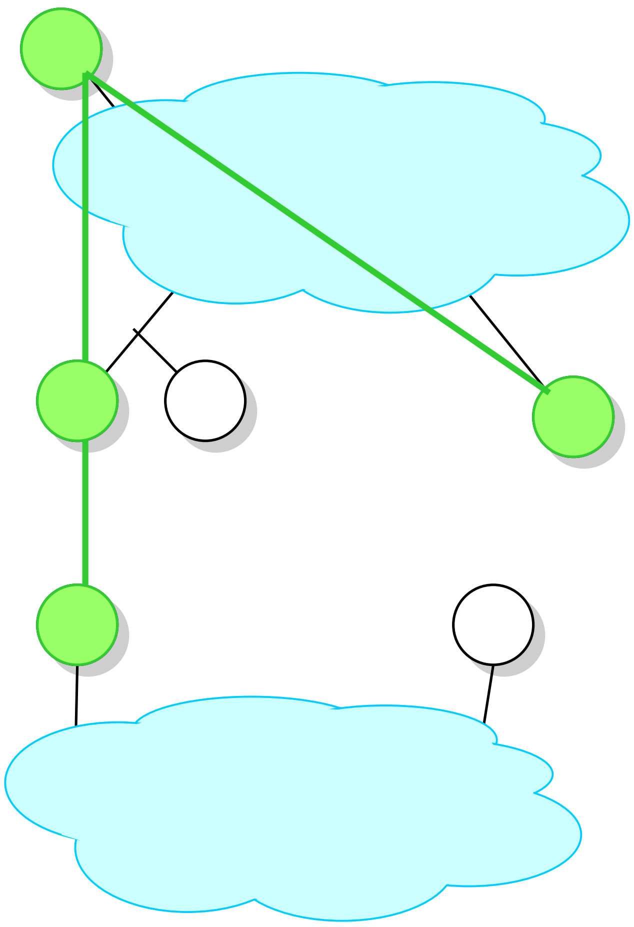
CS 268: Overlay Networks: Introduction and Multicast

Kevin Lai

April 29, 2001

Definition

- Network
  - defines addressing, routing, and service model for communication between hosts
- Overlay network
  - A network built on top of one or more existing networks
  - adds an additional layer of indirection/virtualization
  - changes properties in one or more areas of underlying network
- Alternative
  - change an existing network layer



A Historical Example

- Internet is an overlay network
  - goal: connect local area networks
  - built on local area networks (e.g., Ethernet), phone lines
  - add an Internet Protocol header to all packets

Benefits

- Do not have to deploy new equipment, or modify existing software/protocols
  - probably have to deploy new software on top of existing software
  - e.g., adding IP on top of Ethernet does not require modifying Ethernet protocol or driver
  - allows bootstrapping
    - expensive to develop entirely new networking hardware/software
    - all networks after the telephone have begun as overlay networks

Benefits

- Do not have to deploy at every node
  - not every node needs/wants overlay network service all the time
    - e.g., QoS guarantees for best-effort traffic
  - overlay network may be too heavyweight for some nodes
    - e.g., consumes too much memory, cycles, or bandwidth
  - overlay network may have unclear security properties
    - e.g., may be used for service denial attack
  - overlay network may not scale (not exactly a benefit)
    - e.g. may require n^2 state or communication

Costs

- Adds overhead
  - adds a layer in networking stack
    - additional packet headers, processing
  - sometimes, additional work is redundant
    - e.g., an IP packet contains both Ethernet (48 + 48 bits) and IP addresses (32 + 32 bits)
    - eliminate Ethernet addresses from Ethernet header and assume IP header(?)
- Adds complexity
  - layering does not eliminate complexity, it only manages it
  - more layers of functionality $\rightarrow$ more possible unintended interaction between layers
  - e.g., corruption drops on wireless interpreted as congestion drops by TCP

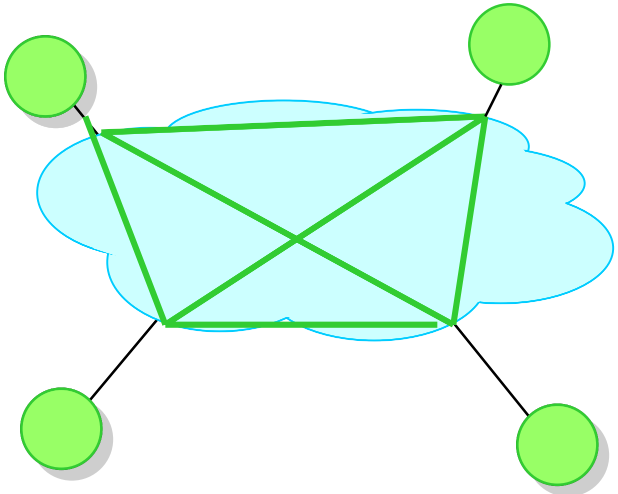
Applications

- Mobility
  - MIPv4: pretends mobile host is in home network
- Routing
- Quality of Service
- Addressing
- Security
- Multicast

