)$ is the instantaneous information gain, and $\nu \ge 0$ balances accuracy versus stability, effectively adding an explicit time-coupled reconfiguration term on top of the static cost model.

Dependencies and Rerouting. More recent works recognize that telemetry requirements often involve spatial and temporal dependencies, and that flow selection may need to be combined with routing decisions to fully satisfy these constraints. Hohemberger *et al.* [72] introduce a dependency-aware model that enforces spatial and temporal dependencies using a randomized constructive heuristic to satisfy these strict logic constraints. Furthermore, due to the fact that shortest-path routing often bypasses critical monitoring targets, INTE [73] extends the decision space to joint flow selection and routing optimization. INTE strategically reroutes a subset of flows via non-shortest paths to improve observability, accepting a marginal increase in latency in exchange. Let $G = (V, E)$ be the network graph and let $\mathcal{F}$ denote the set of flows, where $\mathcal{F}_{\text{opt}} \subseteq \mathcal{F}$ are flows whose routes can be optimized. Binary variables $x_{f,i,j} \in \{0,1\}$ indicating whether flow f traverses directed link $(i,j) \in E$. It maximizes the satisfaction of monitoring requirements while bounding detour length:

$$\max \sum_{m\in\mathcal{M}} \left(\sum_{i\in V}\sum_{r\in\mathcal{R}^{\text{sp}}_m} s_{m,i,r} + \sum_{r\in\mathcal{R}^{\text{tp}}_m} t_{m,r}\right), \tag{41}$$

subject to standard flow-conservation constraints:

$$\sum_{j:(i,j)\in E} x_{f,i,j} - \sum_{j:(j,i)\in E} x_{f,j,i} = \begin{cases} 1, & i = s_f, \\ -1, & i = e_f, \\ 0, & \text{otherwise}, \end{cases} \tag{42}$$

and a hop-count bound to control latency inflation

$$\sum_{(i,j)\in E} x_{f,i,j} \le H_{\max}, \forall f \in \mathcal{F}_{\text{opt}}. \tag{43}$$

Together, these approaches move PINTO towards intent- and dependency-aware orchestration, where telemetry decisions are aligned with application structure and routing flexibility. However, most remain offline or batch-optimized and only partially address real-time adaptation.

Overall, optimization-based flow selection in PINTO has evolved along a clear trajectory, from static formulations that clarify the coverage–overhead–freshness trade-offs, through dynamic closed-loop controllers that stabilize congestion and adapt to flow dynamics, to dependency-aware designs that begin to encode application intent and routing flexibility. Early schemes mainly optimized a fixed subset of user flows under static capacity assumptions, whereas later designs progressively incorporated online adaptation, long-term stability control, and dependency-aware rerouting. This progression highlights how the same bandwidth and stability constraints introduced in Section 2.3 can be addressed at increasingly higher semantic levels. It also reveals several clear trade-offs. Static flow selection exposes the basic balance among coverage, intrusion, and freshness, but it is less responsive to bursty traffic or workload evolution [67, 68, 69]. Dynamic closed-loop designs improve adaptability and congestion awareness, but they require richer runtime information, prediction support, or more complex online control mechanisms [70, 71]. Dependency-aware and rerouting-based approaches further improve monitoring expressiveness and observability, but they often increase orchestration complexity and may introduce additional routing or latency costs [72, 73]. Table 13 summarizes representative schemes along these dimensions, and Table 14 further summarizes the traffic assumptions, evaluation settings, and deployment conditions of selected schemes.Together, these two tables show that the design of optimization-based PINTO flow selection is fundamentally a trade-off among monitoring coverage, telemetry intrusion, adaptability, and decision complexity: stronger observability and better runtime responsiveness are desirable, but they usually require more sophisticated control logic and, in some cases, greater flexibility in traffic steering. Accordingly, optimization-based PINTO flow selection is particularly suitable for production environments with abundant user traffic, such as data-center and cloud networks, where low-intrusion continuous monitoring is desired but telemetry decisions must still adapt to traffic dynamics and operational constraints.

4.2. Data-driven Sampling and Reduction

The flow-selection mechanisms discussed in Section 4.1 determine which user flows carry telemetry, but do not directly control the amount of metadata attached to each packet. In practice, the telemetry overhead added to individual packets is tightly constrained by MTU and bandwidth limits. This subsection therefore shifts to microscopic, data plane techniques that reduce the per-packet telemetry overhead. These data-driven schemes exploit statistical properties of network states and telemetry measurements, such as temporal stability, inter-metric value correlation, and sparsity of flows matrices, to adapt when measurements are collected and how they are encoded or compressed. Concretely, we categorize existing work into three families: probabilistic approximation, exploiting spatiotemporal redundancy, and adaptive-granularity.

Lightweight Compression. When exact per-packet visibility is prohibitively expensive, lightweight INT schemes offer a more deployment-friendly trade-off between telemetry compactness and monitoring fidelity. PINT [16] is a representative approach in this direction that employs a global hashing framework. Instead of accumulating hop-by-hop metadata, it encodes telemetry information into a fixed-size digest, ensuring constant overhead regardless of path length while enabling aggregation queries with bounded error rates. Beyond PINT, recent work has made the lightweight INT design space more explicit by jointly considering deterministic and probabilistic P4-enabled mechanisms [74]. In particular, DLINT reduces transmission overhead through deterministic per-flow aggregation, spreading telemetry values across packets of the same flow with switch coordination and per-flow telemetry states, whereas PLINT adopts a probabilistic strategy based on reservoir sampling to insert telemetry values with equal probability without explicit switch coordination. These schemes further clarify the trade-offs within lightweight telemetry design: probabilistic mechanisms such as PINT and PLINT achieve higher flexibility and more efficient header-space utilization, but rely more on approximation or multi-packet inference, while deterministic designs such as DLINT can better preserve path-trace semantics and continuous path reconstruction, at the cost of additional switch-side coordination and per-flow state management. Similarly, INT-Label [75] delegates decision-making to the data plane through a fully distributed labeling mechanism. By combining interval-based coverage with probabilistic aggregation, it allows devices to autonomously trigger telemetry insertion only when necessary.

Exploiting SpatioTemporal Redundancy. The inherent strong spatiotemporal correlations inherent in network states that there is a lot of redundancy in the telemetry data. Taking advantage of this redundancy is key to achieving efficient compression by eliminating repetitive data across different dimensions. For instance, DeltaINT [76] exploits on temporal stability by maintaining the last reported values as references on the data plane. A threshold-based trigger reports telemetry data only when the real-time values deviate significantly from these references, effectively filtering out redundant updates. OffsetINT [77] further exploits value

Table 13
Comparison of Optimization-based Flow Selection Strategies

Scheme	Year	Optimization Goal	Key Methodology	Granularity		Adaptivity	
				Flow	Port	Stat	Dyn
Marques *et al.* [67]	2019	Min active flows and reduce per-flow telemetry burden	ILP and Greedy Heuristic	✓		✓	
Hohemberger *et al.* [72]	2020	Satisfy monitoring demands by enforcing spatial and temporal dependencies	Randomized Heuristic	✓		✓	
Zhang *et al.* [68]	2021	Trade-off Intrusion and Freshness	Non-dominated Sorting Genetic Algorithm II	✓		✓	
Sel-INT [70]	2022	Anticipate flows surges and proactively throttle sampling rates before congestion	LSTM and Lagrangian	✓			✓
INTE [73]	2024	Reroute partial flows via non-shortest paths to maximize essential node coverage	Joint Routing Optimization	✓		✓	
INT-Selection [69]	2025	Port-level full coverage by selecting the minimal flow subset that traverses all ports	Dynamic Programming and Greedy Heuristic		✓	✓	
D-INTO [71]	2025	Dynamically balance instantaneous measurement accuracy and stability	Lyapunov optimization	✓			✓

Flow/Port: Selection Granularity; **Stat/Dyn:** Static Configuration and Dynamic Adjustment.

Table 14
Preconditions, Evaluation Settings, and Deployment Conditions of Selected Optimization-based Flow Selection PINTO Schemes

Scheme	Preconditions	Evaluation Setting	Deployment Condition
Marques *et al.* [67]	Quasi-stationary flows; fixed routes; limited telemetry capacity	Real WANs; coverage efficiency and load balancing	Managed network; selected production flows
Sel-INT [70]	Observable flow statistics; forecastable QoS demand	Real OVS-POF/ONOS testbed; adaptive selection	SDN control; selective telemetry updates
INTE [73]	Reroutable production flows; flexible path assignment	CPLEX evaluation; routing flexibility and monitoring demand	Central orchestration; routing flexibility
INT-Selection [69]	Selectable flow subset; MTU-bounded packet size	Existing baselines; bandwidth and port coverage	Centralized passive INT; active-port monitoring
D-INTO [71]	Time-varying states; slotted long-term orchestration	Simulation; accuracy, stability, adaptability	Dynamic network; online stability control

correlation by encoding metrics as small offsets relative to a base value, using a compact bitmap format to achieve high compression ratios. Taking a signal processing perspective, SaR4IO [78] treats INT counters as linear aggregates of flow intensities. Let the network be a directed graph $G = (V, E)$, let $\mathbf{x} \in \mathbb{R}_+^K$ denote the traffic-matrix vector, and let $\mathbf{y} \in \mathbb{R}_+^M$ collect the available INT counters. SaR4IO models the measurements as:

$$\mathbf{y} = \mathbf{H}\mathbf{x}, \qquad \mathbf{H} \triangleq \begin{bmatrix} \mathbf{R} \\ \mathbf{B} \end{bmatrix}, \tag{44}$$

where $\mathbf{R}$ is the routing matrix and $\mathbf{B}$ captures segment-level aggregation relationships. Given a sampled subset S of counters, define the diagonal selector $\mathbf{D}_S = \mathrm{diag}(\mathbf{1}_S)$ and $\mathbf{y}_S = \mathbf{D}_S\mathbf{y}$. The recovery step then estimates $\mathbf{x}$ by solving a non-negative sparse reconstruction problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\geq 0} \left\|\mathbf{D}_S\left(\mathbf{y} - \mathbf{H}\mathbf{x}\right)\right\|_2^2 + \lambda\|\mathbf{x}\|_1, \tag{45}$$

which enables accurate telemetry inference while collecting only a small fraction of counters.

