International Journal of Computer Applications (0975 - 8887)
Volume 187 - No.125, July 2026

PathAB: Hybrid Two-Phase Probing for Available Bandwidth Estimation with Standalone Support

Debashis Roy
Independent Researcher
Redmond, WA, United States

ABSTRACT

Accurate estimation of end-to-end available bandwidth is essential for network-aware applications including adaptive streaming, congestion control, and traffic engineering. Existing active probing techniques either assume fluid cross-traffic models, require cooperative software at both endpoints, or exhibit high estimation error when cross-traffic conditions vary. This paper presents PathAB, a hybrid active probing method that estimates available bandwidth in two phases: a single exponentially spaced probing train provides a rapid coarse estimate, followed by multiple Poisson-distributed probing trains that refine the estimate using a linear regression model. PathAB supports both client–server and standalone modes. In standalone mode, small 28-byte echo packets are placed behind larger probe packets to minimize reverse-path interference, eliminating the need for receiver-side software. The algorithm is evaluated through NS-2 simulations at four bottleneck capacities (1.5, 5, 10, and 15 Mbps) under four utilization levels, and through network test-bed experiments in single-hop, multi-hop, and 100 Mbps configurations. PathAB is compared against PathChirp, PoissonProb, IGI, Spruce, Pathload, and a stochastic model. Results show that PathAB achieves RMS estimation error below 10% under moderate utilization in both modes, and maintains stable accuracy under high-utilization conditions where competing methods degrade substantially.

General Terms

Network measurement, performance evaluation, algorithms

Keywords

Available bandwidth estimation, active probing, network measurement, NS-2 simulation, Poisson probing, standalone bandwidth estimation

1. INTRODUCTION

End-to-end available bandwidth—the unused capacity along a network path that can be offered to a new flow without disrupting existing traffic—is a fundamental metric for many networked applications. Adaptive media streaming systems use bandwidth estimates to select encoding rates. Overlay routing protocols rely on path quality information to choose forwarding paths. Congestion control mechanisms benefit from timely knowledge of spare capacity. Despite its importance, measuring available bandwidth accurately remains challenging because the metric is dynamic, path-dependent, and influenced by cross-traffic patterns at every link.

Active probing techniques inject carefully designed packet trains into the network and infer available bandwidth from observed delay or dispersion changes. These techniques fall into two broad categories. Gap-based methods [1]–[2] analyze inter-packet spacing changes but require prior knowledge of bottleneck capacity. Rate-based methods [3]–[4] detect congestion onset at increasing probing rates but may inject excessive traffic and often require software at both endpoints. Most existing methods assume a fluid cross-traffic model. However, studies by Cao et al. [5] show that Internet traffic tends toward Poisson behavior at sufficient aggregation levels, suggesting that Poisson-distributed probing may interact more naturally with real network traffic.

A further practical limitation is the deployment model. Most estimation tools operate in client–server mode, requiring cooperative software at both the source and destination. The few tools supporting standalone operation—where only the sender requires specialized software—typically echo full-size probe packets, making them vulnerable to reverse-path cross-traffic distortion.

This paper presents PathAB, a hybrid algorithm that addresses these limitations by combining ideas from MoSeab [6], PoissonProb [7], and PathChirp [8]. PathAB operates in two phases: an initial phase sends a single exponentially spaced probing train to obtain a coarse estimate rapidly, and a direct probing phase sends multiple Poisson-distributed probing trains to refine the estimate through linear regression. In standalone mode, PathAB transmits small 28-byte ICMP echo packets placed behind the larger probe packets, preserving the gap structure created by the forward-path bottleneck while minimizing reverse-path interference.

The main contributions of this paper are:

(1) A two-phase probing strategy that reduces measurement overhead by combining exponential coarse estimation with focused Poisson-distributed refinement.

(2) A hybrid design that integrates the strengths of three existing probing techniques into a unified algorithm.

(3) A standalone operation mode using small echo packets to mitigate reverse-path cross-traffic distortion.

(4) A comprehensive comparative evaluation through NS-2 simulations and hardware test-bed experiments across multiple

topologies, capacities, and cross-traffic configurations, comparing PathAB against six representative algorithms.

2. RELATED WORK

2.1 Gap-Based Approaches

Gap-based algorithms estimate available bandwidth from the dispersion between probe packets. If a probe packet pair of size q is sent with input gap Δ_{in} across a path with tight-link capacity C, cross-traffic queued behind the first packet increases the output gap Δ_{out}. The available bandwidth is estimated as:

$$A = \left(1 - \frac{\Delta_{\text{out}} - \Delta_{\text{in}}}{\Delta_{\text{in}}}\right) \times C \tag{1}$$

Cprobe [1] was the first algorithm to use dispersion of long packet trains. Spruce [2] uses Poisson-distributed inter-pair gaps but requires known bottleneck capacity. IGI/PTR [9] develops a single-hop gap model. These methods share the limitation of requiring prior knowledge of bottleneck capacity and assuming that the bottleneck link is also the tight link, which does not always hold in multi-hop paths.

2.2 Rate-Based Approaches

Rate-based algorithms send probing trains at varying rates and detect the congestion onset point. Pathload [3] uses Self-Loading Periodic Streams with binary search, requiring $\log_2(R_0)$ probing streams to converge, making it slow under high utilization. PathChirp [8] uses exponentially spaced probe packets within a chirp train, allowing probing over a rate range using $\log(G_2) - \log(G_1)$ packets. PathChirp is efficient but operates only in client–server mode. PoissonProb [7] uses Poisson-distributed inter-packet gaps and supports standalone mode, but echoes full-size probe packets, making it vulnerable to reverse-path cross-traffic.

2.3 Hybrid and Model-Based Approaches

MoSeab [6] combines rate-based and gap-based ideas. Its first phase iteratively probes the network starting at 200 Kbps, doubling the rate until one-way delay increases, then reports $\tilde{A} = R/\sqrt{2}$. Its second phase uses four probing trains and a mathematical model to compute the final estimate via linear regression. MoSeab is accurate but requires many iterative trains in its first phase and operates only in client–server mode.

