Search Party:

Using Randomcast for Reliable Multicast with Local Recovery

Adam M. Costello
Steven McCanne

University of California at Berkeley

Infocom'99
1998-Mar-25-Thu

Multicast Loss Recovery

Problem: **repair scope**

- multicast to whole group does not scale
- unicast from source does not scale
- only routers know topology
- no one knows where loss occurred

End-to-end approaches generally organize members into a **hierarchy**. (RMTP, LMS, OTERS, LGC, RMA)

LMS uses a network-constructed hierarchy and:

- scales to arbitrarily large groups
- uses network links efficiently
- makes small demands on routers
- usually sends few unsolicited repairs
- sends no duplicate repairs
- recovers from losses quickly

Can we get similar results without a hierarchy? Yes.

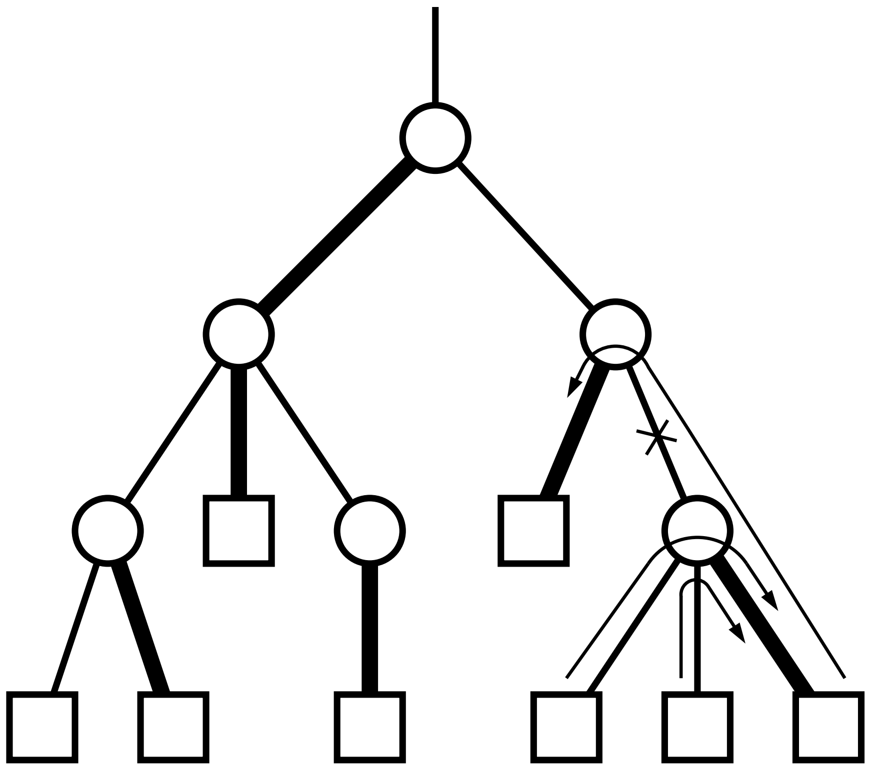
Outline

- LMS (Papadopoulos)
- Randomcast
- Search Party
- Simulation Results
- Conclusions

LMS (Papadopoulos et. al. Infocom'98)