Applications: Routing

- Flat space
  - every node has a route to every other node
  - n^2 state and communication, constant distance
- Hierarchy
  - every node routes through its parent
  - constant state and communication, log(n) distance
  - too much load on root
- Mesh (e.g., Content Addressable Network)
  - every node routes through 2d other nodes
  - O(d) state and communication, $n^{1/d}$ distance
- Chord
  - every node routes through O(log n) other nodes
  - O(log n) state and communication, O(log n) distance

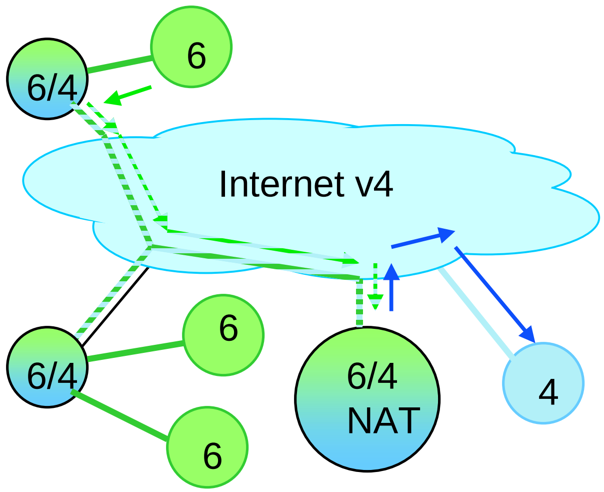
Applications: Quality of Service



- Resilient Overlay Networks [Anderson et al 2001]
  - overlay nodes form a complete graph
  - nodes probe other nodes for lowest latency
  - knowledge of complete graph $\rightarrow$ lower latency routing than IP, faster recovery from faults
  - ongoing work on providing stronger QoS models using FEC

Applications: Addressing

- provide more address space than underlying network
- 6bone
  - IPv6 on IPv4
  - requires NAT-like gateways for IPv6-only hosts to communicate with IPv4-only hosts
  - main current deployment of IPv6
- TRIAD, IP-NL
  - enhanced NAT
  - separate Internet into realms, each with its own IPv4 address space
  - use overlay network for inter-realm routing



Applications: Security (VPN)

- provide more security than underlying network
- privacy (e.g., IPSEC)
  - overlay encrypts traffic between nodes
  - only useful when end hosts cannot be secure
- anonymity (e.g., Zero Knowledge)
  - overlay prevents receiver from knowing which host is the sender, while still being able to reply
  - receiver cannot determine receiver exactly without compromising every overlay node along path
- service denial resistance (e.g., FreeNet)
  - overlay replicates content so that loss of a single node does not prevent content distribution