BET [10] integrates packet-train dispersion, PathChirp's exponential chirp, and Pathload's SLoPS into a three-phase process. While BET also combines multiple techniques, its three-phase design introduces significant complexity. PathAB achieves comparable accuracy with a simpler two-phase design and additionally supports standalone operation.

A stochastic-model-based approach [11] uses queueing-theoretic analysis for bandwidth estimation, performing well under certain traffic conditions but requiring up to 300 samples to converge under Poisson traffic.

Table 1 summarizes the key characteristics of representative algorithms.

2.4 Recent Developments

Since the early 2010s, bandwidth estimation research has broadened. Hemmati and Bhattacharyya [12] studied dispersion-based techniques on 10 Gbps links, showing that traditional inter-packet gap assumptions break down at high speeds due to hardware timestamping limitations. Zhu et al. [13] proposed packet-level telemetry for datacenter networks, enabling inline bandwidth monitoring without active probing but requiring switch-level support. Yin et al. [14] investigated machine-learning-based bandwidth prediction at ultra-high speeds. Grigorescu et al. [15] explored ML-based prediction for SDN-based networks.

Despite these advances, the fundamental challenge of providing accurate, low-overhead, standalone estimates on general Internet paths remains. ML-based approaches require training data and path-specific models; telemetry approaches require infrastructure support. PathAB's hybrid design and small-echo-packet standalone mode address practical deployment constraints that newer approaches have not fully resolved.

Table 1. Comparison of Representative AB Estimation Algorithms

Algorithm	Approach	SA	Key Limitation
Cprobe [1]	Gap	Yes	BN = tight link
Spruce [2]	Gap	No	Known BN cap.
IGI/PTR [9]	Gap	No	Single-hop model
Pathload [3]	Rate	No	Slow convergence
PathChirp [8]	Rate	No	CS only
PoissonProb [7]	Rate	Yes	Rev.-path CT
MoSeab [6]	Hybrid	No	Iterative; CS only
Stoch. Model [11]	Model	No	300 samples
PathAB	**Hybrid**	**Yes**	**See Sec. 3**

CS = client–server; SA = standalone; CT = cross-traffic; BN = bottleneck.

3. PATHAB ALGORITHM

PathAB estimates available bandwidth in two phases: an initial probing phase for coarse estimation and a direct probing phase for refinement. The algorithm supports both client–server and standalone modes and assumes FIFO queuing at all routers along the path.

3.1 Definitions

Consider an end-to-end path of H links. Let C_i denote the capacity and $\lambda_i(t)$ the cross-traffic rate on link i. The available bandwidth of link i over interval T is:

$$A_i(t,T) = C_i - \frac{1}{T}\int_t^{t+T} \lambda_i(\tau)\, d\tau \tag{2}$$

The path available bandwidth is $A_{\text{path}} = \min_{i=1}^{H} A_i$. The link with minimum capacity is the *bottleneck link*; the link with minimum available bandwidth is the *tight link*. These may differ.

3.2 Initial Probing Phase

PathAB sends a single train of probe packets with exponentially decreasing inter-packet intervals. The instantaneous rate increases from $R_{\min}$ to $R_{\max}$ (100 Kbps to 100 Mbps by default). Let b denote the probe packet size (1200 bytes) and $\gamma = 1.2$ the spread factor.

At the receiver, the one-way delay (OWD) increase ratio is computed:

$$R_{\text{OWD}}(i) = \frac{OWD(i) - OWD(i-1)}{OWD(i-1)} \tag{3}$$

If δ_i is the i-th input gap, the instantaneous rate is $r_i = b/\delta_i$. When $r_i > A$:

$$R_{\text{OWD}}(i) = \frac{b - A \cdot \delta_i}{C \cdot \delta_i} \tag{4}$$

Traffic fluctuations cause spikes in the R_{OWD} curve. PathAB applies a spike-removal procedure that enforces $R_{\text{OWD}}(i) \geq R_{\text{OWD}}(i-1)$ for all i while preserving the total sum:

Algorithm: RemoveSpikes
for $i = 2$ **to** N **do**
  if $R_{\text{OWD}}(i) < R_{\text{OWD}}(i-1)$ **then**
    $\text{sum} = \sum_{j=1}^{i-1} R_{\text{OWD}}(j)$
    $d = \frac{R_{\text{OWD}}(i-1) - R_{\text{OWD}}(i)}{\text{sum} + R_{\text{OWD}}(i-1)}$
    for $j = 1$ **to** $i-1$ **do**
      $R_{\text{OWD}}(j) = (1-d) \times R_{\text{OWD}}(j)$
    $R_{\text{OWD}}(i) = R_{\text{OWD}}(i-1)$

After smoothing, an exponential trend line $y = ae^{cx}$ is fitted to the R_{OWD} curve. The trend line is used for threshold detection: the coarse estimate $\tilde{A}$ is taken at the point where the fitted curve first exceeds 10%:

$$\tilde{A} = b/\delta_k, \quad k = \min\{i : R_{\text{OWD}}(i) > 0.10\} \tag{5}$$

The 10% threshold was selected empirically by evaluating PathAB at thresholds of 5%, 10%, 15%, and 20% across 1.5–15 Mbps bottleneck capacities. At 5%, false triggering on noise caused premature underestimates (error increased by 3–8 percentage points). At 15–20%, the coarse estimate consistently overshot the true value, causing Phase 2 to probe an inefficient rate range. The 10% threshold provided the best trade-off, yielding Phase 1 estimates within 20–30% of the true available bandwidth across all tested capacities—sufficient to centre Phase 2's probing range around the true value.
If Phase 1 fails (e.g., due to excessive packet loss at very low bandwidth), PathAB selects a fallback estimate of $\tilde{A} = R_{\min} \times \gamma^{N/3}$, which centres Phase 2 near the lower third of the probing range. This adaptive fallback avoids a fixed constant and scales with the configured rate range. Using a single train significantly reduces overhead compared with MoSeab's iterative approach, which may require 8–12 trains before its first phase converges.

