Overlay Multicast Networks

A/P YEO Chai Kiat
NTU
Singapore

OSAKA UNIVERSITY

PRIUS

10 – 11 Jan 2007

NANYANG TECHNOLOGICAL UNIVERSITY

Background

- Media Streaming is currently one of the reigning applications over Internet eg. Youtube videos
- AccuStream iMedia Research reported a total of 18 billion video streams served in 2005.
- It projected the total annual video streams will reach 28 billion in 2006.

Multipoint Communications accomplished via:

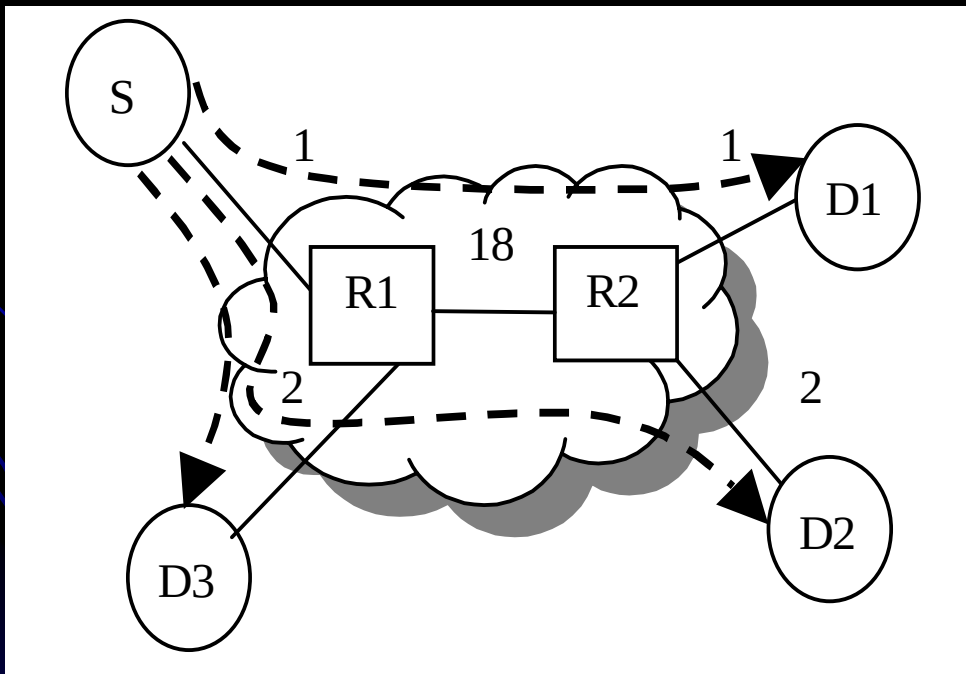
- Unicast
- Multicast
- Application Level Multicast Overlay – Motivation?

Scope

1. Background
2. Application Level Multicast (ALM)
3. Basic Design Considerations of ALM
4. Performance Metrics for Evaluation
5. General Classification of ALM
6. Case Studies
7. Conclusion
8. Future Directions

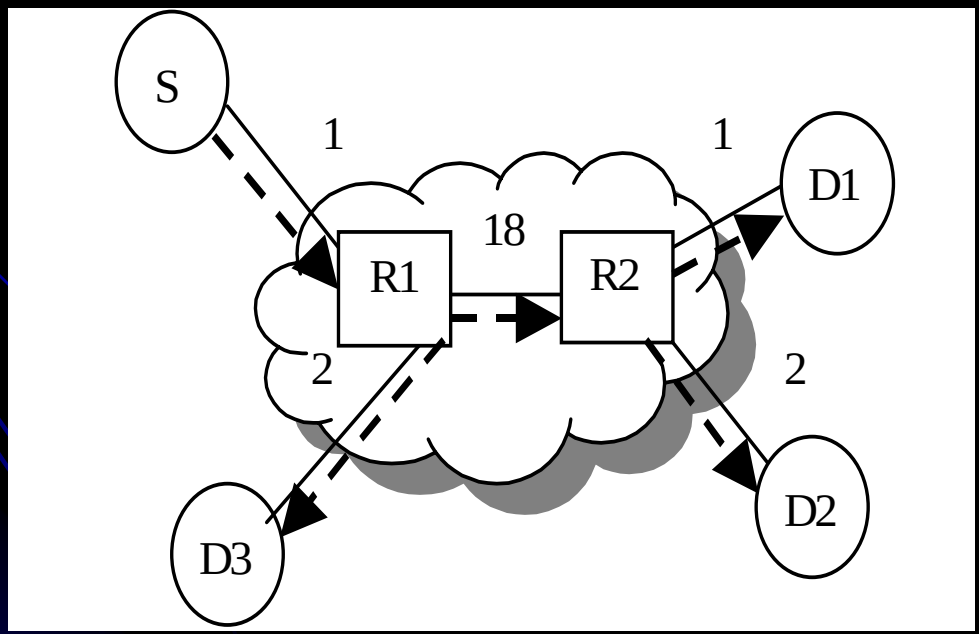
Unicast

- Conventional point-to-point unicast protocol from the source to the multiple destinations.
- Inefficient and non-scaleable. Duplication of data by source to each and every client in the group.
- Source becomes a hot-spot leading to network congestion and server overload.



IP Multicast

- Regarded as ‘ideal’ for multipoint communication as it is inherently bandwidth efficient and scaleable.
- Reserves a set of addresses to identify groups of receivers.
- Source only needs to send a single copy of its datagram. Data forwarding is effected along a distribution tree which spans all members of a group.



Problems with IP Multicast

- Routers need to maintain separate routing state for each multicast group resulting in high complexity and serious scaling constraints at the IP layer.
- Lack of access control as any arbitrary source can send data to any arbitrary group.
- Groups entail complicated membership control as well as network management and provisioning.
- Difficult to ensure scaleability and consistency with distributed multicast address allocation as every group needs to dynamically obtain a globally unique address from the limited multicast address space.

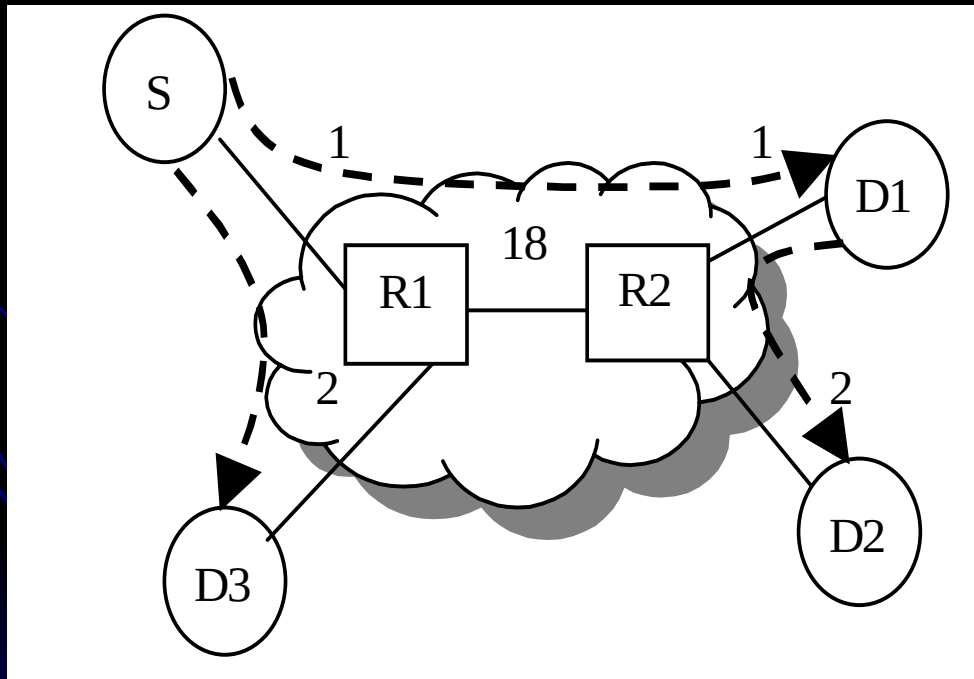
Problems with IP Multicast (cont.)