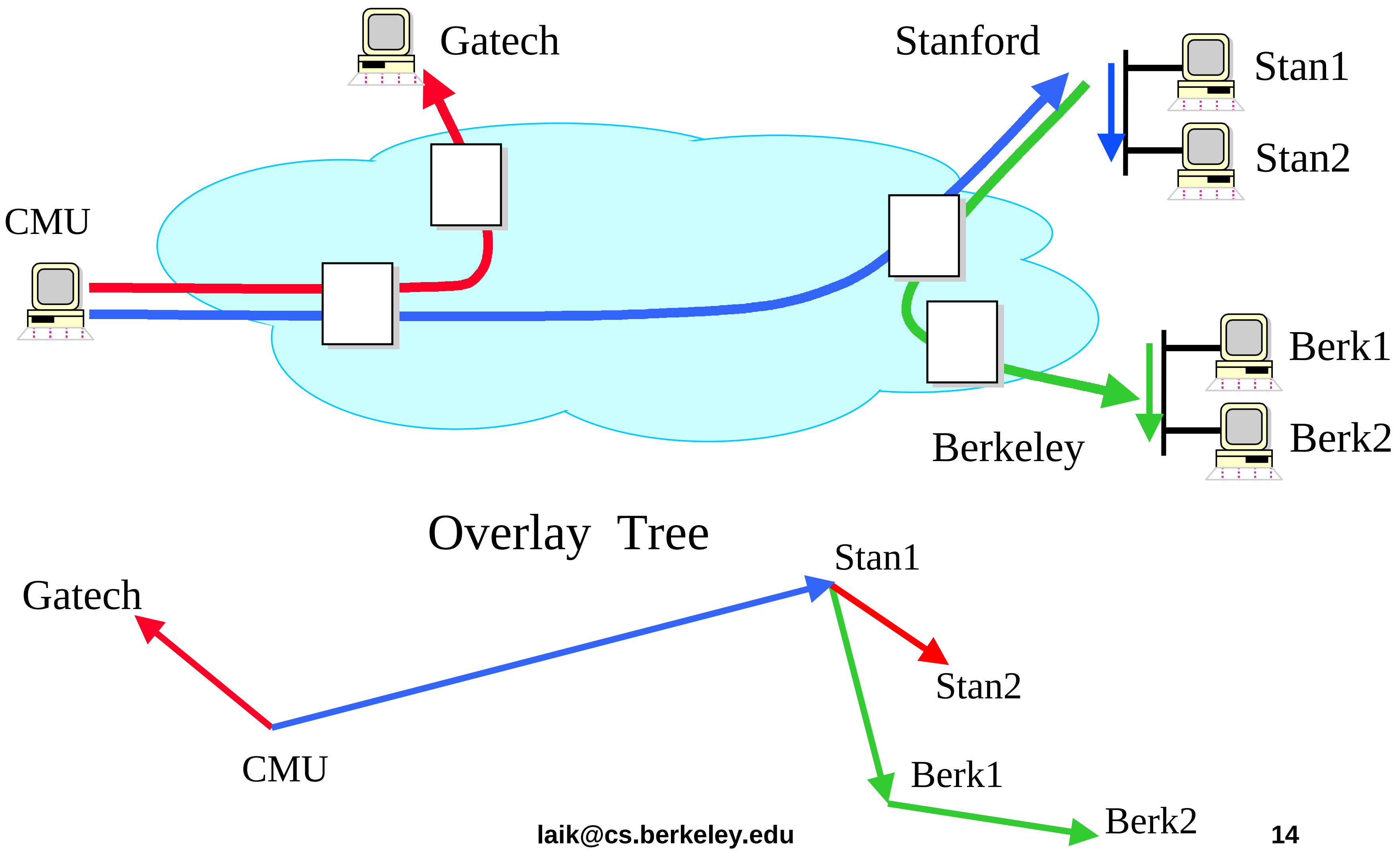
Problems with IP Multicast

- Scales poorly with number of groups
  - A router must maintain state for every group that traverses it
- Supporting higher level functionality is difficult
  - IP Multicast: best-effort multi-point delivery service
  - Reliability and congestion control for IP Multicast complicated
    - scalable, end-to-end approach for heterogeneous receivers is very difficult
    - hop-by-hop approach requires more state and processing in routers
- Deployment is difficult and slow
  - ISP's reluctant to turn on IP Multicast

Overlay Multicast

- Provide multicast functionality above the IP layer → overlay or application level multicast
- Challenge: do this efficiently
- Narada [Yang-hua et al, 2000]
  - Multi-source multicast
  - Involves only end hosts
  - Small group sizes <= hundreds of nodes
  - Typical application: chat

Narada: End System Multicast



Potential Benefits

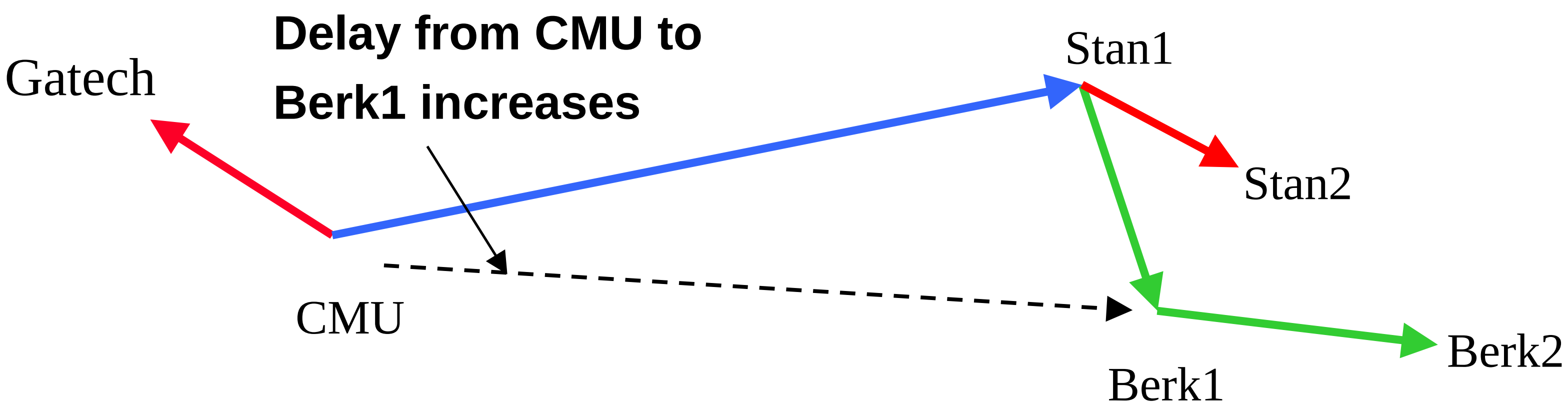
- Scalability
  - routers do not maintain per-group state
  - end systems do, but they participate in very few groups
- Easier to deploy
  - only requires adding software to end hosts
- Potentially simplifies support for higher level functionality
  - use hop-by-hop approach, but end hosts are routers
  - leverage computation and storage of end systems
  - e.g., packet buffering, transcoding of media streams, ACK aggregation
  - leverage solutions for unicast congestion control and reliability