New forwarding service for requests: **parentcast**

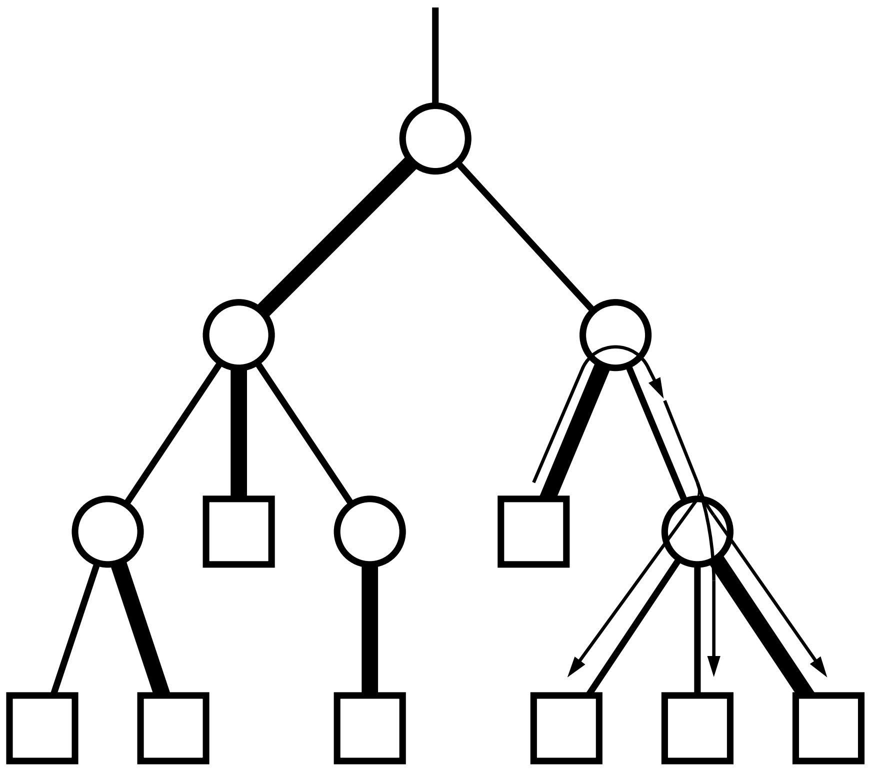
Router chooses one downward *replier link*.
Forward requests to replier links, except:
Forward requests from replier links upward.
Insert *turning point* link ID into request.



LMS (cont.)

New forwarding service for responses: **subcast**

Response goes to subtree below recorded link.

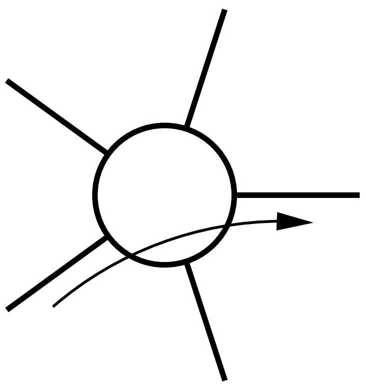


Hosts need not know topology.
Routers need not know protocol.
Many requests, but always one response!
Fast—only one round-trip.

But hierarchy concentrates responsibility.

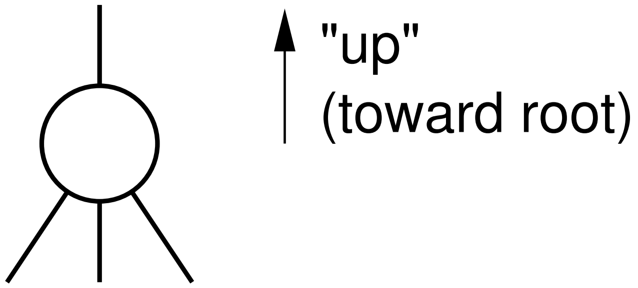
Randomcast

New forwarding service: **randomcast**



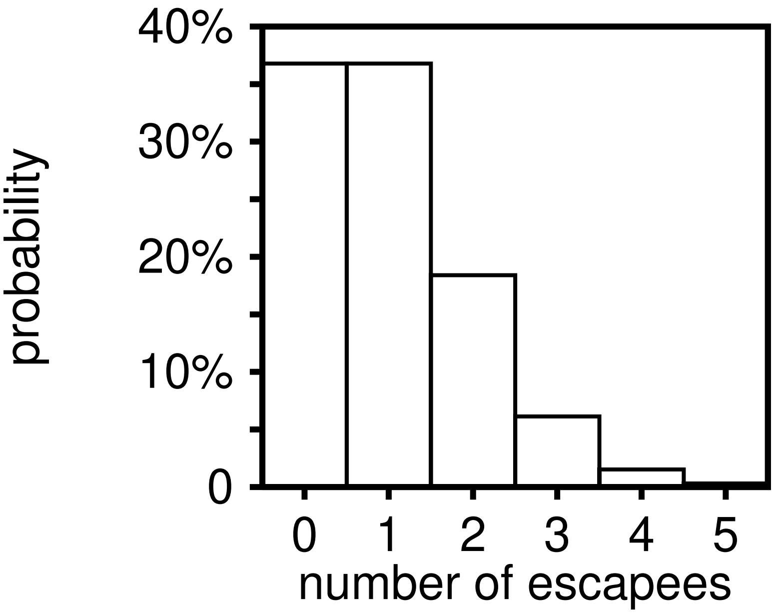
Forward to random on-tree edge, but don't backtrack.