Adaptive Granularity. To overcome the rigidity of static metadata configurations, recent works introduce on-demand and adaptive telemetry granularity. FINT [79] proposes a dynamic triple-bitmap mechanism that allows switches to greedily insert only the most valuable telemetry fields that fits within the remaining MTU budget. Event-Sample [80] adopts an event-driven strategy, triggering reports solely upon detecting flow-level anomalies, achieving near-zero overhead during stable periods. Furthermore, Hohemberger *et al.* [81] integrate ML to estimate telemetry variability and prioritize sampling accordingly. They start from a baseline MILP that orchestrates which telemetry items should be carried by which flows under packet-capacity constraints. Let $\mathcal{F}$ denote the set of flows and let $\mathcal{T}_i$ be the set of telemetry items

available at node $i \in V$. The binary variable $y_{i,t,f}$ indicates whether flow $f \in \mathcal{F}$ carries item $t \in \mathcal{T}_i$ collected at node i. Let $\mathcal{M}$ be the set of monitoring applications, and let $\mathcal{R}_m^{\text{sp}}$ and $\mathcal{R}_m^{\text{tp}}$ denote the spatial and temporal dependency groups required by $m \in \mathcal{M}$, where each group r consists of a set of telemetry items that must be collected jointly. Using binary indicators $\bar{s}_{m,i,r}$ and $\bar{t}_{m,r}$ to denote whether spatial and temporal dependencies are satisfied, respectively, the baseline objective is:

$$\max \sum_{m\in\mathcal{M}} \left(\sum_{i\in V} \sum_{r\in\mathcal{R}_m^{\text{sp}}} \bar{s}_{m,i,r} + \sum_{r\in\mathcal{R}_m^{\text{tp}}} \bar{t}_{m,r} \right), \tag{46}$$

subject to the per-flow packet-capacity constraint

$$\sum_{i\in\text{path}(f)} \sum_{t\in\mathcal{T}_i} y_{i,t,f}\, S(t) \le K_f, \forall f \in \mathcal{F}, \tag{47}$$

and a non-redundancy constraint preventing the same item from being sampled by multiple flows:

$$\sum_{f\in\mathcal{F}} y_{i,t,f} \le 1, \forall i \in V,\ \forall t \in \mathcal{T}_i. \tag{48}$$

A dispersion index for each telemetry item t is defined as:

$$\text{DI}(t) = \frac{\text{Var}(t)}{\mathbb{E}[t]}, \tag{49}$$

so that highly volatile items have larger DI(t). Based on this index, the controller applies a fading rule to decide how long a telemetry tuple $\mathbf{q}$ remains active for orchestration:

$$T(\mathbf{q}) = \begin{cases} W, & \text{DI}(\mathbf{q}) \le 1, \\ W(1-\rho_d), & \text{DI}(\mathbf{q}) > 1, \end{cases} \tag{50}$$

where $\rho_d \in [0,1]$ controls how aggressively over-dispersed items are revisited, thereby prioritizing the sampling of highly variable telemetry under packet-capacity constraints.

Table 15 provides a systematic comparison of these data-driven strategies, and Table 16 further summarizes their traffic assumptions, evaluation settings, and deployment conditions. Together, these two tables show that data-driven PINTO reduction has evolved from fixed-size probabilistic compression to spatiotemporal redundancy exploitation and finally to adaptive granularity control. By shifting from indiscriminate collection to intelligent reduction, through probabilistic sketches, signal reconstruction, or adaptive granularity, these schemes effectively decouple monitoring precision from packet overhead. This evolution reveals several clear trade-offs. Probabilistic summaries and compact encodings greatly reduce packet overhead, but they usually replace exact per-hop visibility with bounded approximation or aggregate inference [16, 75]. Redundancy-aware schemes improve compression efficiency by exploiting temporal or spatial correlation, but they depend more strongly on traffic stability, threshold design, or reconstruction assumptions [76, 77, 78]. Adaptive granularity designs further improve efficiency by inserting only high-value telemetry when needed, but they require more dynamic control logic or accurate estimates of telemetry variability [79, 80, 81]. Therefore, the design of data-driven PINTO reduction is fundamentally a trade-off between monitoring precision and telemetry compactness: lower packet overhead and better feasibility under bandwidth and MTU limits are desirable, but they often come at the cost of approximation, stronger data assumptions, or greater orchestration complexity. Accordingly, data-driven PINTO reduction is particularly suitable for large-scale continuous monitoring settings where telemetry must remain lightweight, such as production cloud networks and other bandwidth-sensitive environments that require low-intrusion observability over long periods. They serve as the microscopic foundation of PINTO, ensuring that even selected flows do not exceed physical bandwidth and MTU limits.

4.3. Summary and Insights on PINTO

As summarized in Table 13 and Table 15, the evolution of PINTO can be organized around two practical questions. The first concerns which flows should be instrumented. Early designs typically rely on static flow selection to expose the fundamental trade offs among coverage, overhead, and freshness. Later work moves toward closed loop control, enabling orchestration policies to adapt to traffic variability while maintaining stable behavior under fluctuating workloads. More recent schemes further incorporate application-level dependencies and may expand the decision space to include routing choices. This line of work treats flow selection as a primary control lever, reflecting the fact that PINTO is fundamentally constrained by the traffic matrix and can only observer what user flows naturally traverse the network.

The second question addresses how much information should be carried per packet. Instead of embedding complete raw telemetry, recent schemes explicitly acknowledge strict packet-size and bandwidth constraints and exploit statistical structure such as temporal stability, inter-metric correlation, and sparsity. The dominant trend is to reduce overhead through probabilistic digests, redundancy elimination via delta or offset encoding, signal reconstruction, and adaptive field granularity. These techniques provide a practical way to preserve useful visibility even when per-packet budgets are strict.

Taken together, these two directions transform PINTO from passive piggybacking mechanism into a controllable observation framework, in which operators can jointly regulate where telemetry is injected and how much and how compactly information is inserted into the packets.

5. Hybrid In-band Network Telemetry Orchestration

The preceding sections addressed AINTO and PINTO as distinct orchestration approaches. Specifically, Section 3 systematized AINTO under a generic optimization model, categorizing approaches by task and methodology. Section 4

Table 15
Comparison of Data-driven Sampling and Reduction Strategies

Scheme	Year	Goal	Key Methodology	Overhead Reduction Mechanism	Adaptivity
Hohenberger [81]	2019	Reduce irrelevant telemetry collection while maintaining acquisition accuracy under packet capacity constraints.	ML-based dispersion index priority.	Importance-aware prioritization.	Dyn
PINT [16]	2020	Bound per-packet telemetry overhead under a user-defined budget.	Global hash and probabilistic in-packet digest.	Probabilistic digest.	Stat
FINT [79]	2022	Balance INT flexibility with performance impact.	Triple-bitmap based dynamic field selection and Greedy insertion of high-value metadata.	Dynamic field selection under MTU budget.	Dyn
DeltaINT [76]	2023	Reduce INT bandwidth by avoiding redundant state reporting.	Threshold-based delta embedding.	Delta encoding-triggered reporting.	Dyn
OffsetINT [77]	2023	Achieve high accuracy and generality with low bandwidth overhead.	Reference-based offset encoding with recovery at the collector or end host.	Offset compression.	Dyn
SaR4IO [78]	2023	Reduce telemetry overhead via sampling while enabling accurate reconstruction.	Recover unseen values via reconstruction.	Sparse sampling + reconstruction.	Stat
Event-Sample [80]	2024	Reduce overhead by reporting only when events/anomalies occur.	Event-triggered telemetry reports.	Event-driven reporting.	Dyn
INT-Label [75]	2024	Lightweight network-wide telemetry with minor bandwidth overhead.	Probabilistic-based control with feedback to tune labeling frequency.	Distributed labeling and adaptive labeling frequency.	Dyn

Stat/Dyn: Static Configuration and Dynamic Adjustment.

elucidated how PINTO leverages user flows through flow selection and in-packet information reduction. Building on these foundations, this section presents HINTO, which coordinates AINTO and PINTO to satisfy application-specific monitoring objectives. In contrast to the more extensively developed AINTO and PINTO, existing HINTO studies remain relatively limited and are largely organized around specific application scenarios. Accordingly, instead of introducing a similarly fine-grained taxonomy, this section focuses on the common coordination patterns and core problem formulations that characterize current HINTO designs. Fig. 7 illustrates a typical hybrid scenario in which active and passive telemetry coexist within the network.