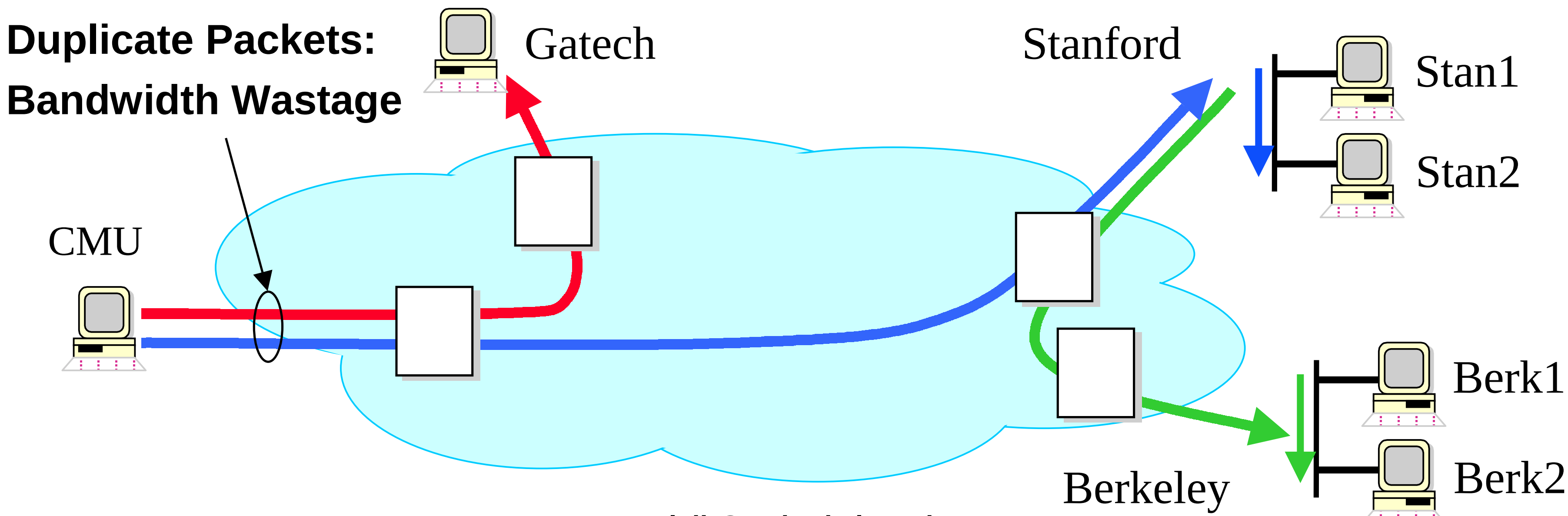
End System Multicast: Narada

- A distributed protocol for constructing efficient overlay trees among end systems

- Caveat: assume applications with small and sparse groups
  - Around tens to hundreds of members

