Scaling Issues for the Integrated Services Internet

Henning Schulzrinne and Jonathan Rosenberg
Dept. of Computer Science
Columbia University
schulzrinne@cs.columbia.edu

DIMACS: Workshop Architecture and Algorithmic Aspects of Communication Networks
DIMACS Center, CoRE Building, Rutgers University

January 27, 1997

Overview

- Internet integrated services architecture
- scaling:
  - multicast
  - resource reservation
  - transport

Internet Integrated Services Architecture

link-level: scheduling, link-layer-specific

IP multicast: trade-off state vs. tree optimality

reservation: set up scheduling state

transport: timing, QOS feedback

applications: playout delay adjustment, error concealment, bit-rate adaptation, ...

philosophy: work both with and without resource reservation ➠ robustness

Multicast

Issues:

- refresh, prune traffic
- channel surfing, DIS ➠ linger time after leaving
- state: per group G, per sender S

assumption “few senders” wrong (RTCP)

Multicast

DVMRP, dense-mode PIM: send unless told otherwise:

- prune timeout ➠ periodic flooding
- S, G state

sparse-mode PIM: rendezvous points (RPs)

- RP selection?
- source, group aggregation?
- single tree (G) $\rightarrow$ source-based tree S, G

CBT: single tree ➠ only G state

Resource Reservation

- soft-state: state disappears unless periodically refreshed
- per-flow state?
- per-flow scheduling?
- variable aggregation in backbone and leaves – “Internet VPs”?
- make refresh implicit in data traffic?

Alternatives

- voice, CBR interactive video:
  - deterministic
  - very delay-sensitive
  - low bandwidth traffic ➠ 20,000 flows in OC-12
  - probably small fraction ($\rho < 0.5$) of link
- ➠ pack into single high-priority class

RTP

Data:

- end-to-end transport of streams with timing information
- labeling: source, content
- encryption, sequencing, loss detection, ...

Control = RTCP:

- member names:
  - member list
  - loop detection
  - multiple media $\rightarrow$ single conference
- sender reports ➠ rate estimation, lip-sync
- receiver reports ➠ loss, jitter, delay feedback

RTCP Algorithm

- RTCP packets sent periodically
- in $[1.25\mathrm{s}, 3.75\mathrm{s}]$ after joining
- media bandwidth b, RTCP packet size p, session size N
- period

$$T = \min(5\mathrm{s}, \frac{pN}{0.05b \cdot 0.75})$$

- compute $\rightarrow$ set timer $\xrightarrow{T}$ send
- randomized $[0.5, 1.5] \leftrightarrow$ packet synchronization

RTCP Scaling Problems

works fine for high-speed networks, smaller groups (< 1000)

initial transient: lots join at the same time: “movie at ten”, channel surfing ➠ flooding, announcements lost

state storage: each receiver stores whole receiver list

slow estimate: learn only $\leq 1/\mathrm{s}$ for 28.8 kb/s

QOS transients: can’t distinguish 50% loss, 1 minute vs. 5%, 10 minutes

RTCP Transient Suppression

Reconsideration: if N at send instant $> N$ at timer computation, recompute T and check interval

Aggressive reconsideration: recompute even if N hasn't increased

Announcement: accelerate convergence ➠

- send in RTCP packet: observed number of users
- compute $T = f(\max(\text{observed}, \text{announced}))$
- track size announcement for each member

Characteristics:

Learning curve: $L(t)$ = estimate at each member: f(loss,

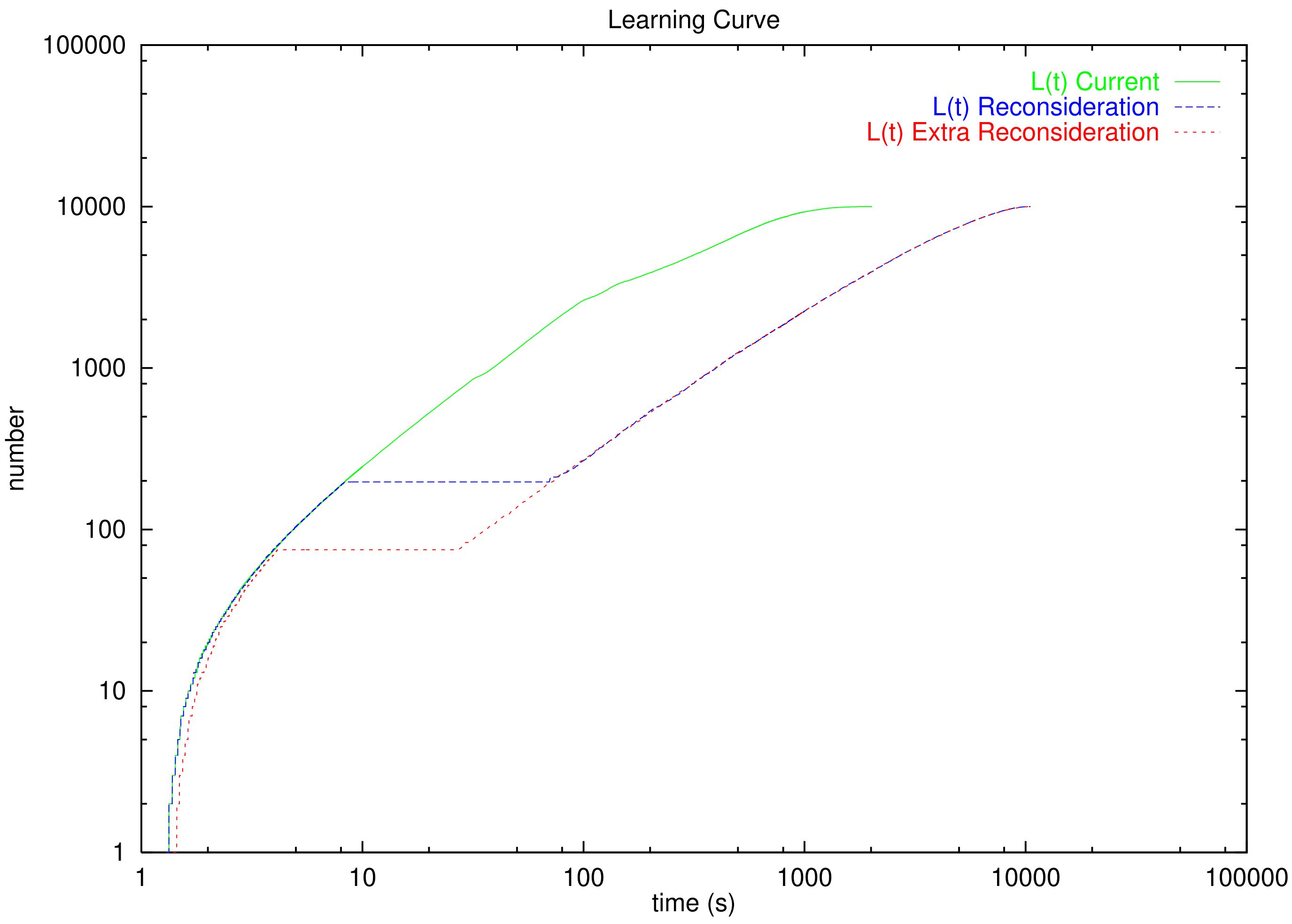
Packets sent: ideally: = $L(t)$

Reconsideration: Experiments

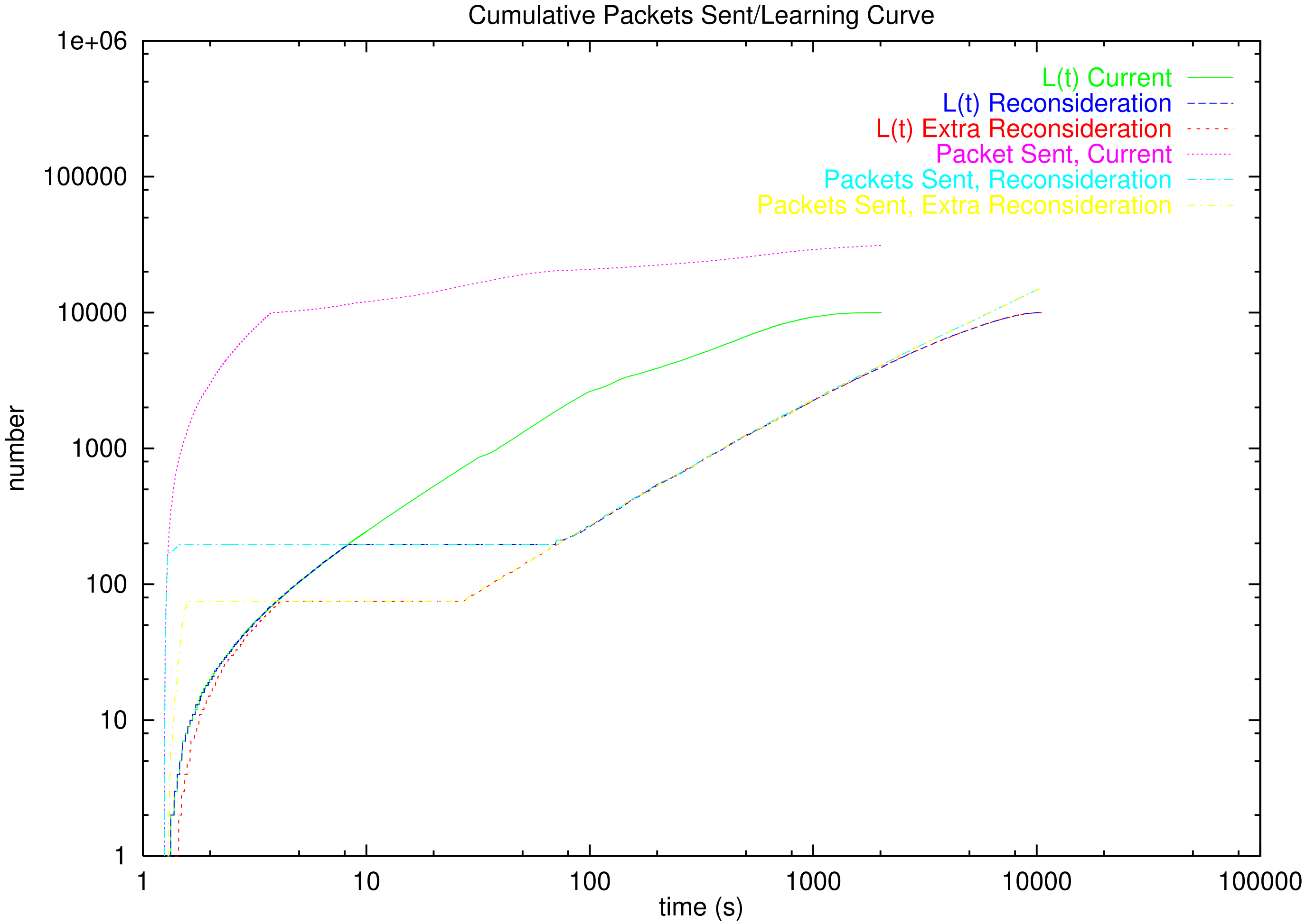
- 10,000 modem (28.8 kb/s) users in group
- join at $t = 0$ or 200 users every 200 ms
- 300 ms average delay, uniformly on [0 ms, 600 ms]
- 100 kB network buffer
- 128 byte RTCP packets