5.1. Motivation and Problem Definition

Relying solely on AINTO or PINTO exposes fundamental limitations. AINTO guaranties visibility but incurs high resource overhead and potential interference, whereas PINTO is resource-efficient but suffers from inevitable blind spots due to its dependence on user flows patterns. To bridge this gap, HINTO orchestrates both modalities as complementary tools, aiming to achieve comprehensive visibility and responsiveness with minimal operational costs.

Formally, in this paper we view HINTO as an orchestration problem over multiple telemetry modalities within a unified optimization framework. Accordingly, the decision variables are extended from a pure probe-path set P or a flow and sampling configuration (F, S) to a joint hybrid configuration $(P^{\text{act}}, F^{\text{pas}}, S^{\text{pas}})$. Here, P^{act} denotes active probe paths, F^{pas} denotes user flows carrying passive INT, and S^{pas} denotes data plane sampling or reduction strategies. Consequently, the optimization objective is generalized to a hybrid form $\mathcal{J}_{\text{hyb}}(P^{\text{act}}, F^{\text{pas}}, S^{\text{pas}})$, subject to the constraint structures defined in Section 2.3 and Section 3, now applied jointly to all telemetry flows. This formulation clarifies that HINTO is not a third, independent mechanism, but a joint optimization over shared objectives with an expanded decision space.

This unified formulation also provides a common lens to interpret existing hybrid designs: different systems instantiate HINTO by making different choices about how to couple $(P_{\text{act}}, F_{\text{pas}}, S_{\text{pas}})$ and how to allocate shared telemetry budgets between the two modalities. Although current HINTO

Table 16
Preconditions, Evaluation Settings, and Deployment Conditions of Representative Data-driven Sampling and Reduction PINTO Schemes

Scheme	Preconditions	Evaluation Setting	Deployment Condition
Hohenberger [81]	Partial item collection; learnable item importance	Prior heuristics; anomaly detection and visibility quality	Learning-based orchestration; production traffic
PINT [16]	Approximate telemetry; fixed per-packet bit budget	Real topologies/traffic; overhead and telemetry tasks	P4-capable devices; strict overhead control
FINT [79]	Runtime-adjustable tasks; MTU-bounded packet length	BMv2-based P4 simulation; FCT and bandwidth impact	Data-center network; MTU/performance constraints
DeltaINT [76]	Small adjacent state changes; significance-triggered embedding	Software simulation and P4/Tofino; bandwidth and accuracy	Stateful data plane; low-overhead telemetry
OffsetINT [77]	Small or correlated values; offset encoding	BMv2/Tofino; bandwidth reduction and accuracy	Accurate telemetry; lower per-packet overhead
SaR4IO [78]	Sparse target signals; reconstruction from partial observations	Preliminary analysis; reconstruction accuracy and overhead	SR-based network; controller-side recovery
Event-Sample [80]	Event/threshold-driven reporting	Simulation; overhead reduction and path tracing	Event-driven telemetry; low reporting overhead
INT-Label [75]	Distributed packet labeling; feedback-adjusted frequency	Software P4 switches; coverage and packet reduction	Lightweight telemetry; no centralized path planning

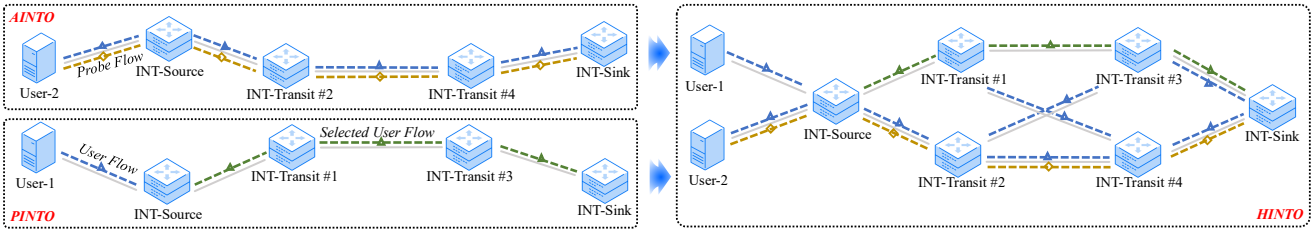


Figure 7: Illustration of HINTO.

deployments are still largely scenario-driven, they already exhibit several recurring coordination patterns. In the next subsection, we distill these patterns and illustrate them using representative systems.

5.2. Active–Passive Coordination Patterns in HINTO

Viewed through $(P_{\text{act}}, F_{\text{pas}}, S_{\text{pas}})$, existing HINTO systems can be grouped according to how they coordinate responsibilities between passive instrumentation and active probing. The most common pattern is passive-first coverage, where PINTO is configured to exploit user flows as much as possible, and AINTO is invoked to fill residual blind spots. Another recurring pattern is trigger-based verification, where PINTO performs continuous monitoring, and AINTO is dispatched only for on-demand confirmation and localization. We next discuss representative systems under these patterns.

Passive-first coverage. From the decision-vector viewpoint, this pattern primarily fixes $(F_{\text{pas}}, S_{\text{pas}})$ to maximize passive coverage, and then optimizes P_{act} to compensate for the remaining uncovered targets. Hawkeye [82] targets link coverage in wide-area networks by coupling proactive probing with passive piggybacking. It partitions the topology into a passive part and a proactive part based on historical flows information: links with sufficient service flows are monitored via PINTO, while the remaining links are covered by AINTO. Hawkeye then plans probe paths to traverse uncovered links while reducing overlap and bandwidth overhead. Patrol [83] follows a similar two-stage strategy in software-defined networks. In each telemetry epoch, it first configures PINTO to maximize coverage and telemetry collection, and then plans a small set of AINTO paths to cover the remaining blind spots. To limit redundant probing and bandwidth overhead, Patrol further optimizes active paths by shortening path length and constraining probe growth, which also mitigates MTU-related risks on long probing paths.

Trigger-based AINTO. From a decision-vector perspective, this pattern maintains $(F_{\text{pas}}, S_{\text{pas}})$ as the default configuration. It activates P_{act} only when passive telemetry indicates a potential anomaly. GrayINT [84] applies HINTO to gray failure detection and localization in fat-tree data center networks. It uses PINTO to infer feasible path informa-

tion from service flows and employs a timeout mechanism to flag potential failures. Since low flows on a path can also trigger timeouts, GrayINT dispatches AINTO probe packets for confirmation once the timeout condition is met. In this way, passive telemetry provides continuous monitoring, while active probes are reserved for on-demand verification and localization.

Together, Hawkeye, GrayINT, and Patrol illustrate that current HINTO designs often follow a maximize passive, then compensate with active principle. By coordinating responsibilities under shared budgets, HINTO can improve coverage and diagnostic efficiency compared with using only AINTO or only PINTO. At the same time, existing systems remain largely scenario-driven and do not yet provide general abstractions for joint budget allocation and multi-objective trade-offs, motivating further research on HINTO.

6. Future Opportunities and Challenges

Building on the unified model and taxonomy developed in Sections 2–5, this survey also exposes several persistent limitations in current INTO designs. First, both AINTO and PINTO are fundamentally constrained by system budgets, so each mode faces an inherent trade-off between visibility and overhead when used in isolation. Second, PINTO's observability is bounded by the spatiotemporal availability of user flows, leaving unavoidable blind spots, whereas AINTO can provide flexible coverage but may incur higher cost and interference. Third, although hybrid deployments are increasingly adopted in practice, existing HINTO systems remain largely scenario-driven and still lack general abstractions for joint budget allocation and multi-objective trade-offs at network scale.

These limitations point to three broader challenges for future INTO. First, the central systems problem remains how to improve observability without violating strict packet, bandwidth, and device-resource budgets. Second, as orchestration becomes increasingly adaptive, on-demand, and multi-modal, the difficulty shifts from solving isolated optimization problems to managing continuous coordination, recompilation, and uncertainty under dynamic conditions. Third, future INTO must go beyond efficiency alone and provide stronger guarantees of correctness, isolation, and trustworthiness in shared programmable network environments.

Motivated by these broader challenges, this section distills the key open problems and outlines future research directions toward scalable, multi-modal, and trustworthy INTO in operational networks. Fig. 8 summarizes these directions and organizes Section 6.1–6.6 around six themes.

6.1. Multi-modal INTO

Relying exclusively on a single telemetry mode is increasingly proving inadequate for the complexity of modern networks. AINTO collects telemetry by injecting dedicated probe packets, its visibility depends on probe spatiotemporal coverage and probing rate, potentially missing short-lived events unless probing is dense. Conversely, PINTO provides in-situ, flows-correlated evidence but consumes MTU

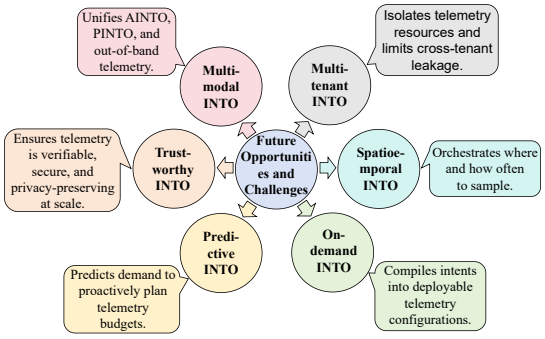


Figure 8: Future opportunities and challenges.

headroom and switch resources, and it remains blind in regions with sparse user flows. Although traditional out-of-band telemetry is lightweight, its lack of fine-grained visibility and causal context makes it difficult to pinpoint the root causes of failures [85]. These limitations are amplified in heterogeneous environments involving diverse domains, devices, and encapsulation protocols. This motivates a shift towards multi-modal telemetry fusion, in which the orchestration layer integrates AINTO, PINTO, and out-of-band telemetry. Such integration enables dynamic balancing between monitoring cost and coverage, offering robustness that single-mode telemetry cannot achieve. However, fusing multiple telemetry modes shifts complexity to the control plane, introducing new orchestration challenges, most notably conflict resolution when multiple applications demand different data granularities. Ultimately, addressing these challenges requires an intent-based orchestration framework that manages these telemetry modes as a unified resource pool, optimizing collection policies to satisfy global constraints [86].