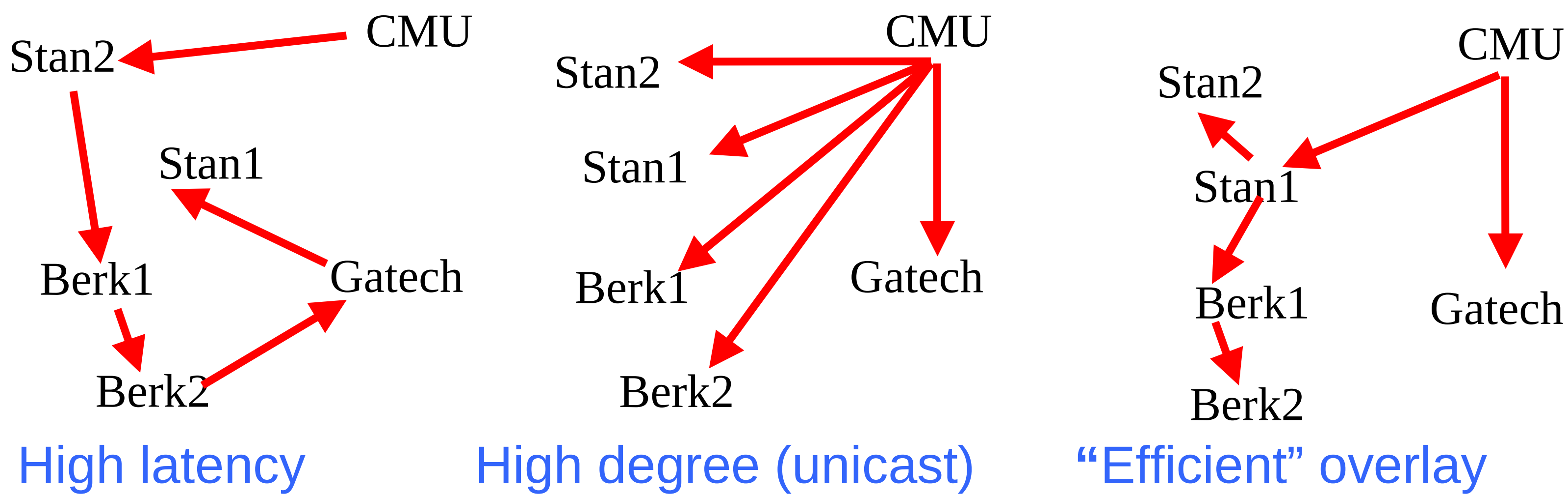
Performance Concerns





Overlay Tree

- The delay between the source and receivers is small
- Ideally,
  - The number of redundant packets on any physical link is low
- Heuristic:
  - Every member in the tree has a small degree
  - Degree chosen to reflect bandwidth of connection to Internet



Overlay Construction Problems

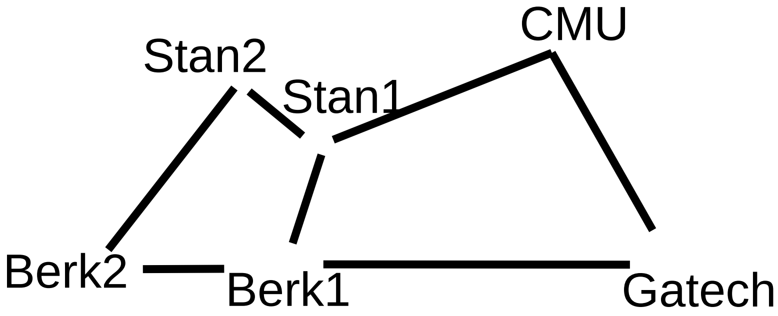
- Dynamic changes in group membership
  - Members may join and leave dynamically
  - Members may die
- Dynamic changes in network conditions and topology
  - Delay between members may vary over time due to congestion, routing changes
- Knowledge of network conditions is member specific
  - Each member must determine network conditions for itself

Solution

- Two step design
  - Build a mesh that includes all participating end-hosts
    - what they call a mesh is just a graph
    - members probe each other to learn network related information
    - overlay must self-improve as more information available
  - Build source routed distribution trees

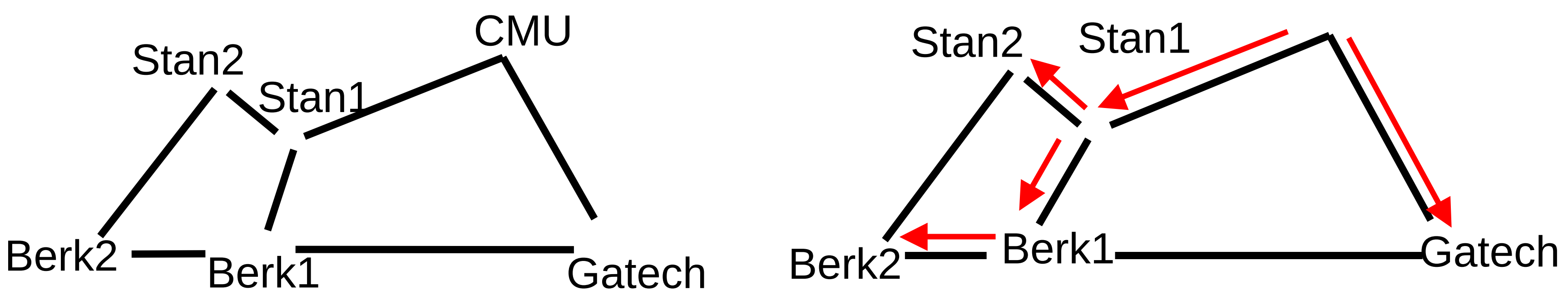
Mesh

- Advantages:
  - Offers a richer topology → robustness; don't need to worry to much about failures
  - Don't need to worry about cycles
- Desired properties
  - Members have low degrees
  - Shortest path delay between any pair of members along mesh is small