Note: backwards-compatible with existing spec

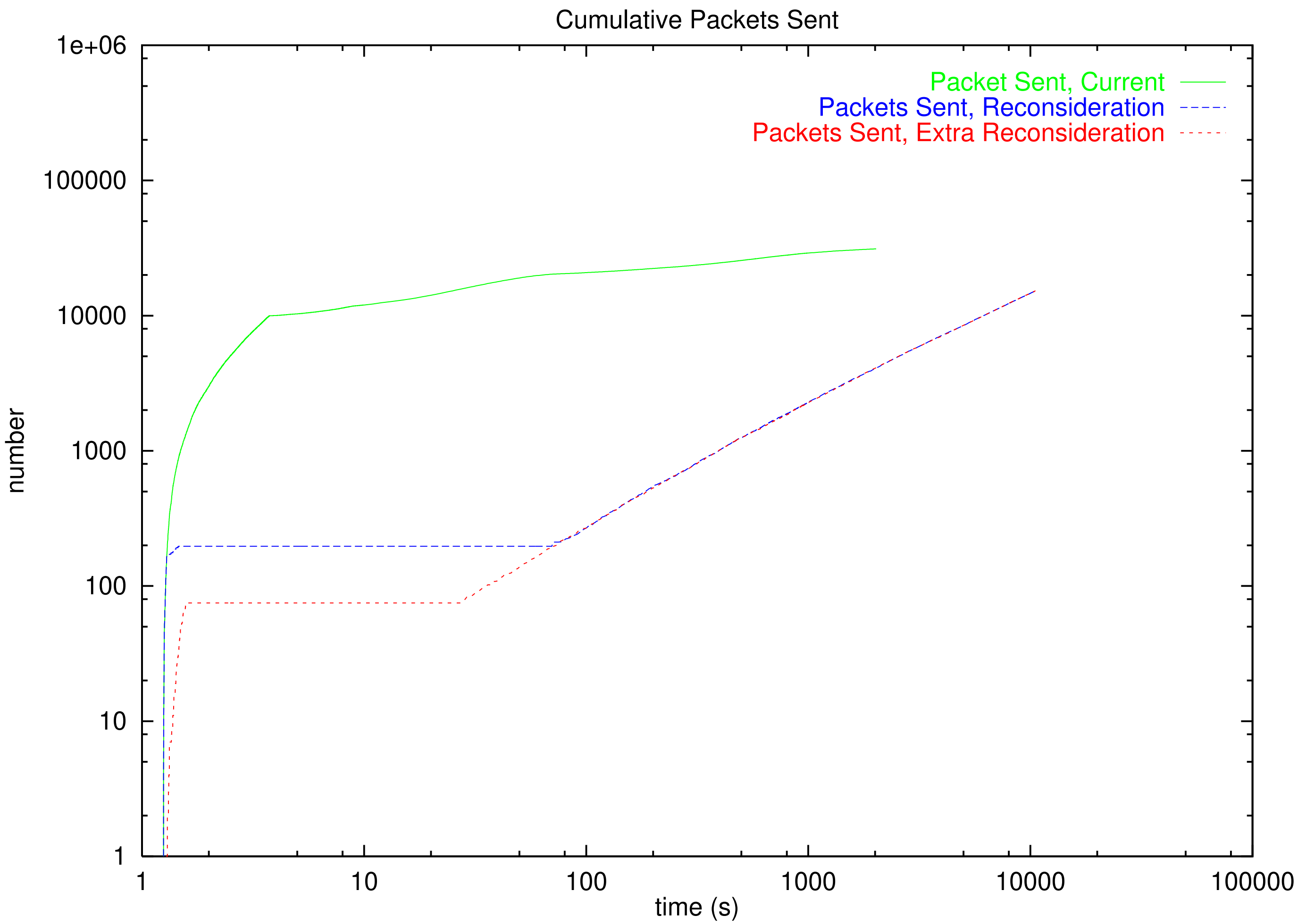
Reconsideration: Step Join



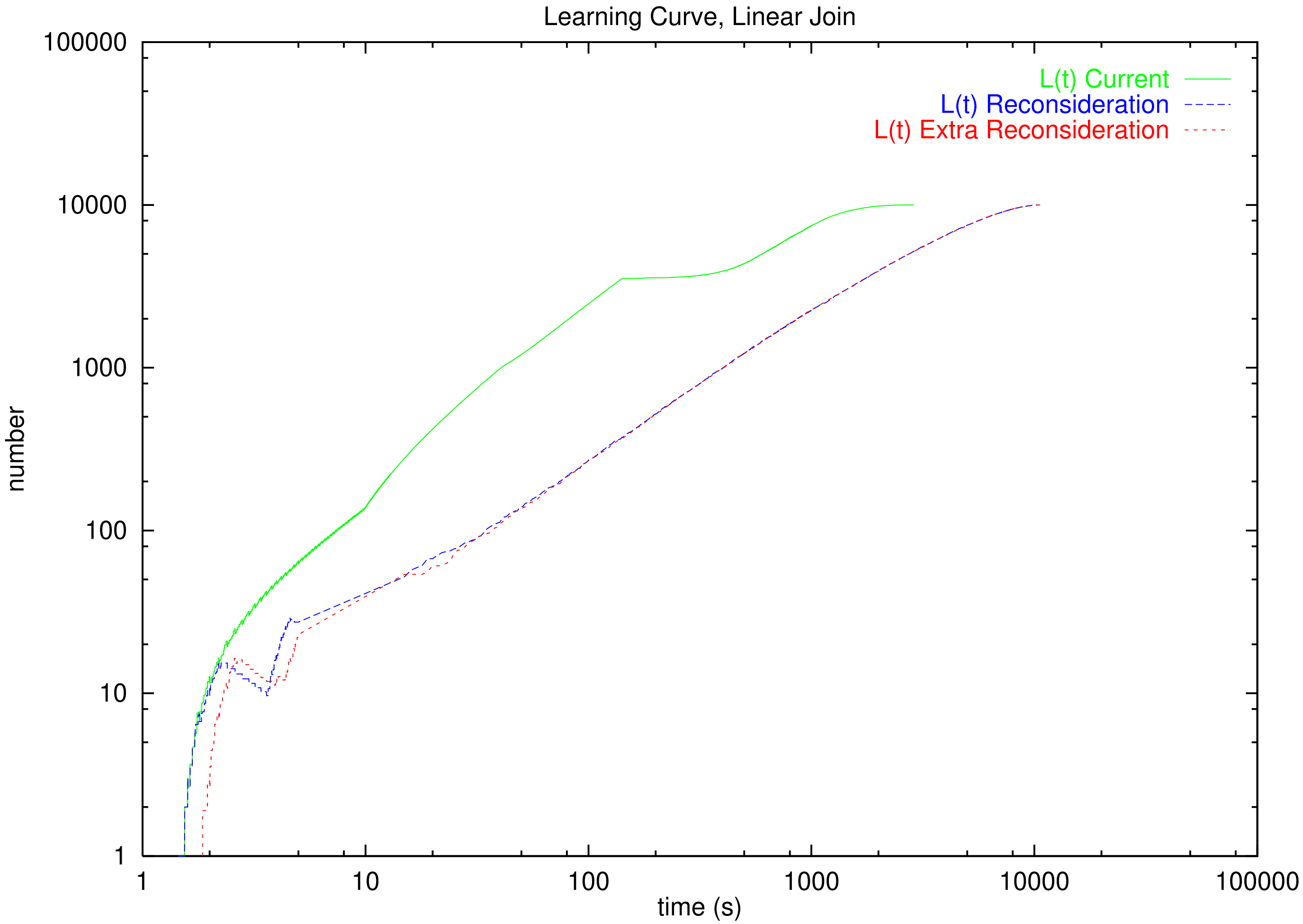
Reconsideration: Step Join



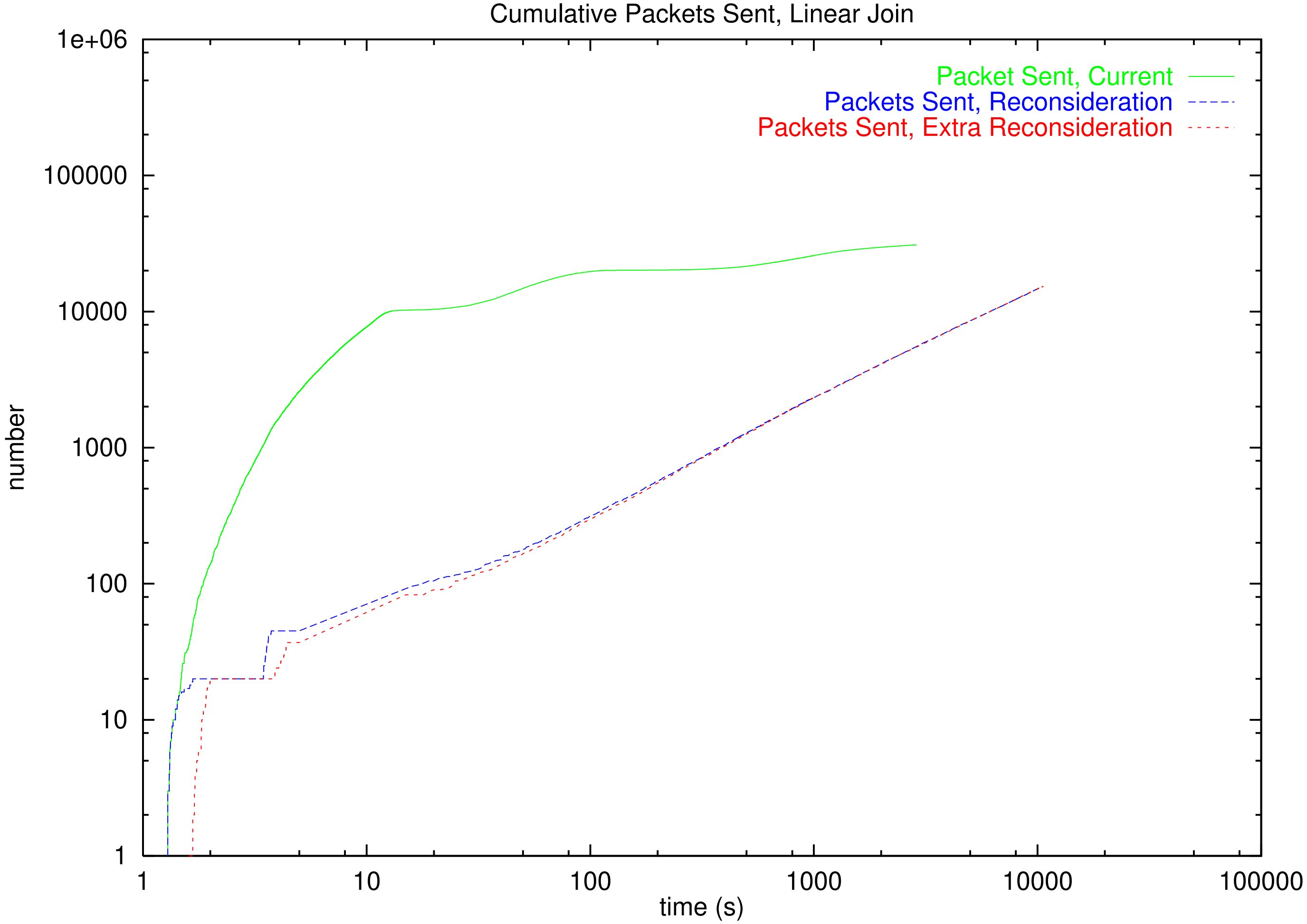
Reconsideration: Step Join



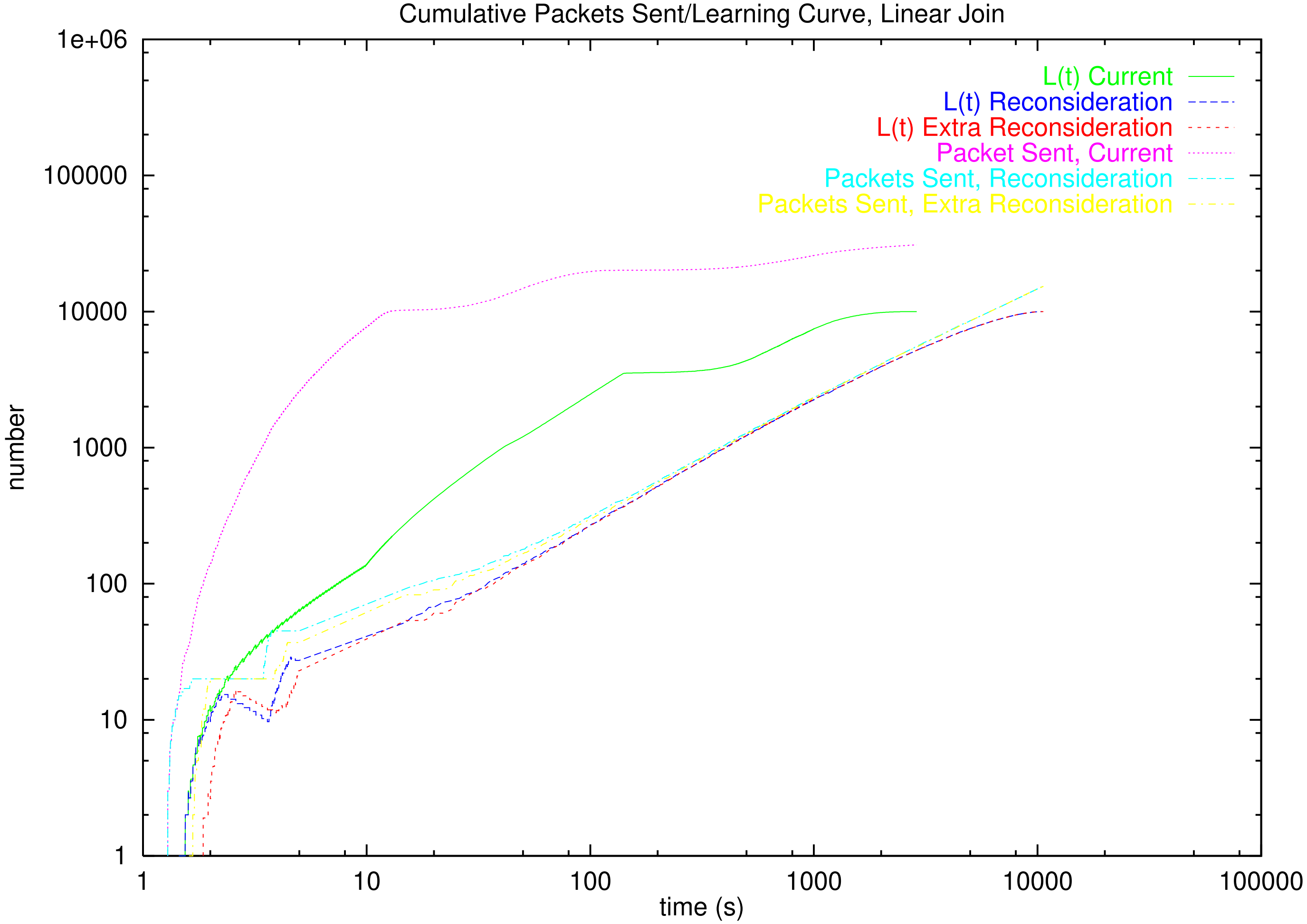
Reconsideration: Linear Join



Reconsideration: Linear Join



Reconsideration: Linear Join



RTCP State Maintenance

- need several MB to count users
- sources identified by 32-bit SSRC, random
- maintain fixed sampling buffer
- keep only fraction of announcements (bit mask, range, threshold)
- if buffer about to overflow:
  - reduce acceptance probability
  - remove non-matches from buffer
- if below threshold, increase probability, maintain count

RTCP Group Size Estimation

- only get to know 3,600 members/hour ➠ not suitable for CATV
- often, only sender cares about group size, QOS ➠ no multicast needed
- may want to get repeat QOS from same member

➠ polling instead of announcements

- send match and mask
- respond only if matched
- allows repeat sampling or new

Summary

- scaling issues at all layers of Internet integrated services architecture
- but: no worse and more choices than WWW or ATM…
- RTCP scaling can be fixed
- applicable to other distributed learning problems?
- e.g., McCanne/Jacobson video adaptation