6.2. Spatiotemporal INTO

Time and space are intrinsically coupled in INTO, serving as two key degrees of freedom. However, prevalent designs often implicitly decouple them by fixing one dimension to optimize the other. This separation severely limits diagnostic utility, especially in heterogeneous environments where resources are constrained. Spatially-biased orchestration often misses transient, short-lived phenomena due to insufficient temporal resolution [87], whereas temporally-biased orchestration fails to eliminate blind spots when observation points are restricted [85]. Therefore, future INTO research must shift towards spatiotemporal co-orchestration, treating time and space as complementary levers rather than isolated variables.

6.3. On-demand INTO

A key limitation of current on-demand telemetry is that application-driven telemetry often stops at the user-facing interface. Operators still manually tune probe scope, telemetry fields, and sampling policies, while the system optimizes a small set of fixed objectives rather than adapting to evolving application intent. As a result, intents expressed at different layers are difficult to unify, and conflicts are handled

implicitly instead of being detected, validated, and resolved with predictable outcomes. This mismatch becomes more severe under multi-tenant contention for MTU, bandwidth, and collector capacity, where what to measure and when to measure must be co-managed.

An open challenge is to build an intent compiler for INTO that translates telemetry intents into machine-checkable constraints and executable configurations, thereby reducing reliance on ad-hoc tuning. This direction aligns with intent-based networking idea of intent to validated realizations [31] and practical intent compilation in SDN controllers [88]. The compiler should also mediate multi-tenant conflicts via explicit arbitration policies and produce auditable outcomes that explain why an intent is only partially satisfied. It should retain provenance across recompilations to support post-incident debugging [89]. Within this compiler framework, large language models (LLMs), or domain-specific NetLLMs, may serve as a promising semantic interface for translating high-level operator intents expressed in natural language or SLA form into candidate telemetry configurations under low-level telemetry and P4 constraints. Such an LLM-enhanced compiler could further improve semantic alignment, constraint extraction, and conflict explanation. Moreover, the same framework could be extended to automated root-cause analysis by combining telemetry evidence with compiler provenance and configuration history, thereby enabling more explainable diagnosis of intent violations, partial satisfactions, and cross-intent conflicts.

6.4. Predictive INTO

A practical limitation of current telemetry orchestration is its largely reactive nature, telemetry is intensified only after a trigger is observed. Many performance pathologies are extremely short-lived and may disappear before configuration changes take effect. For example, high-resolution measurements in production data centers show that microbursts often last less than 1 ms, with the p90 durations of at most 200 µs, implying that turning on richer telemetry after detection may still miss the decisive evidence window [90]. Under tight MTU, bandwidth, and collector budgets, reactive mode switching or sparse sampling can therefore lead to either wasted overhead or missed root-cause signals.

An open challenge is how INTO can incorporate prediction to make orchestration proactive, allocating telemetry budgets and selecting targets ahead for high-risk periods, while remaining robust to prediction errors. Turning prediction into dependable orchestration also raises open issues, coping with uncertainty and concept drift, avoiding over-instrumentation under false alarms, and providing safe fallback behavior when predictions fail.

6.5. Multi-tenant INTO

Multi-tenant deployments introduce an additional challenge for INTO, particularly in cloud data centers where multiple tenants may share the same physical programmable switch. In such settings, telemetry orchestration must consider not only measurement efficiency and fidelity, but also how telemetry-related hardware resources are isolated and arbitrated across tenants. Without explicit tenant-aware control, aggressive telemetry policies from one tenant may consume a disproportionate share of shared resources, including SRAM/TCAM entries, pipeline stages, ALU budgets, and export bandwidth, thereby degrading the telemetry capabilities available to others [91, 92].

This challenge suggests that future INTO frameworks should move beyond global resource optimization and incorporate explicit isolation mechanisms. Examples include tenant-level quotas or partitions for telemetry-related table space, pipeline-stage budgets, and reporting resources, as well as admission and arbitration policies that determine when an intent can be safely realized under shared hardware constraints. Such mechanisms are particularly important in programmable data planes, where compile-time and run-time feasibility depend on fine-grained resource availability rather than bandwidth or PHV limits alone.

Multi-tenant INTO also raises security concerns. Telemetry outputs may inadvertently expose shared infrastructure states, contention patterns, or timing behaviors that enable cross-tenant inference. More broadly, shared low-level resources in programmable switches can create new side channels if telemetry visibility is not properly constrained [93]. Therefore, a promising direction is to develop isolation preserving INTO frameworks that combine tenant-aware resource accounting, leakage-aware telemetry policies, access control, and auditable arbitration, so that telemetry remains both effective and trustworthy in shared programmable networks.

6.6. Trustworthy INTO

A growing gap in current INTO systems is that they optimize what to measure, but often take the trustworthiness of the measurement pipeline for granted. In practice, INTO relies on frequently updated data plane telemetry programs and collection logic, where subtle bugs or mismatches can silently bias measurements and mislead orchestration decisions. Practical P4 verification tools show that automated, path-sensitive checking is feasible for programmable data planes, suggesting that correct telemetry of construction should become a first-class requirement rather than an afterthought [94].

Trustworthiness also encompasses security and privacy. Telemetry packets and reports may traverse shared infrastructure, making them targets for eavesdropping and tampering, meanwhile, rich INT signals can leak operational details. Prior work has explored protecting INT analytics with lightweight encryption and integrity certification, enabling collectors to authenticate received telemetry and reduce leakage risks [95]. At a broader scope, secure inter-domain INT proposals illustrate how authentication can be integrated into telemetry to enable cross-domain use cases [96]. A key open challenge is integrate these guarantees with dynamic INT orchestration. The system must remain efficient under high telemetry rates while preserving verifiable correctness, authenticated provenance, and privacy-aware reporting.

7. Summary

In-band Network Telemetry Orchestration (INTO) addresses the fundamental problem of jointly deciding what to measure and how to measure under practical constraints such as MTU headroom, bandwidth budgets, device resources, and freshness requirements.

This survey provided a unified and comprehensive view of the INTO design space by organizing existing work into three complementary orchestration paradigms: active INTO (AINTO), which injects dedicated probes for guaranteed coverage; passive INTO (PINTO), which piggybacks telemetry on user flows and relies on intelligent flow selection and in-packet reduction; and hybrid INTO (HINTO), which coordinates active and passive telemetry to exploit their respective strengths while mitigating their limitations. Across these paradigms, we established a common system perspective that reveals shared objectives, constraints, and trade-offs underlying seemingly disparate designs. Specifically, we (i) formalized INTO with a unified system model, (ii) derived unified optimization formulations for AINTO probe-path planning, PINTO flow selection and in-packet reduction, and their joint generalization for HINTO, and (iii) developed a structured taxonomy and comparative review summarized by representative tables. Building on this foundation, we identified open challenges and future research directions, including multi-modal telemetry fusion, spatiotemporal co-orchestration, intent-driven, predictive telemetry, multi-tenant isolation, and trustworthy telemetry pipelines.

CRediT authorship contribution statement

Yonghao Zhang: Conceptualization of this study, Writing - Original draft preparation. **Lizhuang Tan:** Project administration, Funding acquisition. **Yuyu Zhao:** Writing - Review & Editing. **Nguyen Van Tu:** Writing - Review & Editing. **Pilar Manzanares-Lopez:** Writing - Review & Editing. **Na Li:** Writing - Original draft preparation. **Huiling Shi:** Writing - Review & Editing. **Wei Zhang:** Project administration. **Peiying Zhang:** Writing - Review & Editing, Supervision. **Wei Su:** Writing - Review & Editing, Supervision. **James Won-Ki Hong:** Writing - Review & Editing, Supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this survey.

Acknowledgment

This work was supported in part by the research projects funded by the National Key R&D Program of China under Grant No.2024YFB2907000, the National Natural Science Foundation of China under Grant No.62402257, the Natural Science Foundation of Shandong Provincial under Grant No.ZR2023QF025 and No.ZR2024LZH006, the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) under Grant No.RS-2024-00392332, the Pilot Project for Integrated Innovation of Science, Education and Industry of Qilu University of Technology (Shandong Academy of Sciences) under Grant No.2025ZDZX01.