3.3 Direct Probing Phase

The sender transmits 15 Poisson-distributed probing trains of 30 packets each. These parameters were selected empirically: 15 trains provide sufficient data points for stable linear regression, and 30 packets per train yield reliable mean OWD estimates while keeping per-train duration short. Mean probing rates increase uniformly from $0.85\tilde{A}$ to $2.00\tilde{A}$. Inter-packet gaps within each train follow a Poisson distribution.
Using MoSeab's model, if Δ_{in} is the inter-packet interval, $R_P = b/\Delta_{\text{in}}$ the probing rate, C the tight-link capacity, and A the available bandwidth, the OWD increase between successive packets is:

$$\Delta_{\text{OWD}} = \frac{b}{C} - \frac{A}{C} \cdot \Delta_{\text{in}} = \alpha - \beta \cdot \Delta_{\text{in}} \tag{6}$$

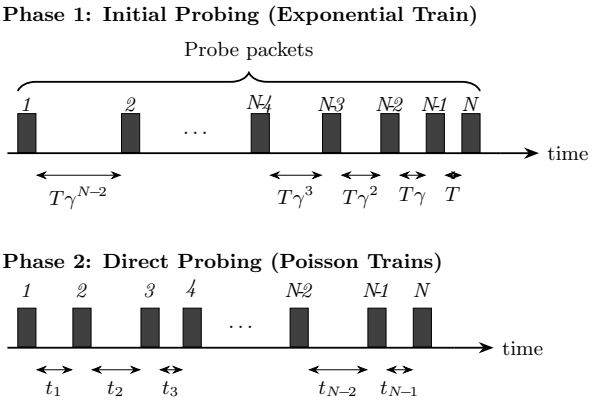

Fig. 1. PathAB two-phase probing structure. Phase 1 (top): exponentially spaced probe packets with decreasing inter-packet gaps $T\gamma^{N-2}, \ldots, T\gamma, T$. Phase 2 (bottom): Poisson-distributed probing trains with random inter-packet gaps. 15 trains of 30 packets each are sent at mean rates from $0.85\tilde{A}$ to $2.00\tilde{A}$.

The linear relationship between Δ_{OWD} and Δ_{in} enables estimation via linear regression. With 15 measurements, α and β are estimated, yielding:

$$A = \frac{\beta}{\alpha} \cdot b, \qquad C = \frac{b}{\alpha} \tag{7}$$

Although Eq. 6 originates from MoSeab's fluid-model analysis, it remains valid under Poisson cross-traffic because the linear relationship holds whenever the *average* output gap is determined by the average cross-traffic rate during the measurement interval. The Poisson-distributed inter-packet gaps in Phase 2 sample the queue state at randomised instants, reducing correlation between successive measurements and improving regression stability compared with periodic probing [5].

3.4 Standalone Mode

Instead of echoing large probe packets, PathAB sends 28-byte ICMP echo packets back-to-back *behind* the large probes. The probe packets are dropped at the destination; the echo packets are bounced back. Because echo packets are very small, they experience minimal reverse-path interference—a key advantage over PoissonProb, which echoes full-size probes.
The echo packet is placed behind the probe packet to prevent gap buildup. A 1200-byte probe takes $t_1 = 1200 \times 8/C$ seconds to traverse a link, while the 28-byte echo takes negligible time. Placing the echo behind ensures it arrives at the router immediately after the probe, preventing cross-traffic from inserting packets between them.
In the initial phase, each exponential-train probe is followed by an echo packet. In the direct phase, only the first and last probes of each train are followed by echo packets. If the gap between the two returning echoes is g, the average output gap is $g/(N-1)$, where N is the train length.
Because Phase 2 in standalone mode measures only the aggregate gap between the first and last echo packets within each train, it trades per-packet OWD granularity for a more robust measurement:

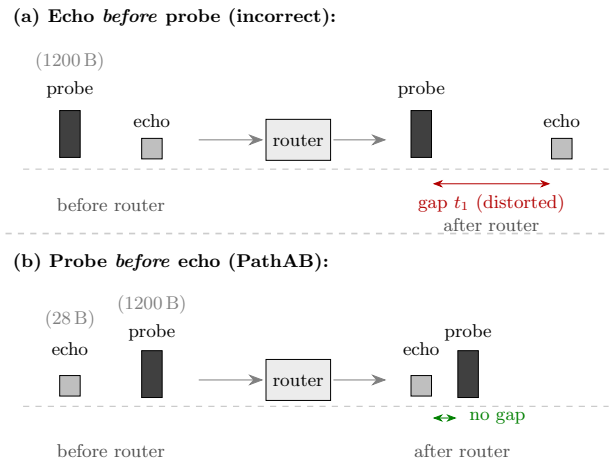

Fig. 2. Standalone echo packet placement. (a) Echo before probe (incorrect): the gap t_1 between probe and echo grows after traversing the router due to the probe's larger serialization delay. (b) Probe before echo (PathAB): placing the small 28-byte echo immediately behind the 1200-byte probe prevents gap buildup.

the aggregate gap averages out individual packet jitter across $N-1$ intervals, yielding a smoother estimate of the mean output gap. This is adequate for the linear regression model (Eq. 6), which requires only the mean OWD increase per train, not per-packet detail.

The standalone mode requires that the destination host responds to ICMP echo requests. In networks where ICMP is filtered, standalone mode cannot operate. Some routers also rate-limit ICMP responses, which could cause echo packet loss at high probing rates.

PathAB was implemented in C++ on Linux using UDP sockets, `clock_gettime()` for nanosecond-resolution timestamps, and `nanosleep()` for inter-packet timing. Full implementation details are available in [16].