- Provides best effort service. More challenging than unicast to provide higher level features such as reliability, flow and congestion control, security.
- Entails substantial modifications at the infrastructural level given the fairly elaborate control support from the routers, in particular membership management and multicast routing protocols.
- Absence of a widely accepted, efficient, scaleable and deployable protocol for inter-domain multicast routing (eg. PIM, BGMP, MBGP, MSDP).
- Hence, not widely deployed by ISPs.

Application Level Multicast (ALM)

Multicast functionality is moved from routers to end users, i.e. from network layer to application layer.

End users (peers) are connected via a virtual network (overlay) with each edge connecting a pair of end users corresponding to a unicast path between them in the physical network.



Motivation for ALM

- Easy deployability. Circumvent deployment barriers to network level resources.
- Flexibility, adaptivity, simplified configuration, enhanced functionalities and services, achieved without requiring changes in the underlying network.
- Self-organising overlay networks form the bedrock of current peer to peer networks.

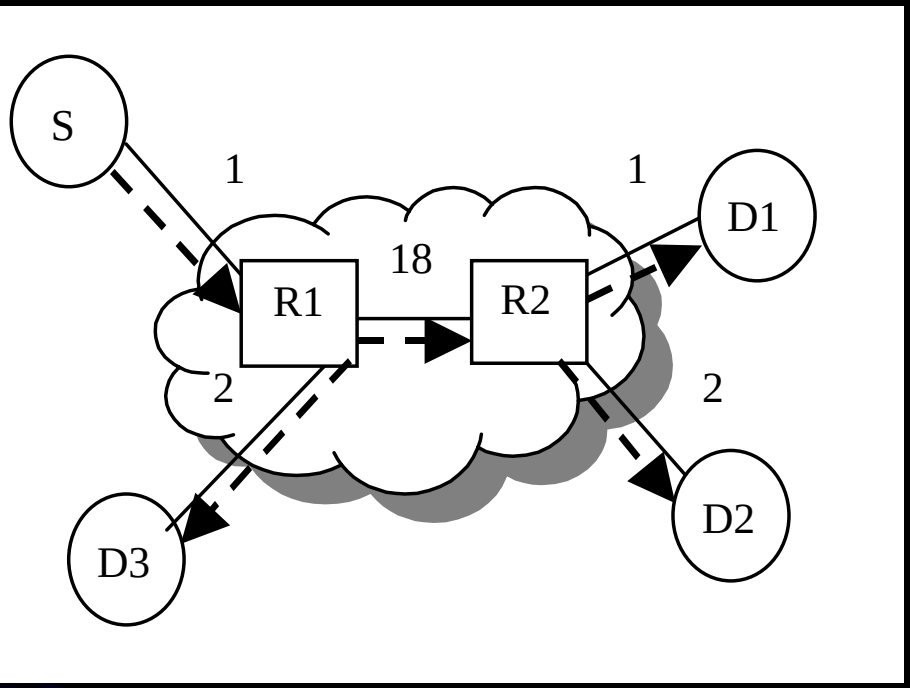
Motivation for ALM (cont.)

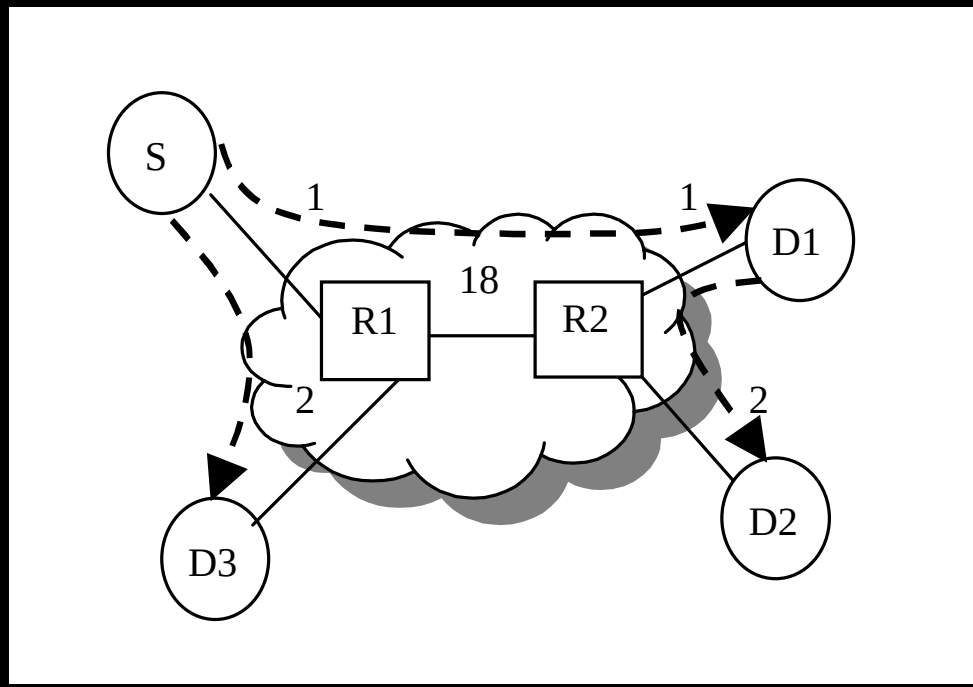
- Data is routed via this virtual overlay network to reach all members without any support from the routers beyond regular unicast transport service.
- No need for a global group identifier such as an IP multicast address.
- Can leverage on services such as flow control, congestion control, reliable delivery, available to unicast.
- Can exploit end host resources such as processing power, caching, buffering for enhanced services

Weaknesses of ALM

- Less bandwidth efficient as IP multicast but more efficient than unicast.
- Data path between end users tends to incur higher latencies compared to IP multicast as overlay paths are incongruent to the underlying physical network.
- Less robust (based on peers rather than dedicated routers)
- Additional overheads to track network characteristics and have peers self-organize into overlays
- No standard protocol

Eg. of Performance Penalty





IP Multicast

- Delay from S to D2 = 21
- Data traversed underlying network links only once

ALM

- Delay from S to D2 = 23
- Data traversed link D1-R2 twice

Basic Design Considerations of ALM

- Target application of ALM
- Service discovery (name, address)
- Membership management
- Overlay construction for data delivery to optimize performance
- Application level routing algorithms
- Peers status tracking
- Network conditions monitoring
- Resilience, recovery from node failures and departures

Performance Metrics for Evaluation

- *Stretch* or *Relative Delay Penalty (RDP): R*atio of overlay delay over physical unicast delay

  - Measures end to end latency as seen by application
  - Metric for quality of the overlay data paths

- *Tree Cost Ratio: R*elative cost of an ALM overlay tree to that of source-rooted shortest path multicast tree

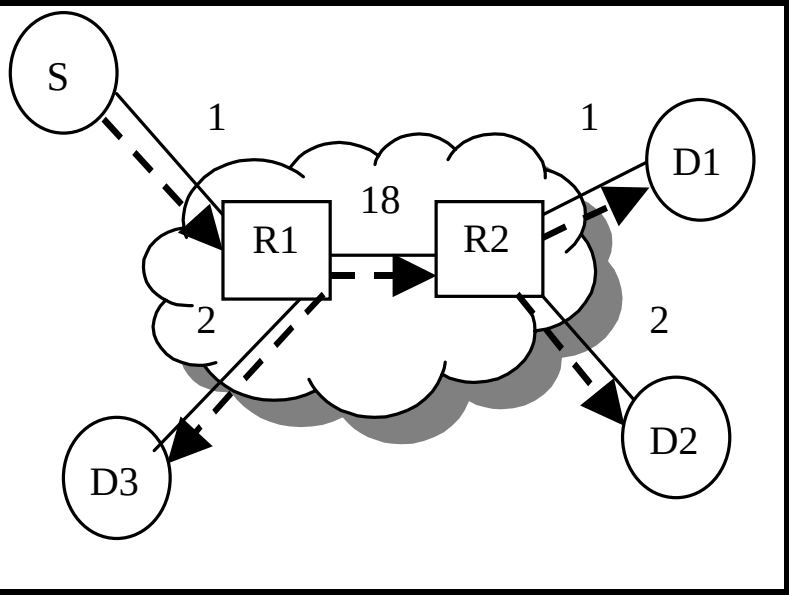
  - Tree cost is the sum of all delays on the tree links
  - Indicates the quality of the overlay tree compared to router-supported IP multicast