References

[1] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, T. Turletti, A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks, IEEE Communications Surveys & Tutorials 16 (3) (2014) 1617–1634.
URL https://doi.org/10.1109/SURV.2014.012214.00180

[2] Y. Zhou, C. Sun, H. H. Liu, R. Miao, S. Bai, B. Li, Z. Zheng, L. Zhu, Z. Shen, Y. Xi, et al., Flow event telemetry on programmable data plane, in: Proc. of SIGCOMM'20, ACM, Virtual Event, USA, 2020, pp. 76–89.
URL https://doi.org/10.1145/3387514.3406214

[3] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, L. J. Wobker, In-band network telemetry via programmable dataplanes, in: Proc. of SIGCOMM'15, Vol. 15, ACM, London, UK, 2015, pp. 1–2.
URL https://anirudhsk.github.io/papers/int-demo.pdf

[4] The P4.org Applications Working Group, In-band Network Telemetry (INT) Dataplane Specification, Specification Version 2.1, P4.org (Nov. 2020).
URL https://p4.org/wp-content/uploads/sites/53/p4-spec/docs/INT_v2_1.pdf

[5] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, D. Walker, P4: programming protocol-independent packet processors, SIGCOMM Computer Communication Review 44 (3) (2014) 87–95.
URL https://doi.org/10.1145/2656877.2656890

[6] F. Brockners, S. Bhandari, et al., Data Fields for In Situ Operations, Administration, and Maintenance (IOAM), RFC 9197 (Oct. 2022).
URL https://datatracker.ietf.org/doc/html/rfc9197

[7] L. Tan, W. Su, W. Zhang, J. Lv, Z. Zhang, J. Miao, X. Liu, N. Li, In-band network telemetry: A survey, Computer Networks 186 (2021) 107763.
URL https://doi.org/10.1016/j.comnet.2020.107763

[8] E. Ollora Zaballa, D. Franco, S. E. Thomsen, M. Higuero, H. Wessing, M. S. Berger, Towards monitoring hybrid next-generation software-defined and service provider MPLS networks, Computer Networks 191 (2021) 107960.
URL https://doi.org/10.1016/j.comnet.2021.107960

[9] F. Alhamed, D. Scano, P. Castoldi, J. J. Vegas Olmos, I. Vershkov, F. Paolucci, F. Cugini, P4 Telemetry collector, Computer Networks 227 (2023) 109727.
URL https://doi.org/10.1016/j.comnet.2023.109727

[10] J. Hyun, N. Van Tu, J. W.-K. Hong, Towards knowledge-defined networking using in-band network telemetry, in: Proc. of NOMS'18, IEEE, Taipei, Taiwan, 2018, pp. 1–7.
URL https://doi.org/10.1109/NOMS.2018.8406169

[11] D. Bhamare, A. Kassler, J. Vestin, M. A. Khoshkholghi, J. Taheri, T. Mahmoodi, P. Öhlén, C. Curescu, IntOpt: In-band Network Telemetry optimization framework to monitor network slices using P4, Computer Networks 216 (2022) 109214.
URL https://doi.org/10.1016/j.comnet.2022.109214

[12] S. Nam, J. Lim, J.-H. Yoo, J. W.-K. Hong, Network anomaly detection based on in-band network telemetry with RNN, in: Proc. of ICCE-Asia'20, IEEE, Seoul, Korea, 2020, pp. 1–4.
URL https://doi.org/10.1109/ICCE-Asia49877.2020.9276768

[13] R. Joshi, T. Qu, M. C. Chan, B. Leong, B. T. Loo, BurstRadar: Practical Real-time Microburst Monitoring for Datacenter Networks, in: Proc. of APSys'18, ACM, Jeju Island, Korea, 2018, pp. 1–4.
URL https://doi.org/10.1145/3265723.3265731

[14] C. Jia, T. Pan, Z. Bian, X. Lin, E. Song, C. Xu, T. Huang, Y. Liu, Rapid Detection and Localization of Gray Failures in Data Centers via In-band Network Telemetry, in: Proc. of NOMS'20, Budapest, Hungary, 2020, pp. 1–9.
URL https://doi.org/10.1109/NOMS47738.2020.9110326

[15] Z. Liu, J. Bi, Y. Zhou, Y. Wang, Y. Lin, NetVision: Towards Network Telemetry as a Service, in: Proc. of ICNP'18, IEEE, Cambridge, UK, 2018, pp. 247–248.
URL https://doi.org/10.1109/ICNP.2018.00036

[16] R. Ben Basat, S. Ramanathan, Y. Li, G. Antichi, M. Yu, M. Mitzenmacher, PINT: Probabilistic In-band Network Telemetry, in: Proc. of SIGCOMM'20, ACM, Virtual Event, USA, 2020, p. 662–680.
URL https://doi.org/10.1145/3387514.3405894

[17] N. S. Kagami, R. I. T. da Costa Filho, L. P. Gaspary, Capest: Offloading network capacity and available bandwidth estimation to programmable data planes, IEEE Transactions on Network and Service Management 17 (1) (2019) 175–189.
URL https://doi.org/10.1109/TNSM.2019.2934316

[18] T. Pan, E. Song, Z. Bian, X. Lin, X. Peng, J. Zhang, T. Huang, B. Liu, Y. Liu, Int-path: Towards optimal path planning for in-band network-wide telemetry, in: Proc. of INFOCOM'19, IEEE, Paris, France, 2019, pp. 487–495.
URL https://doi.org/10.1109/INFOCOM.2019.8737529

[19] K. Yang, S. Long, Q. Shi, Y. Li, Z. Liu, Y. Wu, T. Yang, Z. Jia, SketchINT: Empowering INT With TowerSketch for Per-Flow Per-Switch Measurement, IEEE Transactions on Parallel and Distributed Systems 34 (11) (2023) 2876–2894.
URL https://doi.org/10.1109/TPDS.2023.3303924

[20] Z. Wei, Y. Tian, W. Chen, L. Gu, X. Zhang, DUNE: Improving Accuracy for Sketch-INT Network Measurement Systems, in: Proc. of INFOCOM'23, New York City, NY, USA, 2023, pp. 1–10.
URL https://doi.org/10.1109/INFOCOM53939.2023.10229098

[21] D. Franco, E. Ollora Zaballa, M. Zang, A. Atutxa, J. Sasiain, A. Pruski, E. Rojas, M. Higuero, E. Jacob, A comprehensive latency profiling study of the Tofino P4 programmable ASIC-based hardware, Computer Communications 218 (2024) 14–30.
URL https://doi.org/10.1016/j.comcom.2024.01.010

[22] N. Choi, L. Jagadeesan, Y. Jin, N. N. Mohanasamy, M. R. Rahman, K. Sabnani, M. Thottan, Run-time performance monitoring, verification, and healing of end-to-end services, in: Proc. of NetSoft'19, IEEE, Paris, France, 2019, pp. 30–35.
URL https://doi.org/10.1109/NETSOFT.2019.8806660

[23] R. Hohemberger, A. F. Lorenzon, F. Rossi, M. C. Luizelli, Optimizing distributed network monitoring for NFV service chains, IEEE Communications Letters 23 (8) (2019) 1332–1336.
URL https://doi.org/10.1109/LCOMM.2019.2922184

[24] A. Sacco, A. Angi, F. Esposito, G. Marchetto, HINT: Supporting Congestion Control Decisions with P4-driven In-Band Network Telemetry, in: Proc. of HPSR'23, Albuquerque, NM, USA, 2023, pp. 83–88.
URL https://doi.org/10.1109/HPSR57248.2023.10147977

[25] L. C. de Almeida, W. R. D. da Silva, T. C. Tavares, R. Pasquini, C. Papagianni, F. L. Verdi, DESiRED — Dynamic, Enhanced, and Smart iRED: A P4-AQM with Deep Reinforcement Learning and In-band Network Telemetry, journal = Computer Networks, Computer Networks 244 (2024) 110326.
URL https://doi.org/10.1016/j.comnet.2024.110326

[26] N. Katta, A. Ghag, M. Hira, I. Keslassy, A. Bergman, C. Kim, J. Rexford, Clove: Congestion-Aware Load Balancing at the Virtual Edge, in: Proc. of CoNEXT'17, ACM, Incheon, Korea, 2017, p. 323–335.
URL https://doi.org/10.1145/3143361.3143401

[27] L. Tan, W. Su, W. Zhang, H. Shi, J. Miao, P. Manzanares-Lopez, A Packet Loss Monitoring System for In-Band Network Telemetry: Detection, Localization, Diagnosis and Recovery, IEEE Transactions on Network and Service Management 18 (4) (2021) 4151–4168.
URL https://doi.org/10.1109/TNSM.2021.3125012

[28] C. Fang, H. Liu, M. Miao, J. Ye, L. Wang, W. Zhang, D. Kang, B. Lyv, P. Cheng, J. Chen, VTrace: Automatic Diagnostic System for Persistent Packet Loss in Cloud-Scale Overlay Network, in: Proc. of SIGCOMM'20, ACM, Virtual Event, USA, 2020, p. 31–43.
URL https://doi.org/10.1145/3387514.3405851

[29] N. Katta, M. Hira, C. Kim, A. Sivaraman, J. Rexford, HULA: Scalable Load Balancing Using Programmable Data Planes, in: Proc. of SOSR'16, ACM, Santa Clara, CA, USA, 2016, pp. 1–12.
URL https://doi.org/10.1145/2890955.2890968

[30] H. Yao, T. Mai, X. Xu, P. Zhang, M. Li, Y. Liu, NetworkAI: An Intelligent Network Architecture for Self-Learning Control Strategies in Software Defined Networks, IEEE Internet of Things Journal 5 (6) (2018) 4319–4327.
URL https://doi.org/10.1109/JIOT.2018.2859480