Randomcast (cont.)

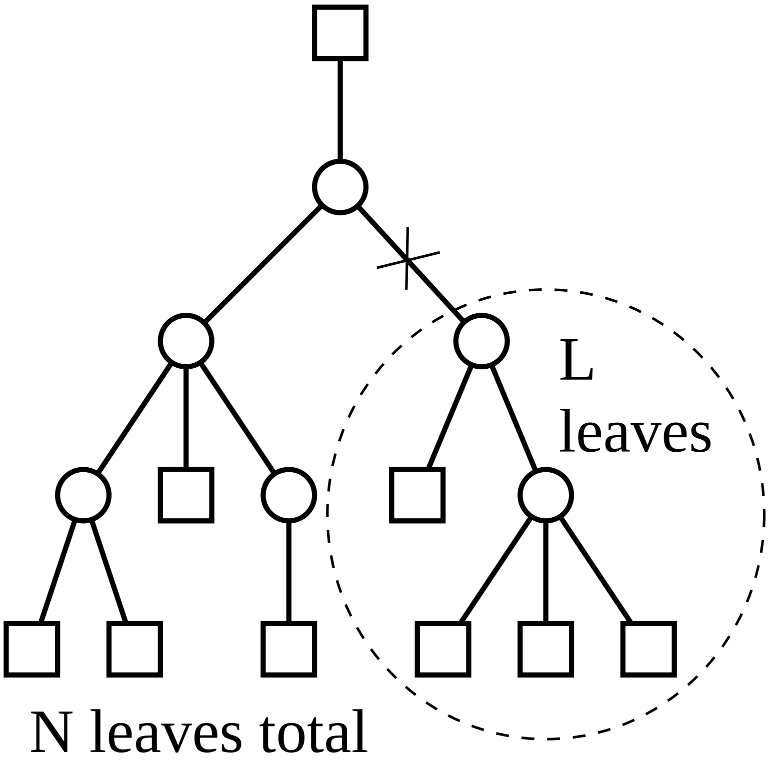


From below (edge E) $\Rightarrow$
Forward up w/prob leaves(E)/leaves(self)
else forward to any other down-edge (uniformly)
and write location info into packet.

From above $\Rightarrow$ Forward to any downward edge (uniformly).



Search Party



- randomcast requests at rate μ (Poisson)
- subcast responses, like LMS
- request traffic (mean):