3.5 Probing Overhead

PathAB's total probing cost can be estimated from its parameters. With spread factor $\gamma = 1.2$ and a rate range from $R_{\min} = 100$ Kbps to $R_{\max} = 100$ Mbps, Phase 1 sends $N_1 = \lceil \log(R_{\max}/R_{\min})/\log(\gamma) \rceil = 38$ probe packets of 1200 bytes, totalling approximately 45.6 KB. Phase 2 sends $15 \times 30 = 450$ packets of 1200 bytes, totalling 540 KB. In standalone mode, Phase 1 adds 38 echo packets of 28 bytes (1.1 KB) and Phase 2 adds $15 \times 2 = 30$ echo packets (0.8 KB). The total probing load is thus approximately 586 KB (client–server) or 588 KB (standalone). Phase 1 completes in a single train whose duration equals the sum of all inter-packet gaps ($\sum_{i=0}^{N_1-2} T\gamma^i \approx 1.2$ s for typical parameters). Phase 2 sends 15 trains with 200 ms inter-train spacing, completing in approximately 5–7 s depending on $\tilde{A}$. A complete PathAB measurement thus takes 6–8 seconds and injects under 600 KB—substantially less than Pathload's binary search (which may require 12–50 streams of 100+ packets each) and comparable to PathChirp (typically 300–500 KB per chirp sequence).

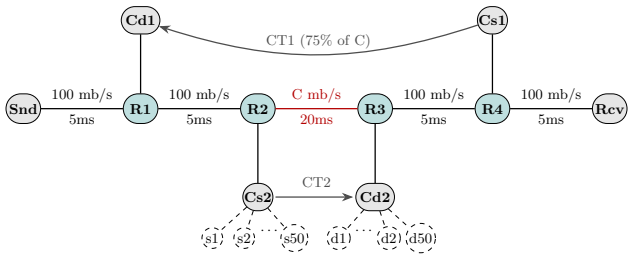

Fig. 3. NS-2 simulation topology. Four routers R1–R4 with bottleneck link R2–R3 (capacity C, 20 ms delay). All other links: 100 Mbps, 5 ms. CT1: reverse-path cross-traffic at 75% of C. CT2: 50 Poisson sub-sources generating bottleneck cross-traffic.

4. EXPERIMENTAL EVALUATION

PathAB was evaluated using NS-2 simulations (NS-2 v2.31, now discontinued but widely used in the bandwidth estimation literature) and network test-bed experiments, compared against PathChirp [8], PoissonProb [7], IGI [9], Spruce [2], Pathload [3], and a stochastic model [11]. The RMS error percentage was the primary metric:

$$\text{Error}\% = \frac{|A - \tilde{A}|}{A} \times 100 \tag{8}$$

Each experiment was repeated 15–20 times under identical traffic conditions but with different random seeds. The RMS error reported in all tables is $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\text{Error}\%_i)^2}$ computed over all n repetitions.

PathAB computes OWD *differences* between successive probe packets at the receiver (Eqs. 3–4), not absolute one-way delays. This differencing eliminates the need for clock synchronization between sender and receiver, since any constant clock offset cancels. Only clock *drift* during a single measurement (<8 s) could introduce error; at typical drift rates (<1 μs/s for modern NTP-synchronised hosts), the accumulated error is negligible.

4.1 NS-2 Simulation Setup

A source (Snd) and receiver (Rcv) are connected through four routers (Fig. 3). The link R2–R3 is the bottleneck with capacity C and 20 ms delay. All other links have 100 Mbps capacity and 5 ms delay. Cross-traffic is generated by 50 Poisson sources. Packet sizes are random between 64 and 1500 bytes. Returning-path cross-traffic (for standalone evaluation) is set at 75% of C. Router queue lengths are set very large to eliminate packet-loss effects. Experiments used bottleneck capacities of 1.5, 5, 10, and 15 Mbps under 20%, 50%, 75%, and 90% utilization.

4.2 Simulation Results: PathAB Across All Capacities

Table 2 summarizes PathAB's RMS error across all bottleneck capacities and utilization levels in both client–server (CS) and standalone (SA) modes.

RMS error is within 10% for client–server mode at utilizations up to 75% and for standalone mode at utilizations up to 50%, across all tested capacities. Error increases at 90% utilization, particularly at 1.5 Mbps where low capacity amplifies probing interference. Client–server mode consistently outperforms standalone mode by 1–3 percentage points, confirming that the small echo packets limit but do not eliminate reverse-path effects.

Table 2. RMS Error (%) of PathAB Under Different Bottleneck Capacities and Utilizations

Bottleneck	Mode	20%	50%	75%	90%
1.5 Mbps	CS	6.60	8.66	13.11	18.75
1.5 Mbps	SA	7.12	10.15	16.15	33.62
5 Mbps	CS	6.38	7.03	9.22	13.68
5 Mbps	SA	8.77	10.89	11.52	25.17
10 Mbps	CS	7.61	6.63	8.41	10.55
10 Mbps	SA	9.16	8.88	9.47	15.33
15 Mbps	CS	7.44	7.75	10.22	8.65
15 Mbps	SA	9.61	9.75	10.50	15.19

CS = client–server; SA = standalone (ret.-path CT = 75%C).

4.3 Simulation Results: Comparative Evaluation

Table 3 compares PathAB against all evaluated algorithms across all four bottleneck capacities. To conserve space, the table reports RMS error at 50% and 90% utilization, which capture the moderate and stressed operating regimes respectively. (Results at 20% and 75% follow similar trends and are available in [16].)

PathAB (CS) consistently achieves the lowest or near-lowest RMS error at every utilization level and capacity. PathChirp is competitive at moderate utilization but degrades at 90%. PoissonProb shows erratic behavior, performing well in some scenarios but exceeding 50% error in others. The stochastic model is competitive at 50% utilization but degrades sharply at 90%. IGI and Pathload exhibit catastrophic failure at high utilization, with errors exceeding 100% in multiple scenarios. PathAB's standalone mode adds a modest error premium of 1–5 percentage points over client–server mode while remaining substantially more accurate than most competing algorithms. Full results at all four utilization levels are available in [16].

Fig. 4 visualises the tabular results on a logarithmic scale. Three observations stand out: (i) PathAB (CS) is the only algorithm whose bars remain within a single order of magnitude of the 10% target across every scenario; (ii) standalone-capable methods split into a robust group (PathAB SA, Stoch. Model) and a fragile group (PoissonProb) that fails at 90% utilization; and (iii) classic gap-based methods (IGI, Pathload) exhibit two-order-of-magnitude error inflation once the tight link saturates.