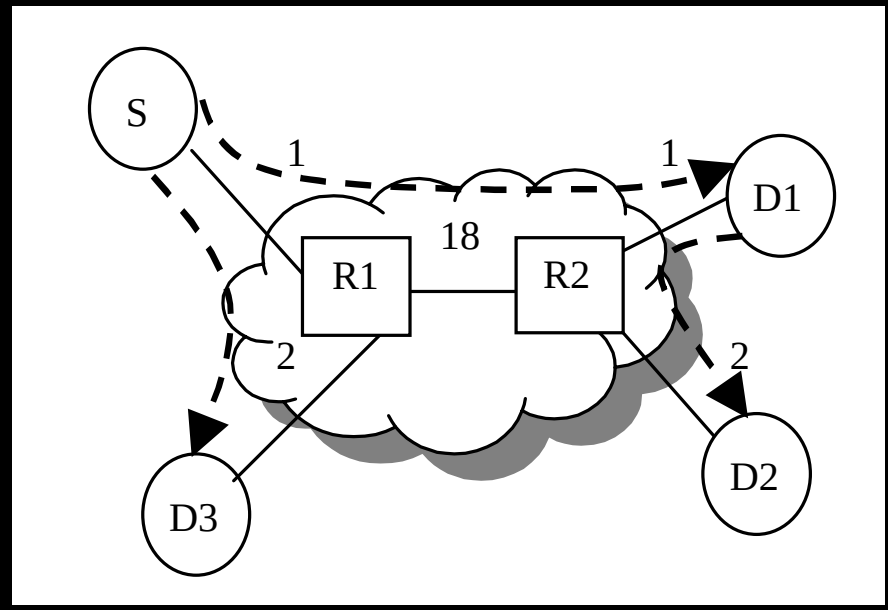
- *Bandwidth:* Measures the application level throughput at the client

Performance Metrics (cont.)

- *Stress:* The number of identical copies of a data packet carried by a physical link
  - Measures the additional network resources consumed by ALM compared to IP multicast
  - Metric to quantify the quality of the overlay data path
- *Resource Usage*: $\sum_{i=1}^{L} d_i * s_i$
  - *L:* number of links active in data transmission
  - *di:* delay of link *i* , *si:* stress of link *I*
  - A metric of the network resources consumed in the process of data delivery to all receivers
- *Normalized Resource Usage: R*atio of the resource usage of the ALM relative to that by IP multicast.

Examples of Calculation





IP Multicast

Tree Cost = 1+18+1+2+2

= 24

(S to R1 + R1 to R2 +
R1 to D3 + R2 to D1 +
R2 to D2)

ALM

Stretch for D2 = 1+18+1+1+2=23

RDP for D2 = 23 / (1+18+2)

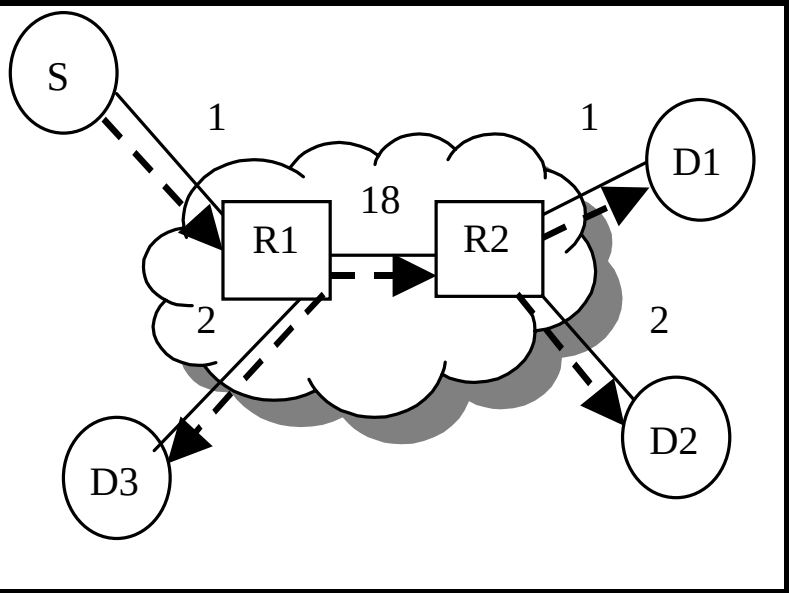
= 23 / 21 = 1.1 x Unicast

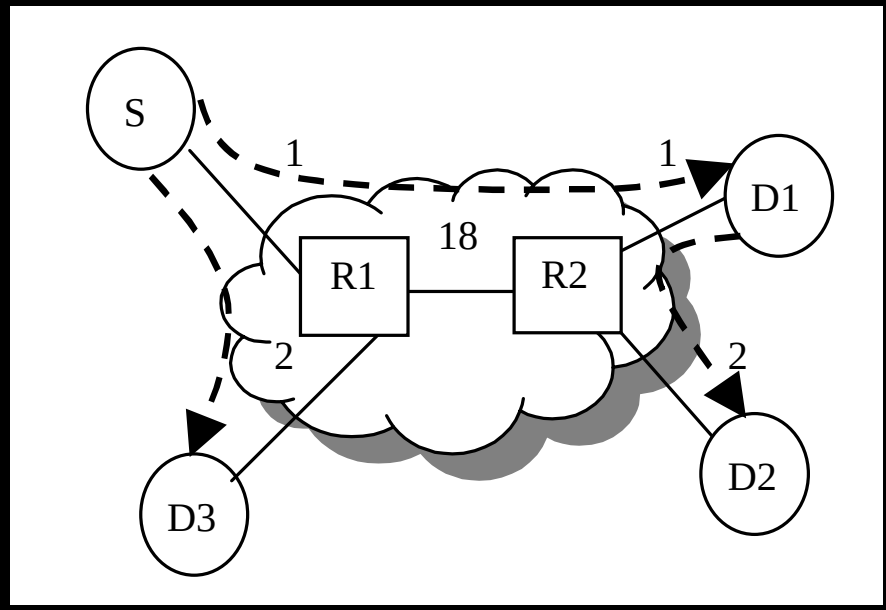
Tree Cost = 20+3+3 = 26

(S to D1 + S to D3 + S to D2)

TCR = 26 / 24 = 1.08 x IP Multicast

Examples of Calculation (cont.)





IP Multicast

Stress on all links = 1

Resource Usage = 5 links x 1

= 5

ALM

Stress R1 to R2, R2 to D2,

R1 to D3 = 1

Stress S to R1, R2 to D1 = 2

Resource Usage = 3 links x 1 +

2 links x 2 = 7

NRU = 7 / 5 = 1.4 x IP multicast

Performance Metrics (cont.)

- *Control Traffic / Protocol Overhead:* Ratio of the total control bandwidth incurred by the ALM over the total bandwidth consumed by the data.

- *Speed of Repair & Recovery*: Determines how fast the overlay recover from a node departure or failure

General Classification of ALM

- Centralized, distributed or hybrid overlay building algorithms
- Infrastructure (proxy) based vs Peer-to-peer (P2P) based

Overlay Building Algorithm - Centralized

Presence of a central controller with full responsibilities for service discovery, group management, overlay computation, optimization and repair.

Controller:

- Maintains group information
- Handles group membership
- Collects measurements from all members
- Computes optimal overlay tree
- Disseminates the routing tables to all members

Overlay Building Algorithm – Centralized (cont.)

Pros:

- Simplification of overlay construction
- More efficient and simpler group management
- More reliable mechanism to prevent tree partitions and routing loops

Cons:

- Scaleability Limitation
- Controller constitutes a single point of failure

Example: ALMI (Application Level Multicast Infrastructure) [1]

Overlay Building Algorithm - Distributed

Receiver-based and it distributes the responsibilities of group membership and overlay computation to the individual peers.

Overlay is self-organising and self-improving.