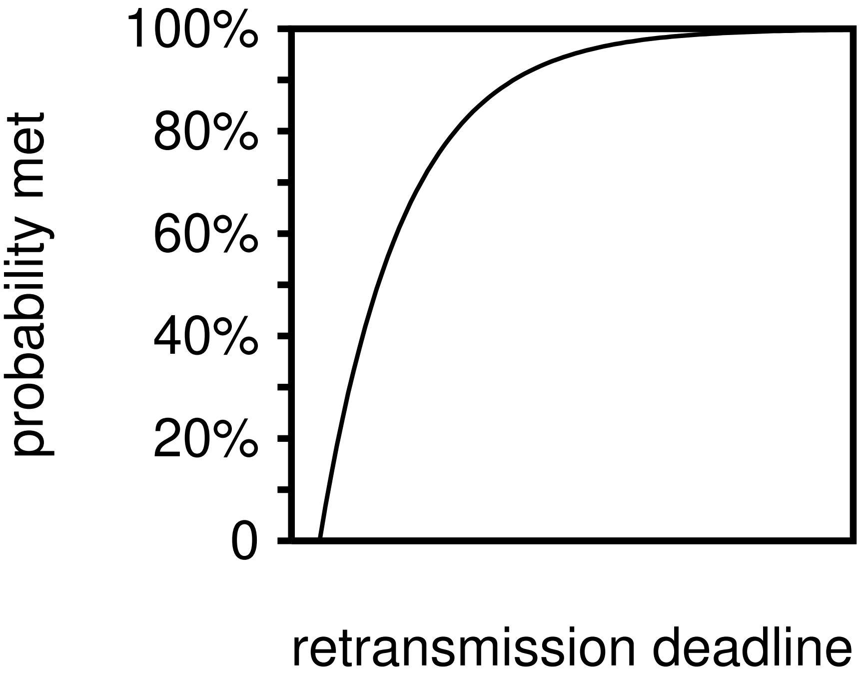
  outside loss subtree: μ total

  inside, upward: μ on each link

  inside, downward: $< 2\mu$ on each link
- response locality: mean exposure $\leq 1 + \ln \frac{N}{L}$

Search Party (cont.)

- retransmission delay:
  depends on RTT, μ
  not on L, N

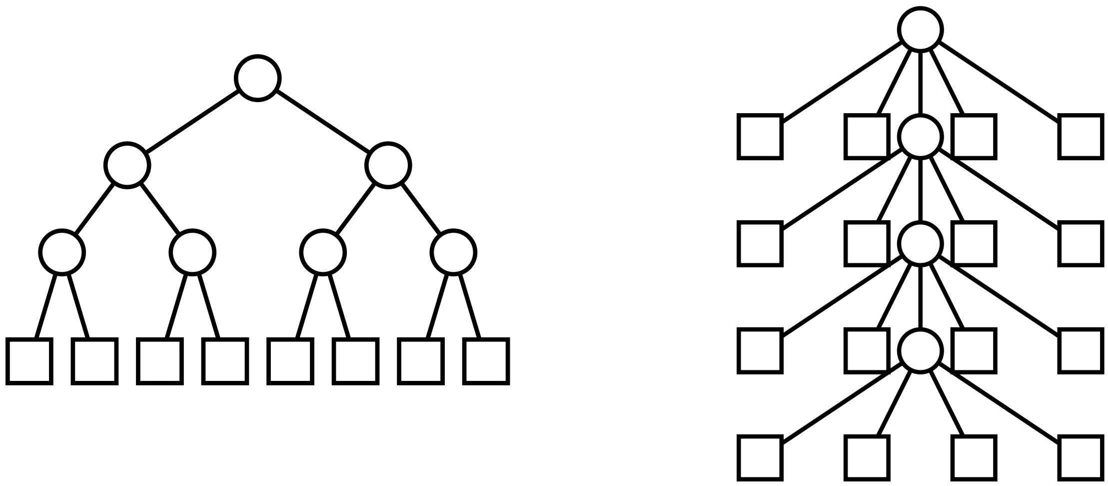


- tunable delay/overhead tradeoff via μ
- responsibility is diffuse (no one is important)