4.4 NS-2 Simulation: Multi-Hop Cross-Traffic

To evaluate robustness under multi-hop conditions, a four-hop topology was used with bottleneck link R2–R3 at 10 Mbps (CT2 = 3 Mbps, yielding AB = 7 Mbps). Pre-bottleneck cross-traffic (CT1 on R1–R2) and post-bottleneck cross-traffic (CT3 on R3–R4) were varied from 0 to 19 Mbps independently. Table 4 shows selected results.

PathAB (SA) tracks the actual available bandwidth more closely than most methods, particularly when pre-bottleneck traffic shifts the tight link (CT1 $\geq$ 14 Mbps). At CT = 19 (actual AB = 1 Mbps), PathAB estimates 1.43 Mbps (pre) and 1.74 Mbps (post), while Pathload reports 4.50/4.66 and IGI reports 4.83/5.05. PathAB and PathChirp are the most accurate at extreme cross-traffic levels.

4.5 Test-Bed Setup

The test-bed used Cisco 2651xm routers running Cisco IOS 12.3. End hosts were Linux PCs (Fedora Core 5, kernel 2.6.x) connected via Fast Ethernet NICs. Cross-traffic was generated using D-ITG (Distributed Internet Traffic Generator) configured to produce Poisson-distributed UDP flows with uniformly random packet sizes between 64 and 1500 bytes, matching the simulation profile. All algorithms under comparison were run from the same sender host to ensure identical network conditions. Each measurement was initiated after the cross-traffic had stabilised for at least 10 seconds. Experiments were conducted in three configurations:

—**Single-hop (10 Mbps):** Two routers connected by a 10 Mbps link. Cross-traffic varied from 0 to 9 Mbps (AB from 10 to 1 Mbps). Each experiment repeated 20 times.

—**Multi-hop (10 Mbps):** Three routers; R2–R3 at 8 Mbps (bottleneck), R1–R2 and R3–R4 at 10 Mbps. CT2 = 3 Mbps (AB = 5 Mbps). Pre/post-bottleneck traffic varied 0–8 Mbps.

—**Multi-hop (100 Mbps):** All links at 100 Mbps; the bottleneck is traffic-induced by CT2 = 30 Mbps on R2–R3 (AB = 70 Mbps). Pre/post-bottleneck traffic varied 0–90 Mbps.

4.6 Test-Bed Results: Single-Hop

PathAB is the only algorithm that produces estimates when available bandwidth drops below 2 Mbps. While PathChirp and Spruce achieve slightly lower error at AB = 10 Mbps, their error increases rapidly at lower bandwidths (PathChirp: 34.17% at AB = 4; Spruce: 240.53% at AB = 2). PathAB maintains error below 10% for AB $\geq$ 4 Mbps.

4.7 Test-Bed Results: Multi-Hop (10 Mbps)

At CT1 = 8 Mbps (pre-bottleneck), where the pre-bottleneck link becomes the tight link, competing algorithms show dramatic error increases: PathChirp 123.94%, Pathload 158.82%, Spruce 100.36%. PathAB remains at 24.78% (CS)—still a substantial error, but an order of magnitude better than methods that assume a single bottleneck. In post-bottleneck experiments, PathAB (CS) ranges from 6.75% to 18.03%, while PathChirp degrades to 120.91% and PoissonProb fails entirely at CT3 = 8 Mbps.

Fig. 7 plots the post-bottleneck (CT3) columns from Table 6 on a log scale, making the divergence between PathAB and the other methods more apparent. Even at the most adversarial operating point (CT3 = 8 Mbps), PathAB (CS) is 6.7$\times$ more accurate than PathChirp, 6.9$\times$ more accurate than Pathload, and 2.6$\times$ more accurate than IGI. The near-flat slope of PathAB below CT3 = 4 Mbps confirms that the two-phase estimator correctly identifies R2–R3 as the tight link even when a competing link carries substantial traffic.

4.8 Test-Bed Results: Multi-Hop (100 Mbps)

PathAB estimates in both modes remain closest to the actual available bandwidth across all traffic levels. PathChirp consistently overestimates (e.g., 76.75 Mbps when actual is 70 Mbps in post-bottleneck). IGI shows large deviations when the tight link shifts, estimating 50.19 Mbps when the actual AB is 40 Mbps. PathAB's RMS error remains within 10–12% in most 100 Mbps scenarios. For the 100 Mbps experiments, PathAB used 1500-byte probe packets (see Sec. 4.9) to ensure sufficient serialisation delay at higher link speeds. The `clock_gettime()` timestamps provided sub-microsecond resolution, adequate for measuring inter-packet gaps of 120 μs (1500 bytes at 100 Mbps).

Table 3. RMS Error (%) Comparison Across All Bottleneck Capacities (50% and 90% Utilization)

Algorithm	**1.5M 50%**	**1.5M 90%**	**5M 50%**	**5M 90%**	**10M 50%**	**10M 90%**	**15M 50%**	**15M 90%**
PathAB (CS)	8.66	18.75	7.03	13.68	6.63	10.55	7.75	8.65
PathAB (SA)	10.15	33.62	10.89	25.17	8.88	15.33	9.75	15.19
PathChirp	12.94	30.63	10.06	17.39	10.54	23.86	14.82	17.90
Spruce	19.40	39.47	23.81	33.47	11.18	33.77	18.71	29.10
PoissonProb	11.78	52.95	22.01	62.47	26.56	57.77	12.64	35.63
Stoch. Model	11.98	31.60	15.44	25.27	8.91	36.36	8.33	25.46
IGI	34.91	219.06	40.39	204.78	43.76	135.11	31.91	93.92
Pathload	50.51	443.75	22.21	263.50	31.80	192.03	18.20	176.74