Overlay Trees

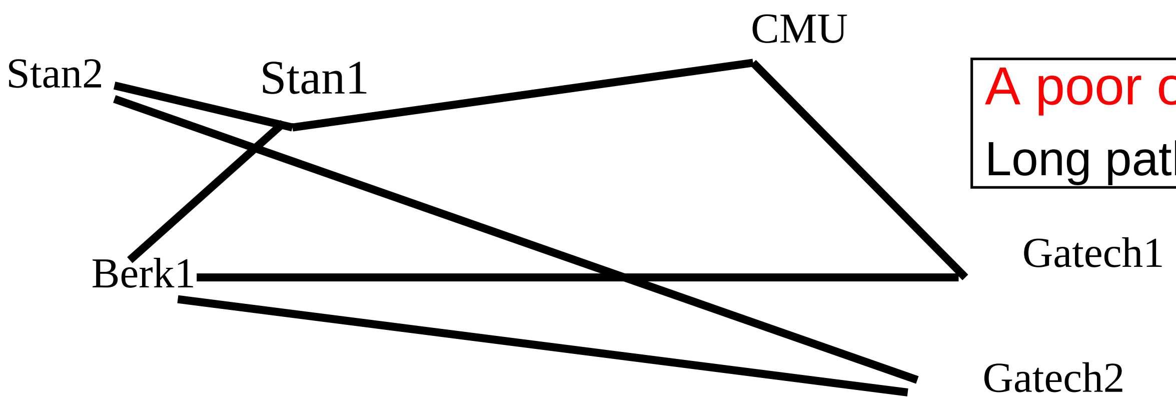
- Source routed minimum spanning tree on mesh
- Desired properties
  - Members have low degree
  - Small delays from source to receivers



Narada Components/Techniques

- Mesh Management:
  - Ensures mesh remains connected in face of membership changes
- Mesh Optimization:
  - Distributed heuristics for ensuring shortest path delay between members along the mesh is small
- Spanning tree construction:
  - Routing algorithms for constructing data-delivery trees
  - Distance vector routing, and reverse path forwarding

Optimizing Mesh Quality



A poor overlay topology:
Long path from Gatech2 to CMU

- Members periodically probe other members at random
- New link added if

  Utility_Gain of adding link > Add_Threshold

- Members periodically monitor existing links
- Existing link dropped if

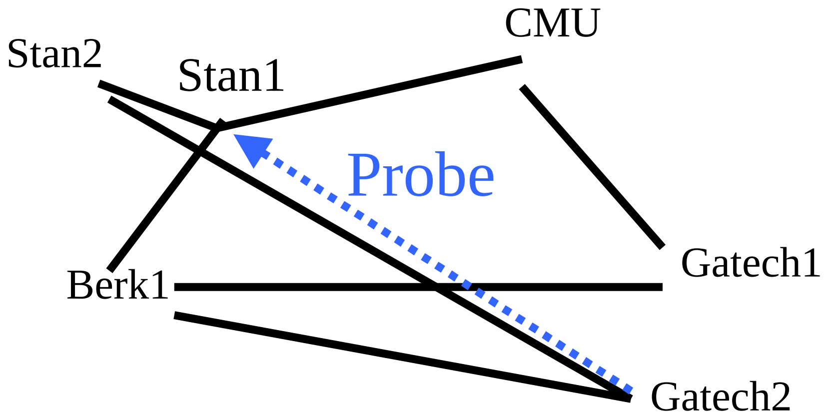
  Cost of dropping link < Drop Threshold

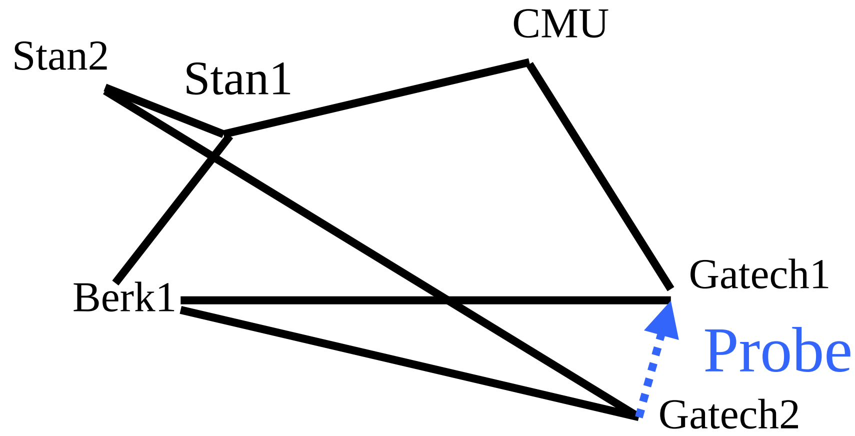
Definitions

- Utility gain of adding a link based on
  - The number of members to which routing delay improves
  - How significant the improvement in delay to each member is
- Cost of dropping a link based on
  - The number of members to which routing delay increases, for either neighbor
- Add/Drop Thresholds are functions of:
  - Member's estimation of group size
  - Current and maximum degree of member in the mesh