Pros:

- Relatively more robust, no single point of failure as failure of an individual node will not impact the whole group

- More scaleable to large groups

Overlay Building Algorithm – Distributed (cont.)

Cons:

- Less efficient in building optimal overlay
- Overheads incurred in discovering peers
- Needs algorithms to handle recovery from node departure and failure
- Limited knowledge of network conditions
- Overlay must self-improve as more information becomes available

Examples: Narada (End System Multicast) [2],
HM (Host Multicast) [3]

Overlay Building Algorithm - Hybrid

A lightweight controller is still used to facilitate some of the group management (e.g. join process), overlay construction and failure recovery while the actual overlay construction and failure recovery are left to a distributed algorithm.

Pros:

- More efficient than distributed systems in building more optimal overlays
- Faster join process than distributed approach
- Better than distributed systems in recovery from node failures as it not only uses distributed mechanisms but can also fall back on the lightweight controller for assistance

Overlay Building Algorithm – Hybrid (cont.)

Cons:

- Lightweight controller is also a single point of failure
- Overheads incurred in discovering peers

A tradeoff between centralized and distributed systems

Examples: HPAM (Hybrid Protocol for Application Level Multicast) [4]

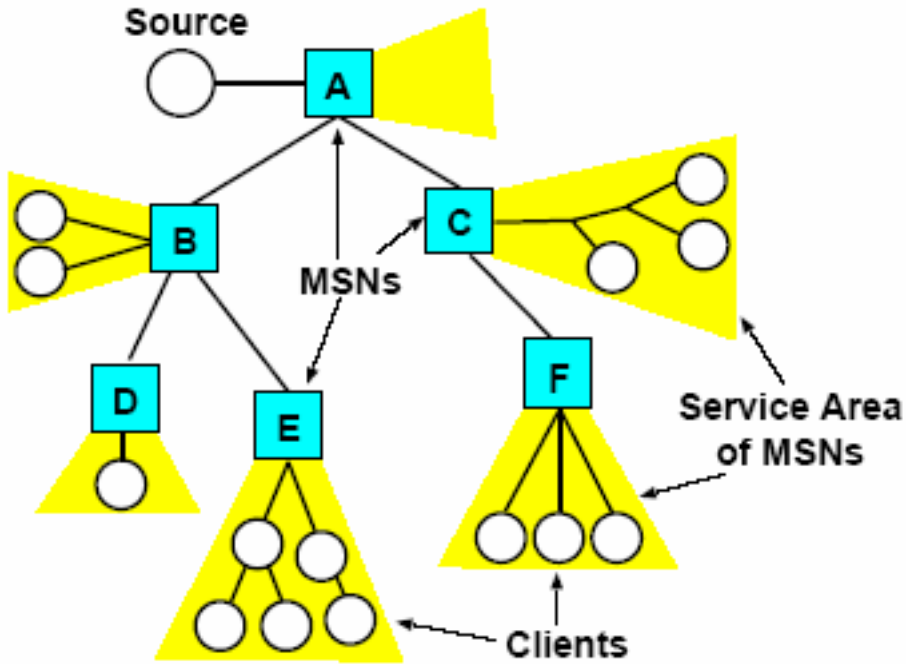
Proxy based ALM

Makes explicit use of infrastructure service agents or proxies or dedicated servers strategically deployed within the network to provide efficient data distribution and value-added services to a set of end hosts.

The overlay is built across these proxies and end hosts attach themselves to proxies closest to them. The proxies act on behalf of the end hosts.

Example: OMNI (Overlay Multicast Network Infrastructure) [5].

Proxies are called MSN (Multicast Service Nodes).



Proxy based ALM (cont.)

Pros:

- Functionally dedicated, homogeneous and better provisioned than individual peers
- Hence more reliable and robust to failure
- Persistent beyond the lifetime of individual peers
- Can provide value-added services such as being application-aware
- Can be positioned at strategic positions such as co-locating with IP routers or at hotspots to provide more efficient services

Proxy based ALM (cont.)

Cons:

- May encounter acceptance and deployment problems if they were to be placed at strategic locations

- Less responsive to changing environment conditions as proxy placement is usually static and manually deployed

Example: OMNI (Overlay Multicast Network Infrastructure) [5]

Peer to Peer (P2P) based ALM

P2P approach constructs the overlay across end-users with all functionalities being vested with these end-users.

Pros:

- Is simple to set up and deploy as peers can be anywhere

- Has vast combined resources such as physical connectivity and computing resources. Although resources may not be uniform, they can be individually harnessed according to their capacities.

- Provides redundancy as a single failure will not radically affect the big group.

- Peers can be dynamically deployed in large numbers and at hot spots quickly with minimum prior configuration

P2P based ALM (cont.)

Cons:

- Is less robust as peers are not dedicated machines and can fail or leave any time
- Has shorter and less persistent lifetime than proxies
- Is difficult to provide added functionalities as peers' capabilities are beyond control by the application

Examples: ALMI, Narada, HM, HPAM

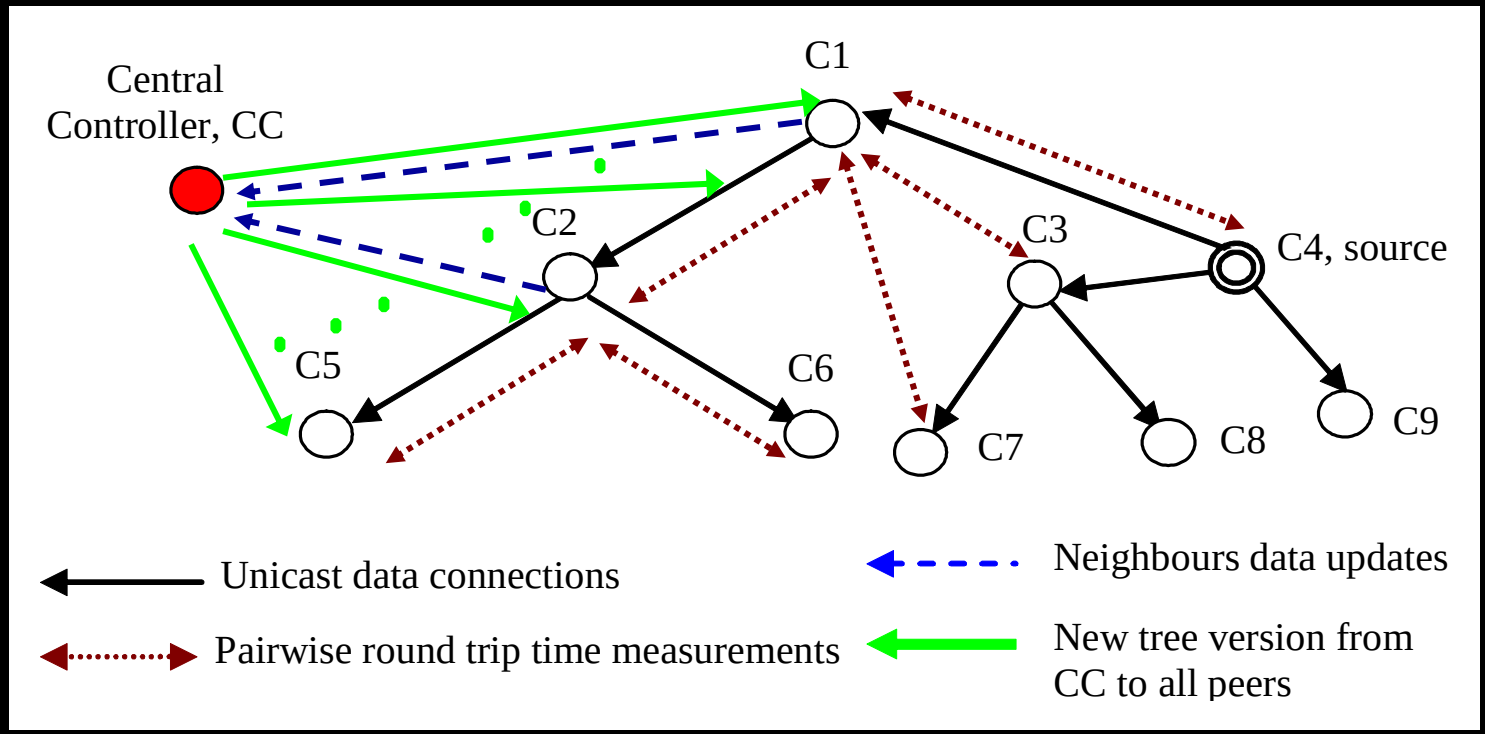
Case Studies

- ALMI (centralized)
- Narada (distributed and build a mesh before building the overlay tree)
- HM (distributed, only build the overlay tree)
- HPAM (hybrid)
- OMNI (hybrid, distributed, proxy-based)

Centralized: ALMI

- ALMI supports groups of relatively small size (several 10s)
- Builds a single shared tree of peers
- Central controller creates minimum spanning tree (MST) which spans up all the peers. Thus, loops are taken care of.
- Peers are connected on the overlay via unicast connections
- Uses latency between members as the link cost to build MST
- Optimizes overlay for low latency data delivery

ALMI (cont.)