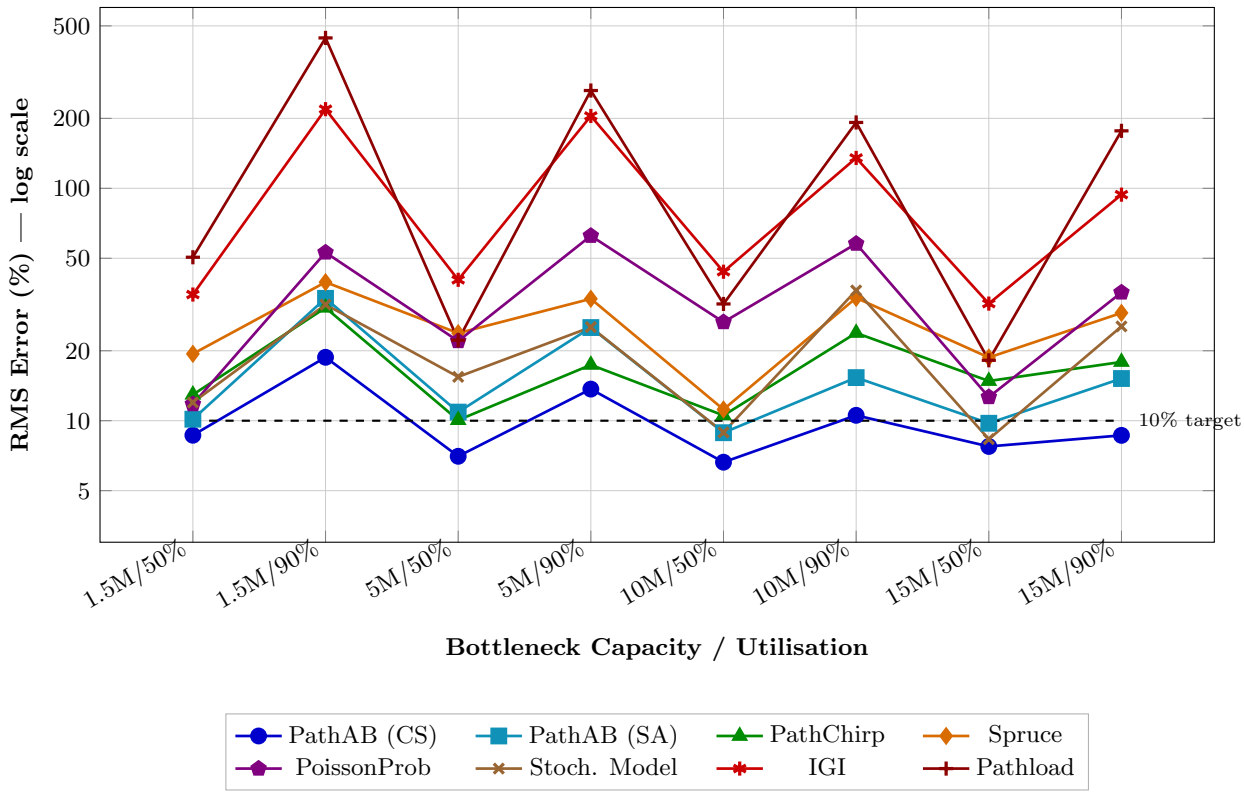

Fig. 4. Comparative RMS error (log scale) across all evaluated algorithms, bottleneck capacities (1.5–15 Mbps), and utilizations (50% / 90%). The dashed line marks the 10% engineering-accuracy target. PathAB (CS) is the only method that stays close to the 10% target at every operating point; IGI and Pathload spike above 100% under stressed (90%) utilization.

4.9 Probe Packet Size

Following Pasztor and Veitch [17], PathAB uses 1200-byte probes for 10 Mbps paths and 1500-byte probes for 100 Mbps paths. Smaller packets may not create enough measurable delay variation, while larger packets increase intrusiveness. The selected sizes correspond to the observed saturation points for these link capacities.

5. DISCUSSION

The results reveal several consistent patterns across all experimental configurations:

Accuracy stability. PathAB (CS) maintains RMS error below 10% in the large majority of simulation and test-bed scenarios. The standalone mode adds a modest premium of 1–5 percentage points. By contrast, most competing algorithms show dramatic error increases under high utilization or when the tight link differs from the bottleneck link.

Graceful degradation. While all algorithms lose accuracy at 90% utilization or very high cross-traffic, PathAB's degradation is gradual (e.g., from 8% to 15%), whereas IGI and Pathload exhibit catastrophic failure with errors exceeding 100%.

Low-bandwidth robustness. In the single-hop test-bed, PathAB is the only algorithm that produces estimates when AB drops below

Table 4. Average AB Estimate (Mbps) Under Pre- and Post-Bottleneck Cross-Traffic (Actual AB = 7 Mbps for CT $\leq$ 13)

CT	Actual	Scen.	Pathload	IGI	Stoch.	PoissonP.	Spruce	PathChirp	PathAB (SA)
0	7	Pre	8.01	6.87	8.00	6.25	7.18	7.45	6.73
8	7	Pre	8.06	7.79	7.28	5.59	5.57	7.11	6.69
15	5	Pre	6.79	6.68	4.70	4.63	5.93	4.65	4.84
19	1	Pre	4.50	4.83	4.61	3.70	3.67	2.11	1.43
0	7	Post	8.01	6.87	8.00	6.57	7.18	7.45	6.73
8	7	Post	8.15	7.28	6.84	7.63	7.48	5.95	6.33
15	5	Post	6.69	5.04	6.21	4.34	4.91	5.47	4.49
19	1	Post	4.66	5.05	5.14	3.82	3.99	2.18	1.74

Selected rows shown; full data at all 20 CT levels per scenario available in [16].

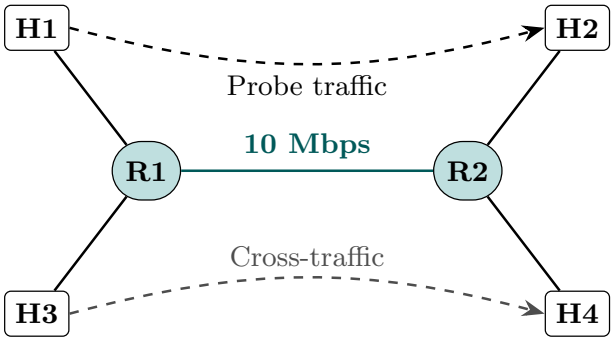

Fig. 5. Single-hop test-bed topology. Two Cisco 2651xm routers (R1, R2) connected by a 10 Mbps link. H1→H2: probe traffic. H3→H4: cross-traffic (0–9 Mbps).