[31] A. Clemm, L. Ciavaglia, L. Z. Granville, J. Tantsura, Intent-Based Networking - Concepts and Definitions, RFC 9315 (Oct. 2022).
URL https://datatracker.ietf.org/doc/rfc9315/

[32] L. M. Dallanora, A. G. Castro, R. I. T. d. C. Filho, F. D. Rossi, A. F. Lorenzon, M. C. Luizelli, Dypro: Dynamic probing planning for in-band network telemetry, in: Proc. of ISCC'22, Rhodes, Greece, 2022, pp. 1–6.
URL https://doi.org/10.1109/ISCC55528.2022.9912881

[33] M. Cui, Y. Peng, Y. Wang, T. Niu, F. Yang, FANT: Flexible active in-band network telemetry, Computer Communications (2025) 108336.
URL https://doi.org/10.1016/j.comcom.2025.108336

[34] Q. Yuan, F. Li, T. Pan, Y. Xu, X. Wang, INT-react: An O(E) Path Planner for Resilient Network-Wide Telemetry Over Megascale Networks, in: Proc. of ICNP'22, IEEE, Lexington, KY, USA, 2022, pp. 1–11.
URL https://doi.org/10.1109/ICNP55882.2022.9940409

[35] P. Zhang, H. Zhang, Y. Pi, Z. Cao, J. Wang, J. Liao, Adapint: A flexible and adaptive in-band network telemetry system based on deep reinforcement learning, IEEE Transactions on Network and Service Management 21 (5) (2024) 5505–5520.
URL https://doi.org/10.1109/TNSM.2024.3427403

[36] P. Bosshart, G. Gibb, H.-S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, M. Horowitz, Forwarding metamorphosis: fast programmable match-action processing in hardware for SDN, SIGCOMM Computer Communication Review 43 (4) (2013) 99–110.
URL https://doi.org/10.1145/2534169.2486011

[37] S. Kaul, R. Yates, M. Gruteser, Real-time status: How often should one update?, in: 2012 Proc. IEEE INFOCOM, IEEE, Orlando, FL, USA, 2012, pp. 2731–2735.
URL https://doi.org/10.1109/INFCOM.2012.6195689

[38] A. Morton, Active and Passive Metrics and Methods (with Hybrid Types In-Between), RFC 7799 (may 2016).
URL https://datatracker.ietf.org/doc/html/rfc7799

[39] B. D. McKay, R. W. Robinson, Asymptotic enumeration of eulerian circuits in the complete graph, Combinatorics, Probability and Computing 7 (4) (1998) 437–449.
URL https://doi.org/10.1017/S0963548398003642

[40] P. L. Ventre, S. Salsano, M. Polverini, A. Cianfrani, A. Abdelsalam, C. Filsfils, P. Camarillo, F. Clad, Segment routing: A comprehensive survey of research activities, standardization efforts, and implementation results, IEEE Communications Surveys & Tutorials 23 (1) (2021) 182–221.
URL https://doi.org/10.1109/COMST.2020.3036826

[41] T. Pan, X. Lin, H. Song, E. Song, Z. Bian, H. Li, J. Zhang, F. Li, T. Huang, C. Jia, et al., INT-probe: Lightweight In-band Network-Wide Telemetry with Stationary Probes, in: Proc. of ICDCS'21, IEEE, DC, USA, 2021, pp. 898–909.
URL https://doi.org/10.1109/ICDCS51616.2021.00090

[42] Q. Yuan, F. Li, T. Pan, Y. Lai, Y. Gu, X. Wang, INT-Segment: MTU-Adaptive Single-Path In-Band Network-Wide Telemetry, in: Proc. of ICNP'22, IEEE, Lexington, KY, USA, 2022, pp. 1–11.
URL https://doi.org/10.1109/ICNP55882.2022.9940397

[43] F. Li, Q. Yuan, T. Pan, X. Wang, J. Cao, MTU-adaptive in-band network-wide telemetry, IEEE/ACM Transactions on Networking 32 (3) (2024) 2315–2330.
URL https://doi.org/10.1109/TNET.2024.3351672

[44] Q. Yuan, F. Li, T. Pan, Y. Lai, Y. Gu, X. Wang, J. Cao, INT-Partition:

Hierarchical and Fault-Tolerant In-Band Network Telemetry, IEEE Transactions on Networking 33 (5) (2025) 2617–2631.
URL https://doi.org/10.1109/TON.2025.3569340

[45] D. Chen, D. Gao, C. H. Foh, H. Yan, CFINT: Cluster Based Fast In-band Network-wide Telemetry in 6G-enabled Networks, in: Proc. of GLOBECOM Wkshps'22, Rio de Janeiro, Brazil, 2022, pp. 1699–1704.
URL https://doi.org/10.1109/GCWkshps56602.2022.10008610

[46] Z. Wang, D. Jiang, K. Zhang, GP-INT: Generating Balanced Network-Wide In-Band Telemetry Path for Digital Twin Networks, in: Proc. of ICC'24, IEEE, Denver, CO, USA, 2024, pp. 1109–1114.
URL https://doi.org/10.1109/ICC51166.2024.10622449

[47] Y. Zhang, T. Pan, Y. Zheng, E. Song, J. Liu, T. Huang, Y. Liu, INT-balance: In-band network-wide telemetry with balanced monitoring path planning, in: Proc. of ICC'23, IEEE, Rome, Italy, 2023, pp. 2351–2356.
URL https://doi.org/10.1109/ICC45041.2023.10279077

[48] K. Yu, S. Chen, F. Li, J. Shen, X. Wang, SRv6-ALINT: SRv6-based Efficient In-band Network-Wide Telemetry across LANs, in: Proc. of ICCCN'25, IEEE, Tokyo, Japan, 2025, pp. 1–9.
URL https://doi.org/10.1109/ICCCN65249.2025.11133970

[49] S. Chen, D. He, X. Ma, Z. Ming, L. Cui, WAN-INT: Cost-effective in-band network telemetry in WAN with a performance-aware path planner, in: Proc. of ICPADS'23, IEEE, Ocean Flower Island, China, 2023, pp. 1861–1868.
URL https://doi.org/10.1109/ICPADS60453.2023.00256

[50] M. Ji, C. Su, Y. Yan, Z. Qian, Y. Chen, Y. Jin, S. Zhang, B. Ye, INTView: Adaptive planner for in-band network telemetry without detours, in: Proc. of ICC'23, IEEE, Rome, Italy, 2023, pp. 5490–5495.
URL https://doi.org/10.1109/ICC45041.2023.10279624

[51] P. Zhong, F. Zhou, L. Feng, W. Li, FINT: Freshness-Based In-Band Network-Wide Telemetry in Resource-Constrained Environments, IEEE Transactions on Network and Service Management 22 (2) (2025) 1314–1329.
URL https://doi.org/10.1109/TNSM.2024.3500586

[52] A. G. Castro, V. H. S. Lopes, F. G. Vogt, F. D. Rossi, A. F. Lorenzon, M. C. Luizelli, Patcher: Towards Fault - Tolerant Probing Planning for In-band Network Telemetry, in: Proc. of LATINCOM'20, IEEE, Santo Domingo, Dominican Republic, 2020, pp. 1–6.
URL https://doi.org/10.1109/LATINCOM50620.2020.9282348

[53] F. Li, Q. Yuan, Y. Lai, Z. Guo, E. Wen, T. Pan, X. Wang, J. Cao, INT-Source: Topology-Adaptive In-Band Network-Wide Telemetry, IEEE Transactions on Networking (2025) 1–15.
URL https://doi.org/10.1109/TON.2025.3589194

[54] M. Cui, Q. Zhang, Y. Peng, F. Yang, ATINT: Planning for Anomaly-Tolerant Probing for Network-Wide In-Band Network Telemetry, in: Proc. of ICC'25, IEEE, Montreal, QC, Canada, 2025, pp. 6523–6529.
URL https://doi.org/10.1109/ICC52391.2025.11160743

[55] Q. Zheng, S. Tang, B. Chen, Z. Zhu, Highly-Efficient and Adaptive Network Monitoring: When INT Meets Segment Routing, IEEE Transactions on Network and Service Management 18 (3) (2021) 2587–2597.
URL https://doi.org/10.1109/TNSM.2021.3069000

[56] B. Chen, F. Chen, S. Tang, Q. Zheng, Z. Zhu, On orchestration of segment routing and in-band network telemetry, IEEE Transactions on Network and Service Management 20 (4) (2023) 4047–4060.
URL https://doi.org/10.1109/TNSM.2023.3254200

[57] T. Wu, H. Yao, W. He, Z. Wang, T. Mai, Z. Xiong, S. Guo, Low-cost network measurement through intelligent in-band network telemetry orchestration, in: Proc. of GLOBECOM'23, IEEE, Kuala Lumpur, Malaysia, 2023, pp. 4326–4331.
URL https://doi.org/10.1109/GLOBECOM54140.2023.10436987

[58] P. Zhang, H. Zhang, Y. Dai, C. Zeng, J. Wang, J. Liao, INT-LLPP: Lightweight in-band network-wide telemetry with low-latency and low-overhead path planning, Computer Communications 236 (2025) 108 142.
URL https://doi.org/10.1016/j.comcom.2025.108142

[59] P. Zhang, H. Zhang, Y. Dai, Y. Pi, J. Wang, J. Liao, Cache-INT: In-network caching-enabled In-band Network Telemetry, Computer Networks (2025) 111404.
URL https://doi.org/10.1016/j.comnet.2025.111404