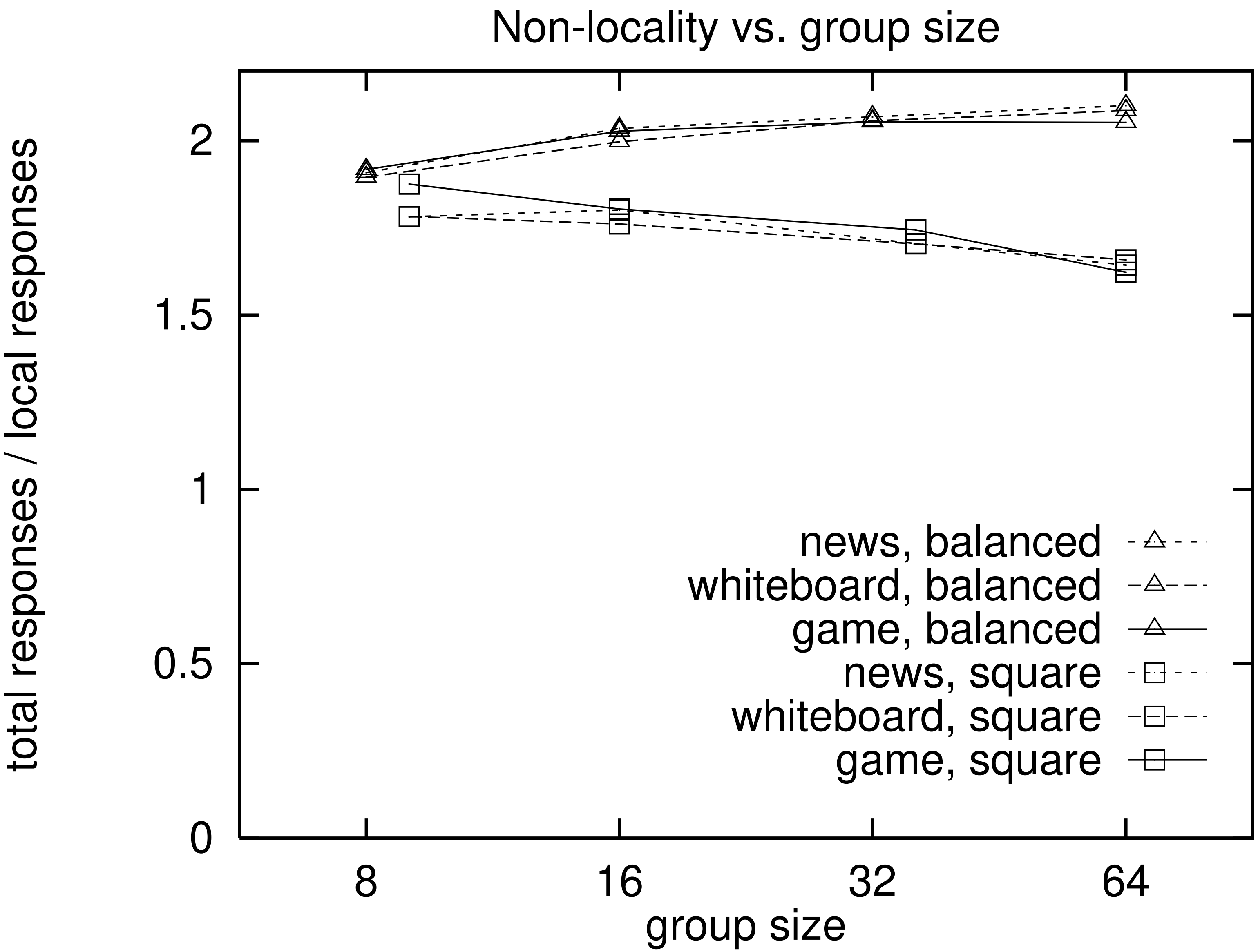
Simulation (**ns**)