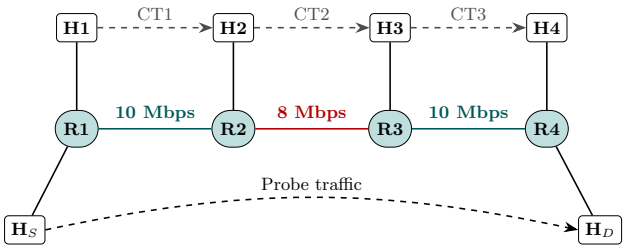

Fig. 6. Multi-hop test-bed topology. R1–R2 and R3–R4 at 10 Mbps; bottleneck R2–R3 at 8 Mbps (red). CT1, CT2, CT3: independent cross-traffic flows. $H_S \rightarrow H_D$: probe traffic.

2 Mbps. This is attributable to its Phase 1 fallback mechanism, which provides a reasonable seed estimate even under extreme conditions.

Multi-hop resilience. The pre-bottleneck and post-bottleneck experiments demonstrate that PathAB handles tight-link shifts better than competing methods. When pre-bottleneck cross-traffic at CT1 = 8 Mbps makes the pre-bottleneck link the tight link, PathChirp's error jumps to 123.94% while PathAB remains at 24.78%.

Standalone viability. The consistent gap between CS and SA mode (1–5 percentage points) confirms that the small echo packet design effectively limits reverse-path distortion, making standalone mode practical for real deployments.

Table 5. RMS Error (%) — Single-Hop (10 Mbps)

Algorithm	10	8	6	4	2	1
PathAB (CS)	5.59	6.76	7.87	7.98	14.58	23.99
PathAB (SA)	5.72	7.26	9.62	8.87	16.59	24.39
PoissonProb	9.73	12.10	10.36	15.93	24.99	—
PathChirp	4.37	7.91	16.36	34.17	25.44	—
IGI	18.79	15.55	10.38	17.01	71.32	—
Spruce	5.24	6.22	24.53	60.45	240.53	—
Pathload	14.00	4.55	14.20	49.16	209.77	—

"—" = algorithm failed. Column headers: AB in Mbps.

Table 6. RMS Error (%) — Multi-Hop Pre- and Post-BN (10 Mbps)

	Pre-BN (CT1)			Post-BN (CT3)		
Algorithm	0	4	8	0	4	8
PA (CS)	6.75	8.10	24.78	6.75	11.13	18.03
PA (SA)	6.86	9.35	25.15	6.89	11.32	22.69
PoissonP.	9.24	11.87	38.87	9.24	12.33	—
PathChirp	12.32	14.87	123.94	12.32	12.35	120.91
IGI	5.92	9.64	60.04	5.92	8.50	46.78
Spruce	10.20	19.68	100.36	10.20	13.08	32.36
Pathload	22.46	28.32	158.82	22.46	19.50	124.92

PA = PathAB. "—" = failed. CT in Mbps. Full data in [16].

Structural advantages. PathAB's accuracy advantage appears attributable to three design choices that complement each other. First, the exponential chirp in Phase 1 covers a wide rate range in a single train, avoiding MoSeab's iterative overhead while still providing a reasonable seed estimate. Second, Phase 2's Poisson-distributed probing reduces temporal correlation between successive packets, yielding more independent samples for the regression and reducing sensitivity to short-term traffic bursts. Third, the two-phase architecture ensures that Phase 2 probes only the neighbourhood of the true available bandwidth, concentrating measurement effort where it matters most rather than wasting packets at irrelevant rates. Together, these properties offer a plausible explanation for why PathAB degrades gradually under stress while single-technique methods (PathChirp, Spruce, Pathload) can fail catastrophically when their operating

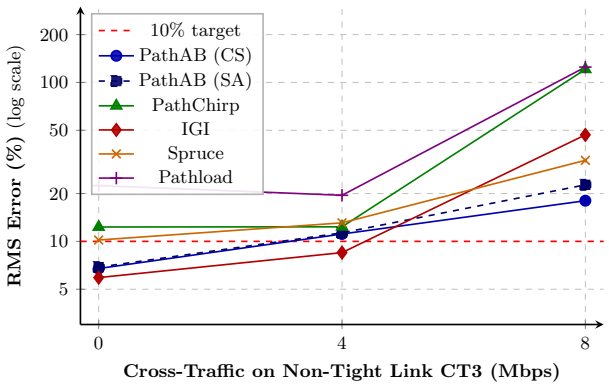

Fig. 7. Multi-hop 10 Mbps test-bed: RMS error (log scale) versus non-tight-link cross-traffic (CT3). Only PathAB (both modes) remains within a small constant factor of the 10% target across the full CT3 range; competing algorithms exhibit near-vertical error growth beyond CT3 = 4 Mbps.

Table 7. Avg. AB Estimate (Mbps) — Multi-Hop (100 Mbps)

CT	Act.	PA-SA	PA-CS	PP	PC	IGI
Pre-bottleneck (CT1 on R1–R2)						
0	70	67.76	69.58	73.25	67.38	60.88
60	40	39.60	38.97	37.73	45.94	50.19
90	10	14.56	13.67	13.76	17.13	28.47
Post-bottleneck (CT3 on R3–R4)						
10	70	68.95	68.32	72.43	76.75	50.44
50	50	46.88	48.01	47.97	53.09	55.30
90	10	8.92	12.34	12.76	15.55	21.77

PA = PathAB; PP = PoissonProb; PC = PathChirp.

Spruce/Pathload excluded (tool limit). Full data in [16].

assumptions are violated. A formal decomposition of Phase 1 vs. Phase 2 contributions is left for future work.