- New peer contacts CC controller to join the session
- CC returns random list of some peers for new node to join
- New peer will join the parent the closest to it after taking round trip time (RTT) measurements
- When a peer fails, the orphaned child will contact CC to rejoin the session (recovery from failure via rejoining)

ALMI (cont.)

- CC instructs all peers to collect pair-wise latency data from all the rest
- Hence protocol overhead is of order $O(N^2)$ where N is the number of peers in the ALMI minimum spanning tree
- Peers update CC periodically with their collected data
- CC periodically re-computes the MST
- Routing information of new MST is sent to all members with a new monotonically increasing version number

ALMI (cont.)

- Peers have to cache different versions of the routing tables until it receives packets with a new version number not in its cache. It then registers with the CC to receive the new MST and discard the old versions.

- ALMI multicast trees have been shown to be close to source-rooted multicast trees in efficiency [1]

- High overheads due to full knowledge of member data and maintaining the different tree versions.

- ALMI proposes the reduction of protocol overhead by asking each peer to collect pair-wise latency data for only 10% of total number of peers instead of all. The incomplete group knowledge, however, reduces the optimality of the ALMI's MST.

Distributed: Narada (End System Multicast)

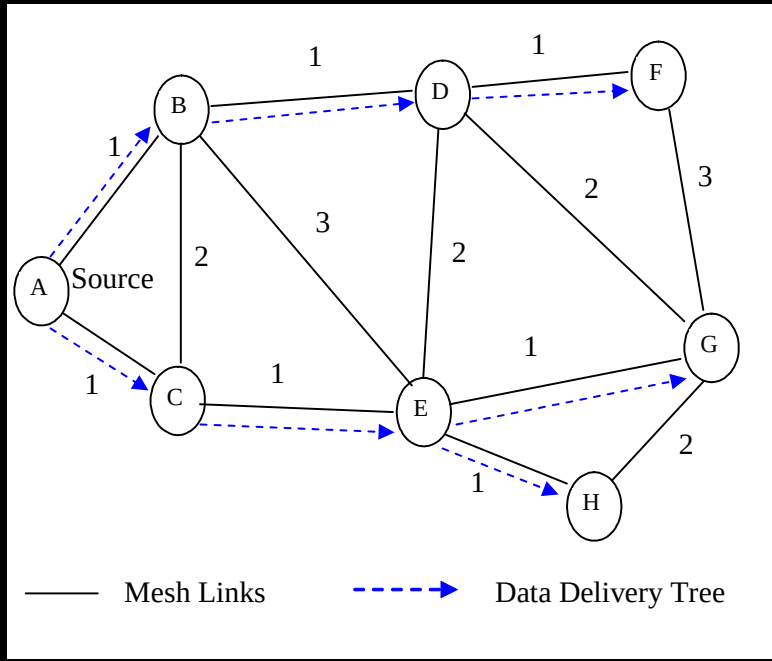
- Targeted to support collaborative applications with small group size where any peer can be a source (eg. conferencing).
- It builds two overlays.
- First it builds a mesh overlay across all peers.
- Next, it builds N source-rooted, shortest delay spanning trees on top of the mesh (N is the total number of peers).
- The trees are rooted at each peer and is hence optimal for each peer unlike a single shared tree for all.
- All overlay related operations are performed on the mesh overlay except for data distribution which takes place on the tree overlay.

Narada (cont.)

Join:

- New member joins the mesh by obtaining an initial few peers from some methods such as a well-known rendezvous point (RP).

- New member randomly selects some of these members as its mesh neighbours and establish a link to them.



Leave:

Leaving member informs its neighbours on the mesh and the routing protocols will find an alternate path for the overlay tree.

Failure:

A failed peer will cause a partition on the mesh. New link will be added to bridge the partition.

Narada (cont.)

Peer Tracking:

- Each peer sends periodic refresh messages with sequence numbers across the mesh overlay.
- In this way, each peer can maintain a list of all other members in the group.

Routing Tables:

- Each peer maintains a routing path and the routing cost to every other peer.
- Routing update messages exchanged between neighbours

Very high control overhead to maintain group membership and routing tables in the order of $O(N^2)$.

A lot of data has to be maintained at each peer for all other peers.

Narada (cont.)

Self-improving, Self-Recovering Mesh Overlay

Add useful link by periodically evaluating the utility of the different mesh links :

- *Peer m* selects a random *Peer n* which is not a neighbour
- With the help of *Peer n*'s routing table, it will add a link to *Peer n* if the link improves the routing delay of *Peer m* to all members beyond a set threshold
- The utility of a link is hence defined as the improvement in routing delay from *Peer m* to the rest of the group.

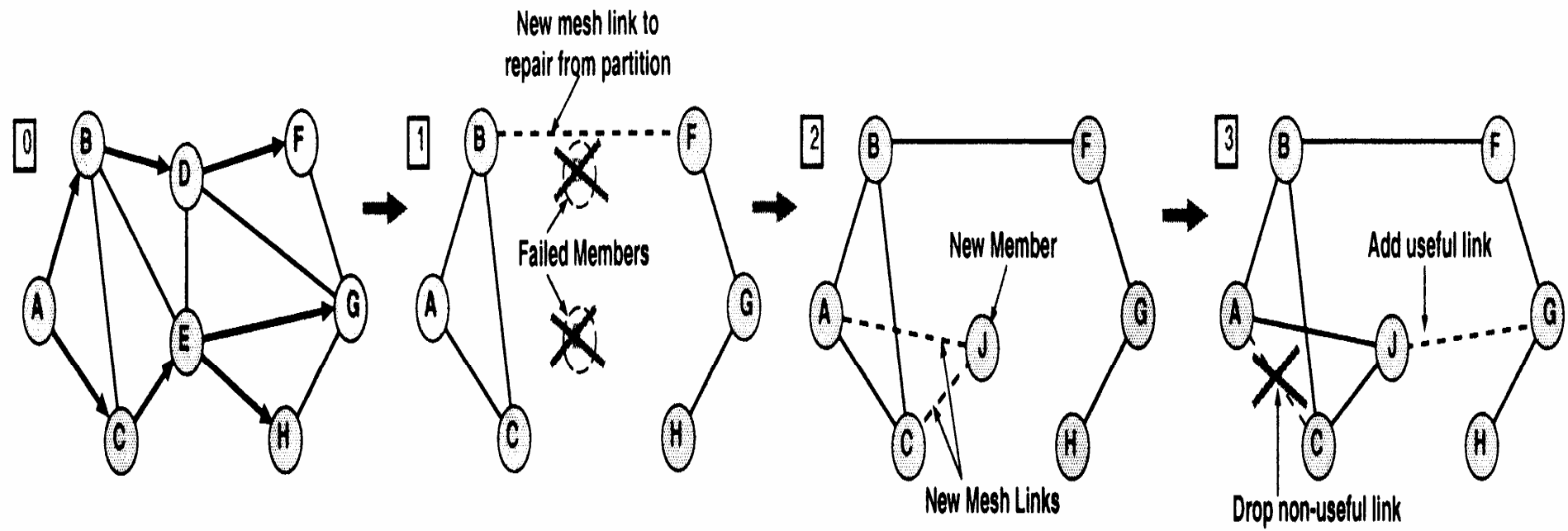
Narada (cont.)