[60] D. Zhao, G. Cheng, X. Chen, Y. Zhao, W. Zhang, L. Lu, S. Zhou, Y. Fu, Probe-Optimizer: Discovering important nodes for proactive in-band network telemetry to achieve better probe orchestration, Computer Networks 257 (2025) 110935.
URL https://doi.org/10.1016/j.comnet.2024.110935

[61] J. Cai, H. Lin, T. Sun, Z. Zhou, L. Zhu, H. Chen, J. Zhou, D. Zhang, C. Wu, Openint: Dynamic in-band network telemetry with lightweight deployment and flexible planning, in: Proc. of INFOCOM'24, IEEE, Vancouver, BC, Canada, 2024, pp. 2488–2497.
URL https://doi.org/10.1109/INFOCOM52122.2024.10621221

[62] A. G. Castro, A. F. Lorenzon, F. D. Rossi, R. I. T. d. C. Filho, F. M. V. Ramos, C. E. Rothenberg, M. C. Luizelli, Near-Optimal Probing Planning for In-Band Network Telemetry, IEEE Communications Letters 25 (5) (2021) 1630–1634.
URL https://doi.org/10.1109/LCOMM.2021.3053485

[63] D. Bhamare, A. Kassler, J. Vestin, M. A. Khoshkholghi, J. Taheri, IntOpt: In-Band Network Telemetry Optimization for NFV Service Chain Monitoring, in: Proc. of ICC'19, IEEE, Shanghai, China, 2019, pp. 1–7.
URL https://doi.org/10.1109/ICC.2019.8761722

[64] Y. Liu, Y. Xia, W. Zhang, W. Jia, J. Wu, SFANT: A SRv6-based flexible and active network telemetry scheme in programming data plane, IEEE Transactions on Network Science and Engineering 11 (3) (2023) 2415–2425.
URL https://doi.org/10.1109/TNSE.2023.3277000

[65] Y. Xie, Y. Zhu, J. Feng, X. Chen, X. Xiong, S. Zheng, INToSR: Building A Novel In-band Network Telemetry over SRv6, in: Proc. of ISPA'24, IEEE, Kaifeng, China, 2024, pp. 1732–1739.
URL https://doi.org/10.1109/ISPA63168.2024.00236

[66] J. Xu, X. Xu, J. Zhao, H. Gao, FDSR-INT: A Flexible On-Demand In-Band Telemetry Approach for Aerial Computing Networks, IEEE Internet of Things Journal 12 (13) (2025) 23257–23274.
URL https://doi.org/10.1109/JIOT.2025.3551279

[67] J. A. Marques, M. C. Luizelli, R. I. Tavares da Costa Filho, L. P. Gaspary, An optimization-based approach for efficient network monitoring using in-band network telemetry, Journal of Internet Services and Applications 10 (1) (2019) 1–20.
URL https://doi.org/10.1186/s13174-019-0112-0

[68] Z. Zhang, W. Su, L. Tan, In-band Network Telemetry Task Orchestration based on Multi-objective Optimization, in: Proc. of APNOMS'21, IEEE, Tainan, Taiwan, 2021, pp. 354–357.
URL https://doi.org/10.23919/APNOMS52696.2021.9562646

[69] Y. Gu, Q. Yuan, F. Li, N. Zheng, K. Guo, T. Pan, X. Wang, Int-Selection: Passive In-Band Network-Wide Telemetry Based on Flow Selection, in: Proc. of IWQoS'25, IEEE, Gold Coast, Australia, 2025, pp. 1–2.
URL https://doi.org/10.1109/IWQoS65803.2025.11143315

[70] S. Tang, S. Zhao, X. Pan, Z. Zhu, How to Use In-Band Network Telemetry Wisely: Network-Wise Orchestration of Sel-INT, IEEE/ACM Transactions on Networking 31 (1) (2022) 421–435.
URL https://doi.org/10.1109/TNET.2022.3194086

[71] T. Ouyang, H. Yao, W. He, T. Mai, F. Wang, F. R. Yu, Self-Adaptive Dynamic In-Band Network Telemetry Orchestration for Balancing Accuracy and Stability, IEEE Transactions on Network and Service Management 22 (2) (2025) 1514–1530.
URL https://doi.org/10.1109/TNSM.2025.3530432

[72] R. Hohemberger, A. F. Lorenzon, F. D. Rossi, M. C. Luizelli, A Heuristic Approach for Large-Scale Orchestration of the In-band Data Plane Telemetry Problem, in: Proc. of AINA'20, Springer, Cham, 2020, pp. 381–392.
URL https://doi.org/10.1007/978-3-030-44041-1_35

[73] T. B. N'Diaye, M. C. Luizelli, J. Degila, L. S. Buriol, Optimizing in-band network telemetry problem, Procedia Computer Science 239 (2024) 2074–2081.

URL https://doi.org/10.1016/j.procs.2024.06.394

[74] K. Papadopoulos, P. Papadimitriou, C. Papagianni, Deterministic and Probabilistic P4-Enabled Lightweight In-Band Network Telemetry, IEEE Transactions on Network and Service Management 20 (4) (2023) 4909–4922.
URL https://doi.org/10.1109/TNSM.2023.3301839

[75] E. Song, T. Pan, C. Jia, W. Cao, J. Zhang, T. Huang, Y. Liu, INT-label: Lightweight In-band Network-Wide Telemetry via Interval-based Distributed Labelling, in: IEEE INFOCOM 2021 - IEEE Conference on Computer Communications, Vancouver, BC, Canada, 2021, pp. 1–10.
URL https://doi.org/10.1109/INFOCOM42981.2021.9488799

[76] S. Sheng, Q. Huang, P. P. Lee, A general delta-based in-band network telemetry framework with extremely low bandwidth overhead, Computer Networks 223 (2023) 109573.
URL https://doi.org/10.1016/j.comnet.2023.109573

[77] M. Qian, L. Cui, F. P. Tso, Y. Deng, W. Jia, OffsetINT: Achieving high accuracy and low bandwidth for in-band network telemetry, IEEE Transactions on Services Computing 17 (3) (2023) 1072–1083.
URL https://doi.org/10.1109/TSC.2023.3323697

[78] S. Sardellitti, M. Polverini, S. Barbarossa, A. Cianfrani, P. Di Lorenzo, M. Listanti, In band network telemetry overhead reduction based on data flows sampling and recovering, in: Proc. of NetSoft'23, IEEE, Madrid, Spain, 2023, pp. 414–419.
URL https://doi.org/10.1109/NetSoft57336.2023.10175471

[79] S. Xie, G. Hu, C. Xing, J. Zu, Y. Liu, FINT: Flexible In-band Network Telemetry method for data center network, Computer Networks 216 (2022) 109–232.
URL https://doi.org/10.1016/j.comnet.2022.109232

[80] Y. Wu, M. Zhang, Event Sampling Based Network Telemetry Approach to Reduce Overhead, in: Proc. of AINIT'24, IEEE, Toulouse, France, 2024, pp. 1011–1014.
URL https://doi.org/10.1109/AINIT61980.2024.10581579

[81] R. Hohemberger, A. G. Castro, F. G. Vogt, R. B. Mansilha, A. F. Lorenzon, F. D. Rossi, M. C. Luizelli, Orchestrating in-band data plane telemetry with machine learning, IEEE Communications Letters 23 (12) (2019) 2247–2251.
URL https://doi.org/10.1109/LCOMM.2019.2946562

[82] K. Zhang, W. Zhang, L. Liu, L. Tan, Y. Zhang, W. Gao, Hawkeye: Efficient In-band Network Telemetry with Hybrid Proactive-Passive Mechanism, in: Proc. of ISPA'22, Melbourne, Australia, 2022, pp. 903–912.
URL https://doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom57177.2022.00120

[83] Y. Cao, Y. Cui, H. Gao, C. Guo, Y. Chen, G. Shen, Patrol: Max-Min Passive-Active Hybrid in-Band Network Telemetry Method in SDN, in: Proc. of ISPA'25, Shenyang, China, 2025, pp. 593–600.
URL https://doi.org/10.1109/ISPA67752.2025.00082

[84] K. Zhang, W. Su, H. Shi, K. Zhang, W. Zhang, GrayINT–Detection and Localization of Gray Failures via Hybrid In-band Network Telemetry, in: Proc. of APNOMS'23, Sejong, Korea, Republic of, 2023, pp. 405–408.
URL https://ieeexplore.ieee.org/abstract/document/10258170

[85] Y. Zhu, N. Kang, J. Cao, A. Greenberg, G. Lu, R. Mahajan, D. Maltz, L. Yuan, M. Zhang, B. Y. Zhao, H. Zheng, Packet-Level Telemetry in Large Datacenter Networks, in: Proc. of SIGCOMM'15, ACM, London, United Kingdom, 2015, p. 479–491.
URL https://doi.org/10.1145/2785956.2787483

[86] Q. Huang, H. Sun, P. P. C. Lee, W. Bai, F. Zhu, Y. Bao, OmniMon: Re-architecting Network Telemetry with Resource Efficiency and Full Accuracy, in: Proc. of SIGCOMM'20, ACM, Virtual Event, USA, 2020, p. 404–421.
URL https://doi.org/10.1145/3387514.3405877