Desirable properties of heuristics

- Stability: A dropped link will not be immediately re-added
- Partition avoidance: A partition of the mesh is unlikely to be caused as a result of any single link being dropped





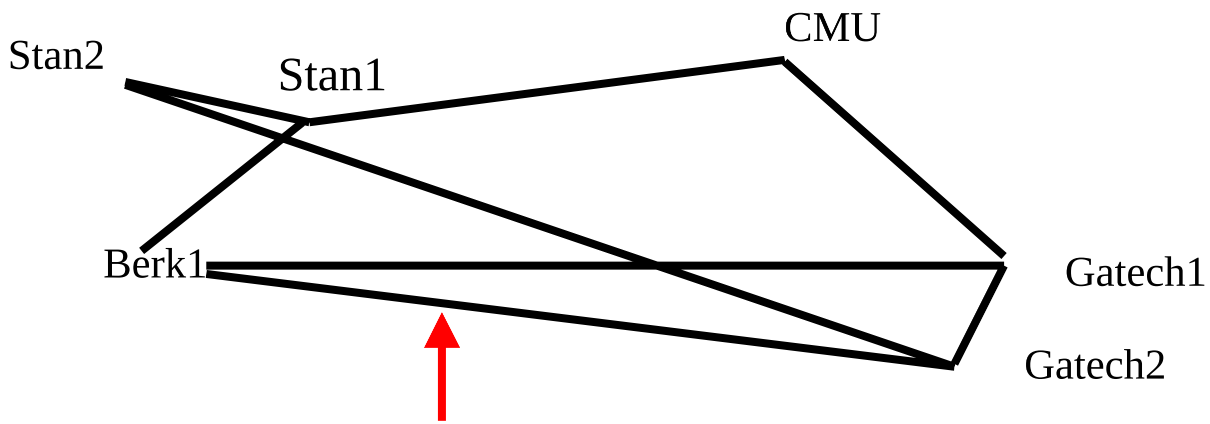
Delay improves to Stan1, CMU but marginally.

Do not add link!

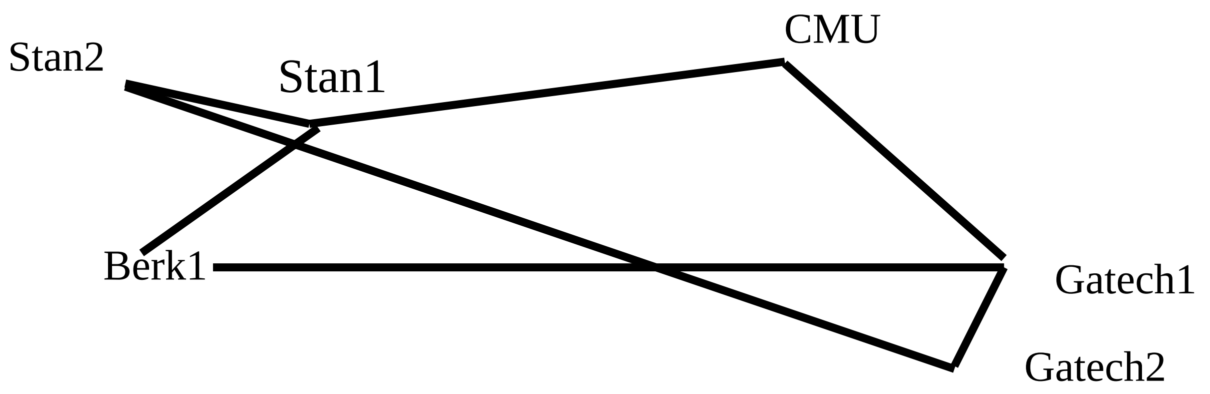
Delay improves to CMU, Gatech1 and significantly.

Add link!

Example



Used by Berk1 to reach only Gatech2 and vice versa: Drop!!