Self-improving, Self-Recovering Mesh Overlay:

Drop Non-useful Mesh Link:

- Cost of a link between *Peers m* and *n* in *m*'s perceptive is the number of peers for which *m* uses *n* as the next hop and vice versa.
- *Peer m* will choose the higher of the two as the consensus cost of the link.
- The link with the lowest consensus cost will be dropped.

Narada (cont.)



0: A is source, arrowed links are data path and also part of mesh.

1: Departure of D, E leads to a mesh partition. Remaining peers detect the departure and B adds a new mesh link to F. Partition repaired.

2: New member joins mesh and sets up mesh-neighbour links with two randomly chosen existing members.

3: Non-useful link deleted from mesh and a useful link added.

Narada (cont.)

Mesh overlay increases connectivity and makes it robust to overlay partitioning.

No single point of failure.

Overlay data tree is optimized for each of the sources for applications with multiple sources.

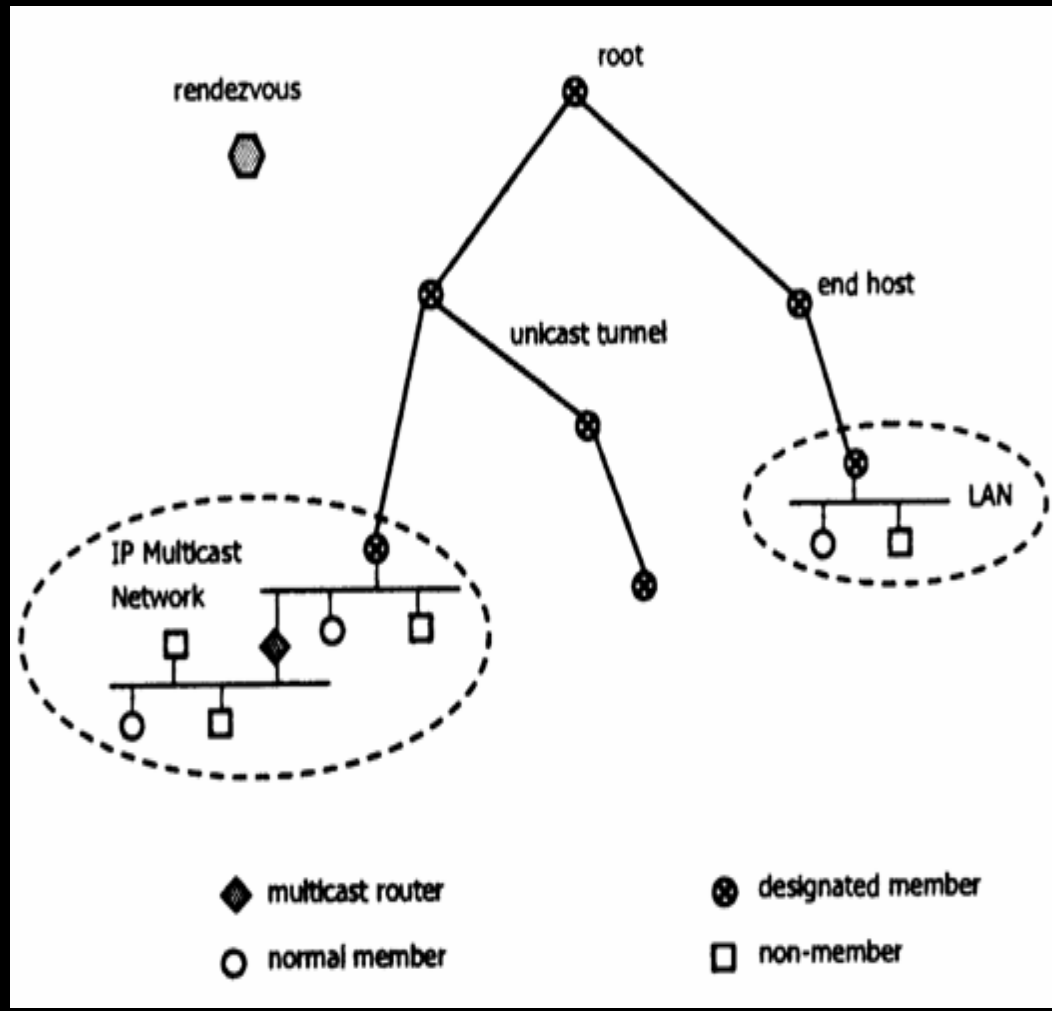
Heavy protocol overhead ($O(N^2)$) given the amount of refresh and update message exchanges among all the peers for full knowledge of group members and routing tables.

Peers need to keep lots of information as it maintains full routing information and full member states for all peers.

Cannot scale to large groups.

Distributed: HM (Host Multicast)

- Best effort delivery.
- Links IP multicast enabled islands and multicast-incapable end hosts via unicast tunnels.
- Multicast islands are connected via UDP tunnels between the Designated Members (DM) where each island elects a DM.
- Data delivery tree is a shared tree where any member can be a source



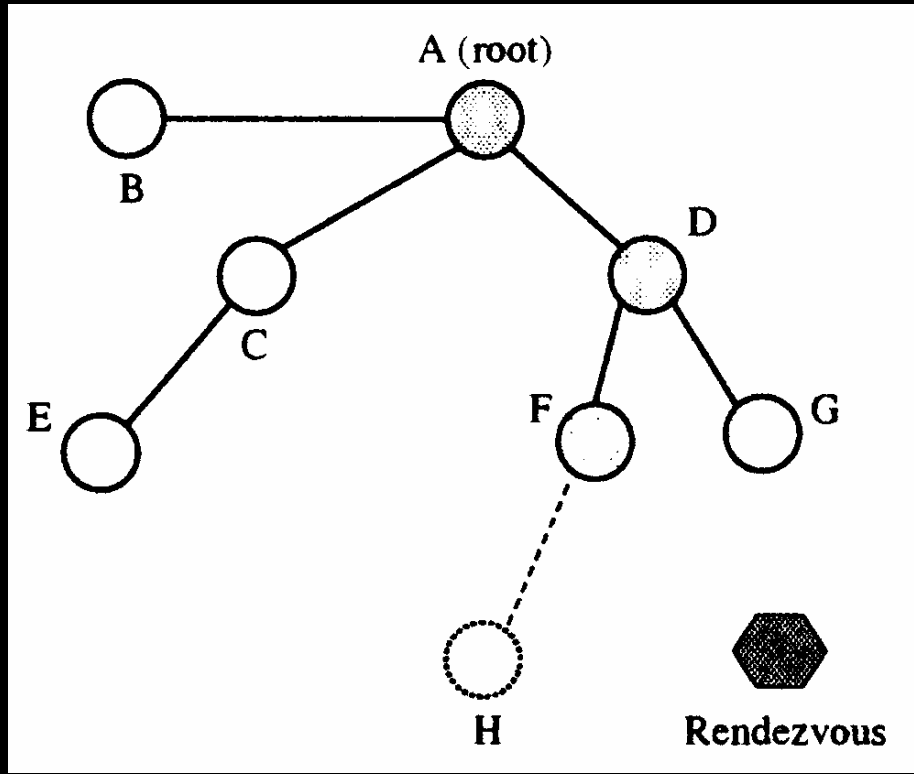
HM (cont.)

Overlay Tree Building Protocol:

- Protocol scales to number of peers, *O(N) (N* is group size).
- New member H discovers the root A of the shared tree through the Rendezvous Point (RP).
- H then sets A as a potential parent and requests for the list of A's children.
- Among A and A's list of children, H picks the closest one in terms of RTT, ie. D is a new potential parent in this case.
- H repeats the process down the tree and finds the next potential parent F.

HM (cont.)

- If H finds that F remains the closest to itself, it issues a join request.
- If F has not breached its degree bound for the number of children, it will accept H's request. Otherwise, H will backtrack by one level and repeat its search.
- Peer-to-peer RTT is collected via end-to-end measurements.



Overlay is built and improved via 'Tree Walk' algorithm.

HM (cont.)

