Multicast Overview

(got through slide 40 on 6-Apr-06)

CSci551: Computer Networks
SP2006 Thursday Section

John Heidemann

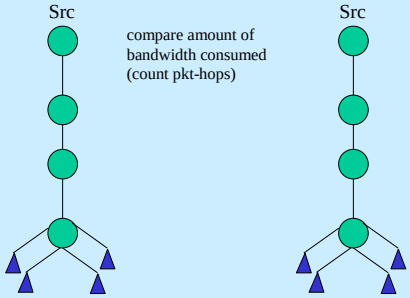
Why Study Multicast?

- more efficient when many people want the same thing
  - particularly with big content like TV and movies
  - distributed processing (big simulations, etc.)
  - on-line meetings and collaboration
- dealing with anonymous groups
  - announcement
  - DNS requests…want to find closest server
- ties in with prior discussions of routing

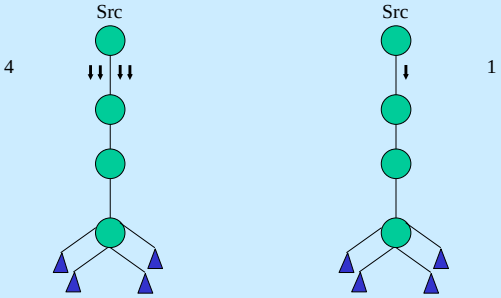
Multicast Goals

- Efficient data distribution
  - send only one copy of pkt over each link, not *n*
- Anonymous group addressing
  - ex. to get a phone number, you call the operator—*any* operator, not just Lilly

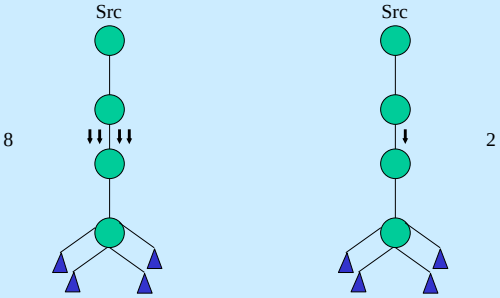
Unicast vs. Multicast



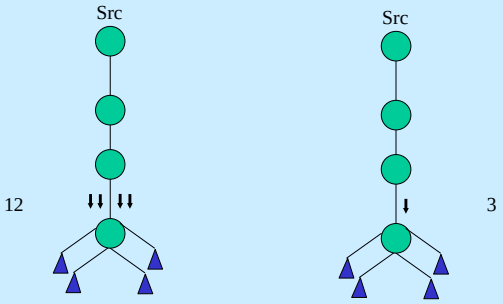
Unicast vs. Multicast



Unicast vs. Multicast

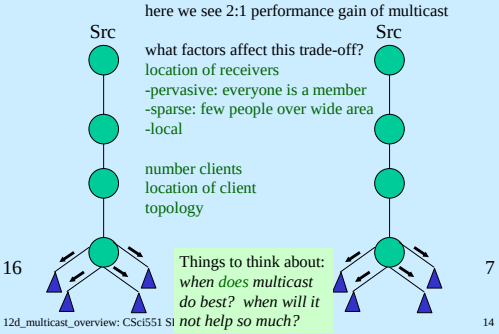


Unicast vs. Multicast



Unicast vs. Multicast

here we see 2:1 performance gain of multicast



what factors affect this trade-off?
location of receivers
-pervasive: everyone is a member
-sparse: few people over wide area
-local

number clients
location of client
topology

Things to think about:
when does multicast do best? when will it not help so much?

Applications of multicast

- videoconferencing
- collaborative workgroups
  - group editing
- file downloads
- IP TV
- distributed games
- distributed databases or computation

Multicast: Anonymous Addressing

- Applications:
  - idea: don't specify explicit destionation (just "the group", not a specific server)
  - ex: on-line advertisments, really large scale distrbutions don't want group membership, server location (ex. DNS), compute farms
  - reason: xxx
- Special case: *anycast*
  - find me the *nearest* receiver in the multicast group

Multicast: Bandwidth Reduction

- applications
  - IP TV, file distribution
- but some caveats
  - reliability? worried about ACK implosions
  - different users with different start times? what happens to what you already sent?
  - bandwidth glut? if you have lots of spare bandwidth, why bother with multicast?
  - other approaches? peer-to-peer

Multicast: Naturally Many-to-Many Apps

- some apps may be *inherently* many-to-many
  - examples: on-line gaming, collaborative apps like teleconfering or shared editing
- if so, is it easier to build them as
  - client/server
  - many-to-many (peer-to-peer?) using unicast
  - many-to-many using multi-cast

Common Problems in Multicast

- scalability
  - number of sources
  - number of receivers
  - geographic/network distance (*sparse vs. dense* groups)
  - network costs
    - xxx
- message *implosion*
- adapting to *many* recievers
- surviving component failure

Common Techniques in Mcast

- soft state
  - rather than reliable send and ACK, send periodically
  - example: keep alives in routing
  - benefit: constantly verify that state is accurate
- response after randomized delay
  - may biased delay to favor certain hosts
  - example: xxx
  - benefit: avoid collisions when multiple people could answer
- suppression of duplicate responses
  - listen to others responses: if they say the same as you, you don't need to respond
  - example: xxx
  - benefit: avoid wasting time with duplicate

Watch for these techniques in many places!