Limitations. The simulations used infinite router queue lengths, which eliminates packet-loss effects present in real networks. With finite queues and tail-drop or RED policies, probe packets may be lost during high utilization, degrading Phase 2's regression fit. The algorithm assumes FIFO queuing at all routers; non-FIFO schedulers such as WFQ or DiffServ could alter the delay signature that PathAB relies on. The experiments were conducted at up to 100 Mbps using Cisco 2651xm routers (circa 2003); while the underlying delay-based measurement principles are hardware-agnostic, validation on modern switches with cut-through forwarding and on Gbps/10Gbps links remains open.

The standalone mode requires ICMP echo support and assumes stable return-path routing during measurement. ICMP rate-limiting, commonly enabled on enterprise routers, could cause echo-packet loss at high probing rates, though PathAB's Phase 2 sends only two echo packets per train, well within typical rate limits ($\geq$100 pps). All competing algorithms used in the comparison predate 2006; however, they remain the most widely cited and experimentally validated active probing tools in the literature. Recent ML-based and telemetry-based approaches [14, 13] target different deployment models and were not directly comparable.

The implementation requires manual parameter adjustment for different link capacities.

6. CONCLUSION

This paper presented PathAB, a hybrid algorithm for end-to-end available bandwidth estimation. PathAB combines exponential chirp probing for coarse estimation, Poisson-distributed probing for refinement, and MoSeab's linear regression model for computation. It supports both client–server and standalone modes, with the standalone mode using small echo packets to minimize reverse-path interference.

Evaluation through NS-2 simulations across four bottleneck capacities and four utilization levels, and through network test-bed experiments in single-hop, multi-hop (10 Mbps), and multi-hop (100 Mbps) configurations, demonstrates that PathAB achieves lower error than PathChirp, PoissonProb, IGI, Spruce, Pathload, and a stochastic model in the majority of tested scenarios. In client–server mode, PathAB's RMS error remains within 10% at utilizations up to 75%; in standalone mode, within 10% at utilizations up to 50%. Competing methods frequently exceed 30–100% under high utilization or multi-hop conditions.

Future work includes automatic parameter selection, validation on high-speed (Gbps) links, evaluation under non-Poisson traffic models, and adaptation for modern environments including cloud, SDN, and edge computing where lightweight standalone bandwidth estimation remains practically relevant. In such environments, PathAB's low overhead (under 600 KB, 6–8 seconds) and standalone capability could make it a useful starting point for on-demand path quality assessment by applications such as adaptive streaming clients and overlay routing protocols, pending validation at higher link speeds.

ACKNOWLEDGMENT

The author gratefully acknowledges the guidance and support of the late Dr. Akshai K. Aggarwal, School of Computer Science, University of Windsor, who co-developed the original algorithm and supervised the thesis research on which this paper is based.

7. REFERENCES

[1] R. L. Carter and M. E. Crovella, "Dynamic server selection using bandwidth probing in wide-area networks," Boston Univ., Comput. Sci. Dept., Tech. Rep. TR-96-007, 1996.

[2] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *Proc. 3rd ACM SIGCOMM Conf. Internet Measurement*, 2003, pp. 39–44.

[3] M. Jain and C. Dovrolis, "Pathload: a measurement tool for end-to-end available bandwidth," in *Proc. Passive and Active Measurements (PAM) Workshop*, 2002.

[4] D. Kiwior, J. Kingston, and S. Akhtar, "PATHMON, a methodology for determining available bandwidth over an unknown network," in *IEEE/Sarnoff Symp. Advances in Wired and Wireless Communication*, 2004, pp. 27–30.

[5] J. Cao, W. S. Cleveland, D. Lin, and D. X. Sun, "Internet traffic tends toward Poisson and independent as the load increases," in *Nonlinear Estimation and Classification*. New York, NY, USA: Springer, 2002, pp. 83–109.

[6] M. Zhang, C. Luo, and J. Li, "Estimating available bandwidth using multiple overloading streams," in *IEEE Int. Conf.*

Communications (ICC), vol. 2, Istanbul, Turkey, 2006, pp. 495–502.

[7] L. Xin, "PoissonProb: a new rate-based available bandwidth measurement algorithm," M.S. thesis, School of Comput. Sci., Univ. of Windsor, Windsor, ON, Canada, 2005.

[8] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrel, "pathChirp: efficient available bandwidth estimation for network paths," in *Proc. Passive and Active Measurement Workshop*, 2003.

[9] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 6, pp. 879–894, 2003.

[10] A. Botta, S. D'Antonio, A. Pescapé, and G. Ventre, "BET: a hybrid bandwidth estimation tool," in *Proc. 11th Int. Conf. Parallel and Distributed Systems (ICPADS)*, vol. 2, 2005, pp. 520–524.

[11] S. R. Kang, X. Liu, M. Dai, and D. Loguinov, "Packet-pair bandwidth estimation: stochastic analysis of a single congested node," in *Proc. 12th IEEE Int. Conf. Network Protocols (ICNP)*, 2004, pp. 316–325.

[12] E. Hemmati and S. Bhattacharyya, "Challenges of available bandwidth estimation in high-speed networks," in *Proc. IEEE Int. Conf. Communications (ICC)*, 2017, pp. 1–6.

[13] Y. Zhu *et al.*, "Packet-level telemetry in large datacenter networks," in *Proc. ACM SIGCOMM*, 2015, pp. 479–491.

[14] Q. Yin, J. Kaur, and F. D. Smith, "Can machine learning benefit bandwidth estimation at ultra-high speeds?," in *Proc. Passive and Active Measurement Conf. (PAM)*, 2020, pp. 49–64.

[15] C. Grigorescu, L. Lorenzi, and N. Cascarano, "Bandwidth prediction using machine learning for SDN-based networks," in *Proc. IEEE/IFIP Network Operations and Management Symp. (NOMS)*, 2020, pp. 1–5.

[16] D. Roy, "PathAB: a new method to estimate end-to-end available bandwidth of network path," M.S. thesis, School of Comput. Sci., Univ. of Windsor, Windsor, ON, Canada, 2009.

[17] A. Pasztor and D. Veitch, "The packet size dependence of packet pair like methods," in *Proc. 10th IEEE Int. Workshop Quality of Service*, 2002, pp. 204–213.