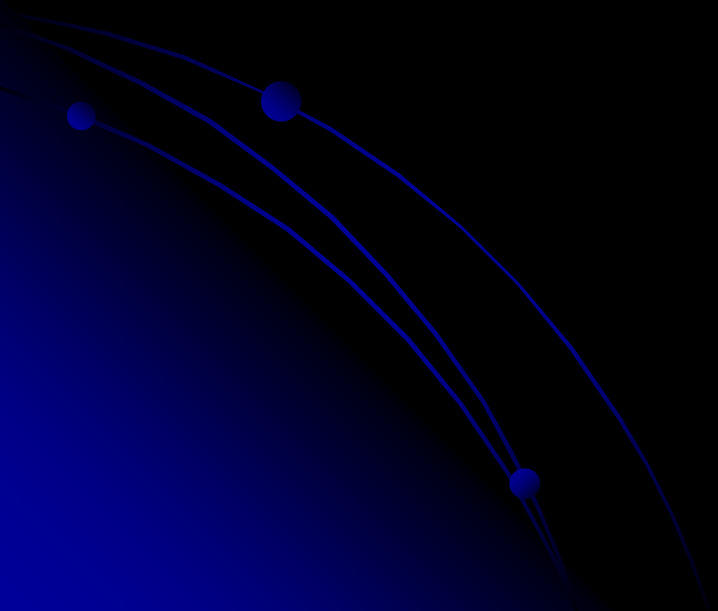
Tree Refinement:

- Each member in HM has to maintain information about all members on its path to the root.
- Updates to root path information are periodically sent by the parent to the children
- Each member will periodically try to find a closer parent by reinitiating the join process from some random member on its root path.
- Loop detection is easily effected as members know its entire root path and avoid using peers in its root path as parent.

HM (cont.)

Recovery from Tree Partition:

- To recover from tree partition, each member can rejoin anyone in its root path or in its cache.
- The cache is built up during the initial process when the member ‘walks’ down the tree.



HM (cont.)

Long join latency and slow recovery from partition due to iterative algorithm to find efficient parent. RTT tests have to be conducted many times during the iterative process.

To speed up the join process, foster out new client to a random parent first until it finds the best permanent parent after the ‘tree walk’.

Less robust than Narada as there is no mesh overlay for increased connectivity.

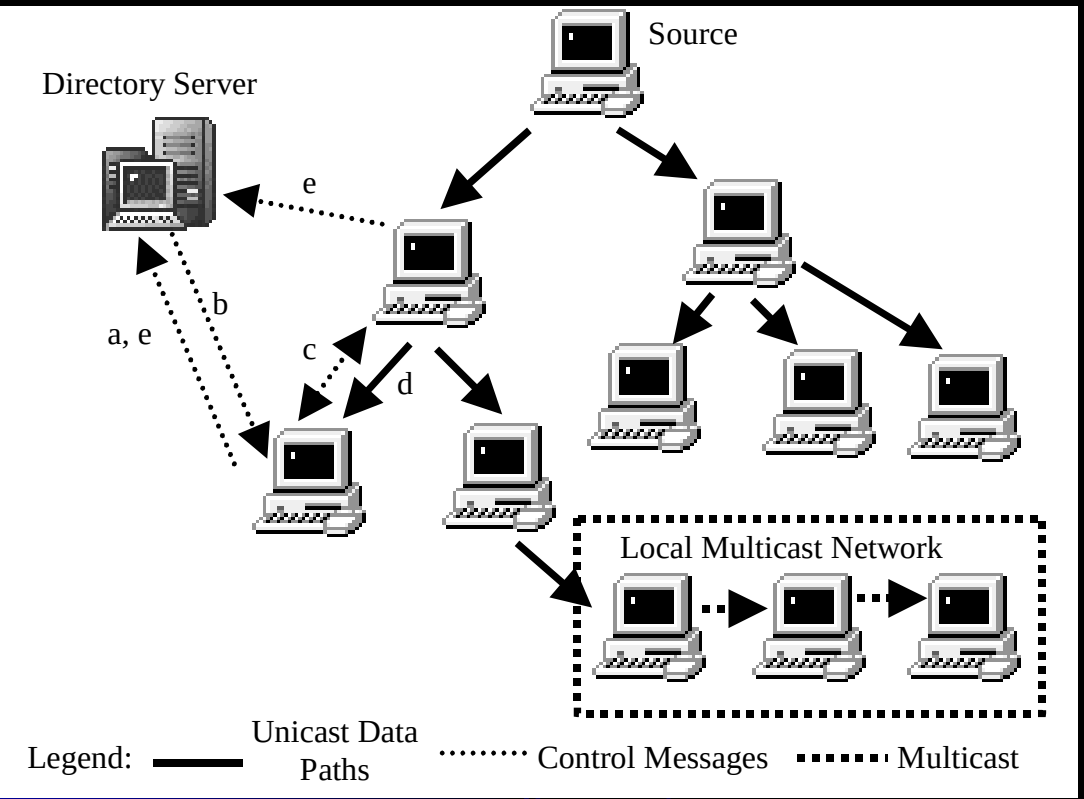
One shared tree for the different sources is not as optimal as Narada’s many source-based trees.

Much less protocol overhead than Narada as it does not need to maintain the entire state information of all the peers and the routing tables.

Hybrid: HPAM

Three main entities:

- Centralized, lightweight directory server (DS).
- Overlay data tree comprising end users built on the fly.
- Protocol control mechanisms (gossips, spirals, RTT tests etc.).



- New client seeks help from DS to join group
- DS returns a set of potential parents
- New client conducts RTT tests to choose the parent who yields the best root latency and also satisfies an acceptable loss rate.

HPAM (cont.)

Exploits simplicity and resourcefulness of lightweight, centralized controller, DS.

- Speeds up distributed tree construction by providing a new client with a list of possible parents.
- Acts as reliable and fast back-up during recovery from tree partition if distributed algorithm fails.
- Does not perform routing computation and topology management.
- Does not stream data. Data distribution still continues even with DS failure. Only glitch is new clients will be prevented from joining the groups.

Integrates the robustness and scaleability of distributed clients. Clients are responsible for tree construction, refinement, repair and routing and data distribution.

HPAM (cont.)

Improve Tree Robustness and Speed up Partition Recovery with 3-layer strategy:

- Layer 1: Hot standby spiral node to adopt orphaned client up to 1 slot beyond its capacity.