Simulation Results

- Simulations
  - Group of 128 members
  - Delay between 90% pairs < 4 times the unicast delay
  - No link caries more than 9 copies
- Experiments
  - Group of 13 members
  - Delay between 90% pairs < 1.5 times the unicast delay

Summary

- End-system multicast (NARADA) : aimed to small-sized groups
  - Application example: chat
- Multi source multicast model
- No need for infrastructure
- Properties
  - low performance penalty compared to IP Multicast
  - potential to simplify support for higher layer functionality
  - allows for application-specific customizations

Other Projects

- Overcast [Jannotti et al, 2000]
  - Single source tree
  - Uses an infrastructure; end hosts are not part of multicast tree
  - Large groups ~ millions of nodes
  - Typical application: content distribution
- Scattercast (Chawathe et al, UC Berkeley)
  - Emphasis on infrastructural support and proxy-based multicast
  - Uses a mesh like Narada, but differences in protocol details
- Yoid (Paul Francis, FastForward/ACIRI)
  - Uses a shared tree among participating members
  - Distributed heuristics for managing and optimizing tree constructions

Conclusion

- Narada demonstrates the flexibility of the application level multicast
  - I.e., the ability to optimize the multicast distribution to the application needs
- Issues
  - 4x unicast delay could be a problem for interactive applications
  - reliability and congestion control for heterogeneous receivers not demonstrated
  - sender access control solution not demonstrated
  - overhead of probes is low for one group, what about for n groups on same host?
  - is stress really an important metric?

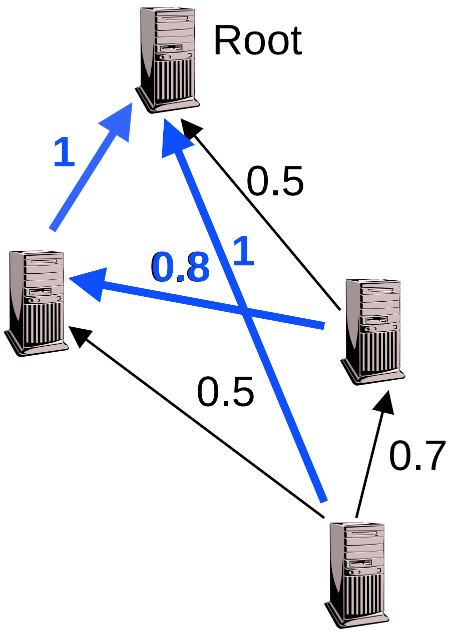
Overcast

- Designed for throughput intensive content delivery
  - Streaming, file distribution
- Single source multicast; like Express
- Solution: build a server based infrastructure
- Tree building objective: high throughput

Tree Building Protocol

- Idea: Add a new node as far away from the route as possible without compromising the throughput!

```
Join (new, root) {
    current = root;
    B = bandwidth(root, new);
    do {
       B1 = 0;
       forall n in children(current) {
          B1 = bandwidth(n, new);
          if (B1 >= B) {
            current = n;
            break;
         }
    } while (B1 >= B);
    new->parent = root;
}
```

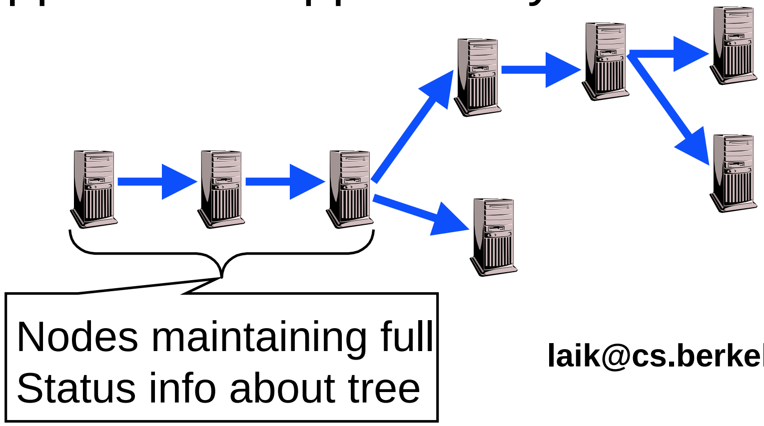


Details

- A node periodically reevaluates its position by measuring bandwidth to its
  - Siblings
  - Parent
  - Grandparent
- The Up/Down protocol: track membership
  - Each node maintains info about all nodes in it sub-tree plus a log of changes
    - Memory cheap
  - Each node sends periodical alive messages to its parent
  - A node propagates info up-stream, when
    - Hears first time from a children
    - If it doesn’t hear from a children for a present interval
    - Receives updates from children

Details

- Problem: root → single point of failure
- Solution: replicate root to have a backup source
- Problem: only root maintain complete info about the tree; need also protocol to replicate this info
- Elegant solution: maintain a tree in which first levels have degree one
  - Advantage: all nodes at these levels maintain full info about the tree
  - Disadvantage: may increase delay, but this is not important for application supported by Overcast



Some Results

- Network load < twice the load of IP multicast (600 node network)
- Convergence: a 600 node network converges in ~ 45 rounds

Summary

- Overcast: aimed to large groups and high throughput applications
  - Examples: video streaming, software download
- Single source multicast model
- Deployed as an infrastructure
- Properties
  - Low performance penalty compared to IP multicast
  - Robust & customizable (e.g., use local disks for aggressive caching)