[87] C. Guo, L. Yuan, D. Xiang, Y. Dang, R. Huang, D. Maltz, Z. Liu, V. Wang, B. Pang, H. Chen, Z.-W. Lin, V. Kurien, Pingmesh: A Large-Scale System for Data Center Network Latency Measurement and Analysis, SIGCOMM Computer Communication Review 45 (4) (2015) 139–152.
URL https://doi.org/10.1145/2785956.2787496

[88] ONOS Project, Intent framework (May 2016).
URL https://opennetworking.org/onos/

[89] B. E. Ujcich, A. Bates, W. H. Sanders, Provenance for Intent-Based Networking, in: Proc. of NetSoft'20, IEEE, Ghent, Belgium, 2020, pp. 195–199.
URL https://doi.org/10.1109/NetSoft48620.2020.9165519

[90] Q. Zhang, V. Liu, H. Zeng, A. Krishnamurthy, High-resolution measurement of data center microbursts, in: Proc. of IMC'17, ACM, London, United Kingdom, 2017, p. 78–85.
URL https://doi.org/10.1145/3131365.3131375

[91] R. Stoyanov, N. Zilberman, MTPSA: Multi-Tenant Programmable Switches, in: Proceedings of the 3rd P4 Workshop in Europe, EuroP4'20, ACM, Barcelona, Spain, 2020, p. 43–48.
URL https://doi.org/10.1145/3426744.3431329

[92] S. Khashab, A. Rashelbach, M. Silberstein, Multitenant In-Network Acceleration with SwitchVM, in: 21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24), USENIX Association, Santa Clara, CA, 2024, pp. 691–708.
URL https://doi.org/10.5555/3691825.3691863

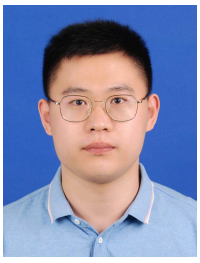
[93] S. Zhu, J. Lu, B. Lyu, T. Pan, C. Jia, X. Cheng, D. Kang, Y. Lv, F. Yang, X. Xue, Z. Wang, J. Yang, Zoonet: a proactive telemetry system for large-scale cloud networks, in: Proceedings of the 18th International Conference on Emerging Networking EXperiments and Technologies, CoNEXT '22, Association for Computing Machinery, Roma, Italy, 2022, p. 321–336.
URL https://doi.org/10.1145/3555050.3569116

[94] J. Liu, W. Hallahan, C. Schlesinger, M. Sharif, J. Lee, R. Soulé, H. Wang, C. Caşcaval, N. McKeown, N. Foster, p4v: practical verification for programmable data planes, in: Proc. of SIGCOMM'18, ACM, Budapest, Hungary, 2018, p. 490–503.
URL https://doi.org/10.1145/3230543.3230582

[95] X. Pan, S. Tang, S. Liu, J. Kong, X. Zhang, D. Hu, J. Qi, Z. Zhu, Privacy-Preserving Multilayer In-Band Network Telemetry and Data Analytics: For Safety, Please do Not Report Plaintext Data, Journal of Lightwave Technology 38 (21) (2020) 5855–5866.
URL https://doi.org/10.1109/JLT.2020.3007491

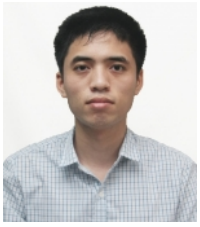
[96] L.-C. Schulz, D. Hausheer, ID-INT: Secure Inter-Domain In-Band Telemetry, in: Proc. of CNSM'24, Prague, Czech Republic, 2024, pp. 1–9.
URL https://doi.org/10.23919/CNSM62983.2024.10814310

Yonghao Zhang is pursuing his M.S. degree with the Shandong Provincial Key Laboratory of Computer Networks, Shandong Computer Science Center (National Supercomputer Center in Ji'nan), Qilu University of Technology (Shandong Academy of Sciences), China. His research interests include Software-defined Networking and Programmable Networks.



Lizhuang Tan is currently an Associate Professor with Shandong Provincial Key Laboratory of Computer Networks, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences). He received his Ph.D. degree from School of Electronic and Information Engineering, Beijing Jiaotong University, China, in 2022. He has published more than 20 journal or conference papers, such as IEEE TCC, IEEE TNSM, ELSEVIER COMNETS, APNOMS, etc. His research interests include Network Measurement, Management and Optimization.

Yuyu Zhao is a currently a Lecturer with the School of Cyber Science and Engineering, Southeast University. He received the B.S. degree in software engineering from the Nanjing University of Science and Technology, China, in 2016 and the M.S. and Ph.D. degrees in cyber security from Southeast University, China, in 2019 and 2023. His research interests include In-band Network Telemetry, Blockchain, and Network Processors.

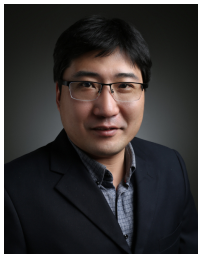


Nguyen Van Tu is an engineer at MangoBoost Inc. He received the B.Sc. degree in electronics and communication from the Hanoi University of Science and Technology, Vietnam, in 2015, the M.Sc. and Ph.D. degree in computer science from Pohang University of Science and Technology, South Korea, in 2018 and 2025. His research is focused on High-performance Networks and Programmable Networks.

Pilar Manzanares-Lopez is an Associate Professor with the Department of Information and Communication Technologies, Universidad Politécnica de Cartagena (UPCT), Spain. She received the M.S. degree in telecommunication engineering from the Universidad Politécnica de Valencia, Spain, in 2001, and the Ph.D. degree in telecommunication engineering from the Universidad Politécnica de Cartagena (UPCT), Spain, in 2006. Her research interests include software-defined networking, programmable data planes, P2P networks, and distributed systems.

Na Li works with the Shandong Branch of National Computer Network Emergency Response Technical Team/Coordination Center (CNCERT/SD). She is pursuing her Ph.D. degree in the School of Cyber Science and Technology, Shandong University, Qingdao, China. She graduated from College of Information Science and Engineering, Ocean University of China with her M.S. degree in communication and information system in 2019. Her research interests are 5G/6G, Software-defined Networking and Communication Security.

Huiling Shi is currently an Associate Researcher with Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences). She received her M.S. degree from Shandong University, P. R. China in 2004. Her research interests include Network Architectures and Supercomputing Network.



Wei Zhang is currently a Professor with the Shandong Provincial Key Laboratory of Computer Networks, Shandong Computer Science Center (National Supercomputer Center in Ji'nan), Qilu University of Technology (Shandong Academy of Sciences). He received his B.E. degree from Zhejiang University, P. R. China, in 2004, M.S. degree from Liaoning University, P. R. China, in 2008, and Ph.D. degree from Shandong University of Science and Technology, P. R. China, in 2018. His research interests include Future-generation Network Architecture, Edge Computing, and Edge Intelligence.

Peiying Zhang is currently a Professor with the College of Computer Science and Technology, China University of Petroleum (East China). He received his Ph.D. from the School of Information and Communication Engineering at Beijing University of Posts and Telecommunications in 2019. He has published 100 IEEE/ACM Trans./Journal/Magazine papers, such as IEEE TII, IEEE T-ITS, IEEE TVT, IEEE TNSE, IEEE TNSM, IEEE TETC, IEEE Network. His research interests include Semantic Computing, Future Internet Architecture, Network Virtualization, and Artificial Intelligence for Networking.

Wei Su is a Professor at the School of Electronic and Information Engineering, Beijing Jiaotong University. He received his B.S., M.S., and Ph.D. degrees in communication and information systems from Beijing Jiaotong University, P. R. China, in 2001, 2004 and 2008. He has published more than 50 research papers and 4 monographs in communications and computer networks. His research interests include Next-generation Network and Mobile Internet.

James Won-Ki Hong is a Professor in the Department of Computer Science and Engineering at Pohang University of Science and Technology (POSTECH), Pohang, Korea. He received his HBSc and MSc degrees in Computer Science from the University of Western Ontario, Canada, in 1983 and 1985, respectively, and the PhD degree in Computer Science from the University of Waterloo, Canada, in 1991. He had worked as the Chief Technology Officer and Senior Executive Vice President for KT (Korea Telecom), the largest telecommunications company in Korea

from March 2012 to February 2014, where he was responsible for leading the R&D effort of KT and its subsidiary companies. He was Chairman of National Intelligence Communication Enterprise Association and Chairman of ICT Standardization Committee in Korea. He cofounded and is currently served as Executive Director of SDN/NFV Forum in Korea. He had served as the Head of Department of Computer Science and Engineering, Dean of Graduate School of Information Technology, Director of POSTECH Information Research Labs, and Head of the Division of IT Convergence Engineering at POSTECH. He has served as Chairman of the IEEE Communications Society, Committee on Network Operations and Management. He has also served IEEE ComSoc Director of Online Content (2004–2005, 2010–2011). He is the Editor-in-Chief of International Journal on Network Management (IJNM), IEEE ComSoc Technology News, and KNOM Review Journal. He is the General Chair of NOMS'24, ICBC'19, NetSoft'16 and APNOMS'06. He is an editorial board member of IEEE Transactions on Network and Service Management, Journal of Network and Systems Management and Journal of Communications and Networks. His research interests include network innovation, such as software-defined networking and network function virtualization, cloud computing, mobile services, IPTV, ICT convergence technologies (e.g., Smart Home, Smart Energy, and Health care), and Internet of Things.