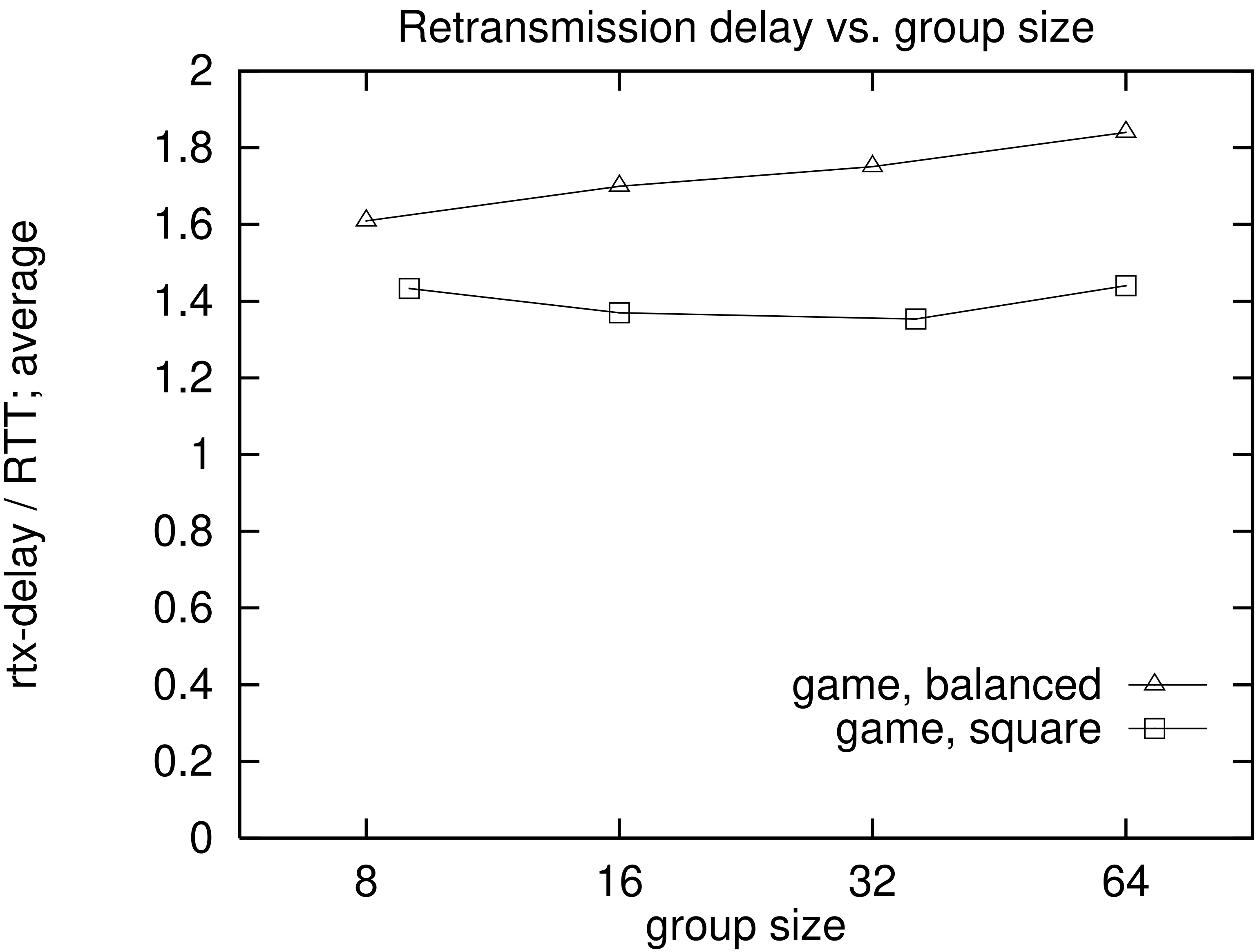
- links are all the same:
  10 ms propagation delay
  background Poisson traffic causes
    2.2% loss rate
    queue delay: 20 ms mean, 15 ms std.dev.

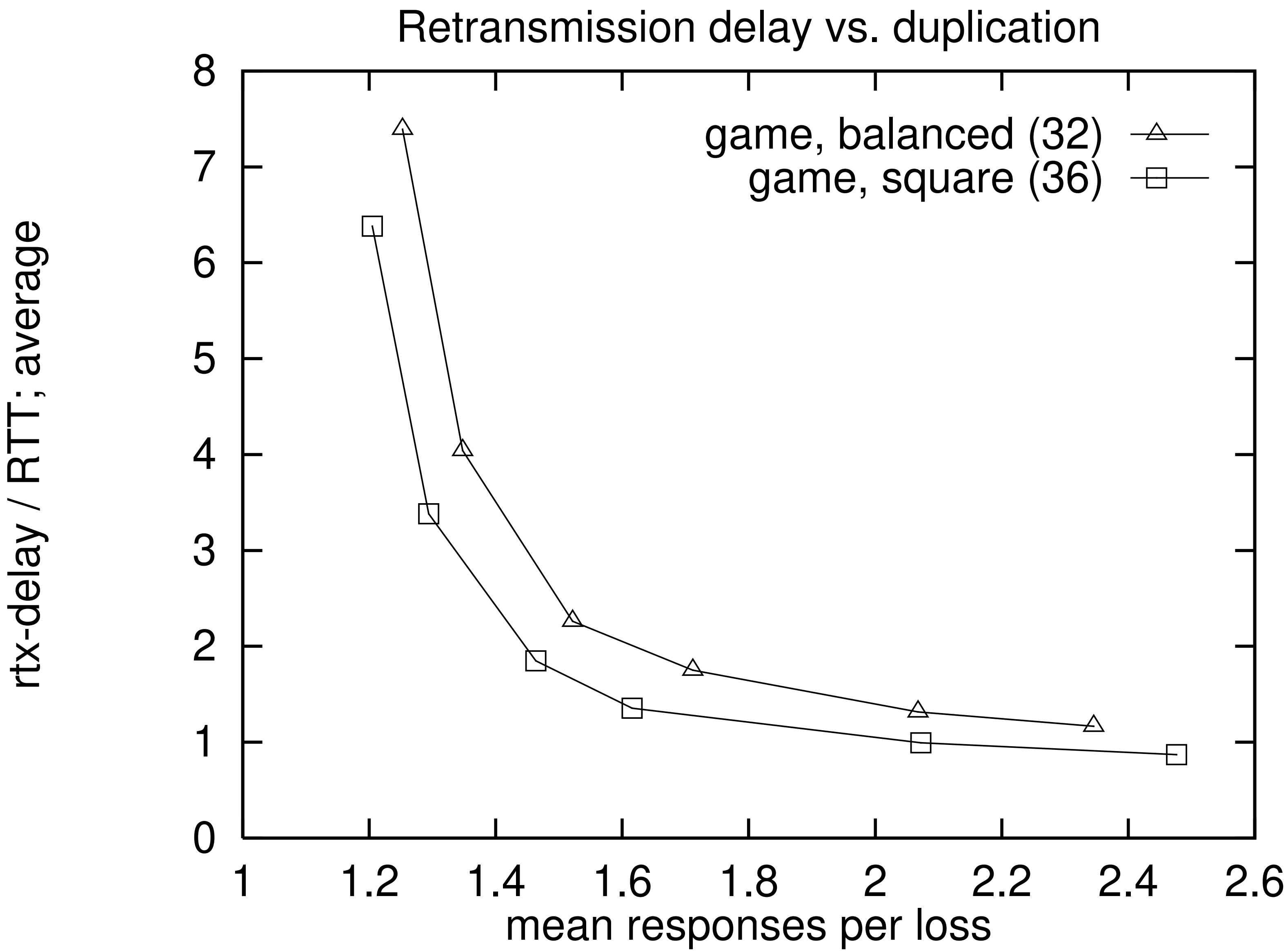
- topologies:



- applications (μ tuning goals):
  news: rtx delay $>$ 10 min w/prob 10^{-9}
  whiteboard: rtx delay $>$ 5 sec w/prob 10%
  game: repair ASAP but avg responses $<$ 4


Non-locality vs. group size
total responses / local responses
group size
0
0.5
1
1.5
2
8
16
32
64
news, balanced
whiteboard, balanced
game, balanced
news, square
whiteboard, square
game, square


Retransmission delay vs. group size
rtx-delay / RTT; average
group size
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
8
16
32
64
game, balanced
game, square


Retransmission delay vs. duplication
rtx-delay / RTT; average
mean responses per loss
game, balanced (32)
game, square (36)
0
1
2
3
4
5
6
7
8
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6

Conclusions

- Search Party is like LMS in structure, complexity, and performance, without using a hierarchy.
- Randomized forwarding can be useful.
- Essence of efficient end-to-end loss recovery is not hierarchy, but topology-dependent request forwarding (e.g. parentcast, randomcast).