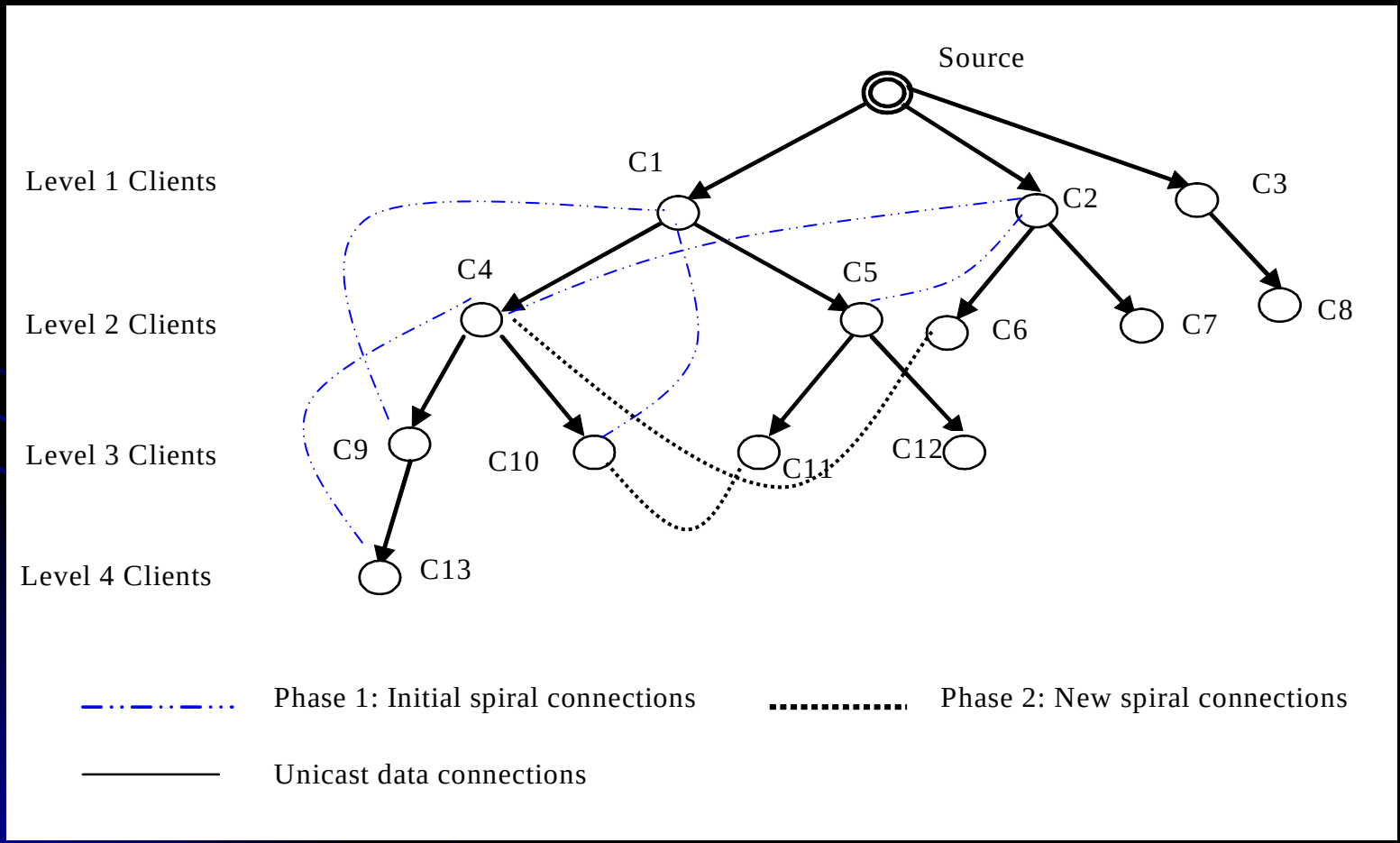
- Layer 2: Ready list in gossip cache as potential parents for orphaned client should spiral node be full or fails.

- Layer 3: Request made to DS to rejoin session.

HPAM (cont.)

Spiral Mechanism:

- Phase 1: Spiral node assigned during join.
- Phase 2: Choose best gossip candidate as spiral node.



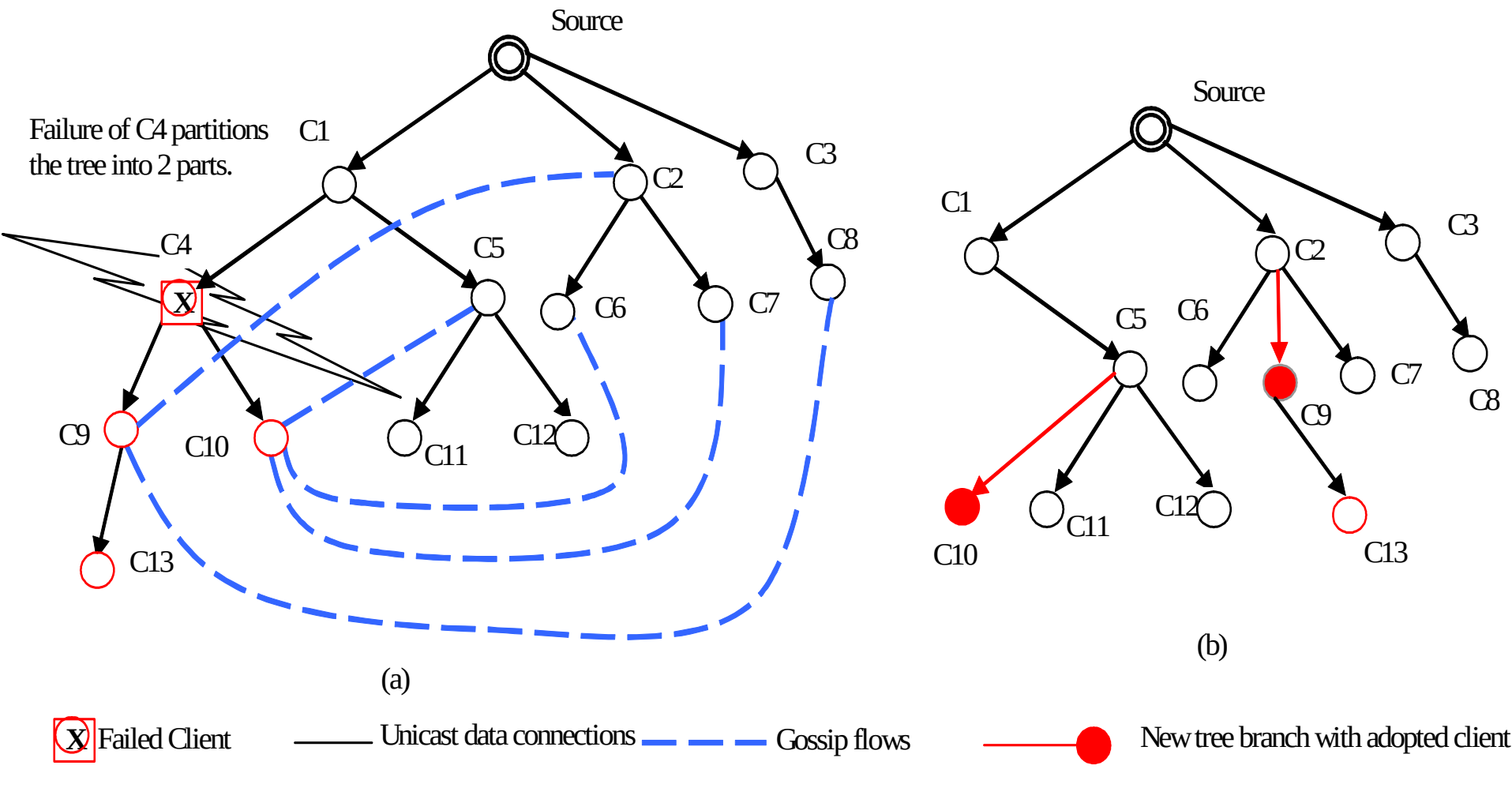
HPAM (cont.)

Gossip Mechanism

- Initial gossip cache formed from potential parent list returned by DS.
- Periodic information exchanges between client and gossiper (eg. Root latency, loss rate, level number).
- Gossip cache refreshed via:
  - Removing gossipers who are full, have poor QoS, have left or have failed.
  - Gossipers supplying client with random selection of gossipers from its own gossip cache.
  - Client requesting DS for supply of fresh parent candidates when gossip cache dwindles below 1/3 of full capacity.

HPAM (cont.)

Gossip Mechanism:



HPAM (cont.)

Self-improving Tree:

- Tree refinement is periodically effected via a client switching to a parent which will provide it with a better QoS.
- The gossiper who yields the best QoS gain for the client is chosen to be the new parent.

$$\Delta Latency_i = (ul_{i,r} + ul_{i,n}) - ul_{n,r}$$

$$QoSGain_i = \frac{\Delta Latency_i - C_{hysteresis} * ul_{n,r}}{C_{hysteresis} * ul_{n,r}}$$

where $i \in$ gossip cache

n is the client

ul is unicast latency

r is the root

$C_{hysteresis} > 0$

HPAM (cont.)

Use of Relative Loss Rate to detect local congestion:

$$RLR_{client} = \frac{LossRate_{client} - LossRate_{parent}}{1 - LossRate_{parent}}$$

- Detection of local congestion can reduce latency and loss rate as well as protocol overheads.
- Client can switch to alternate parent if the current overlay link between itself and its parent is bad.
- Reduce unnecessary parent switching since congestion is not between the client and its current parent.

HPAM (cont.)

Heuristics for Local Congestion Detection

	Parent	Client		Action
	LR	LR	RLR	
Case 1	$\leq L_{th}$	$\leq L_{th}$	x	None
Case 2	$\leq L_{th}$	$> L_{th}$	x	Local congestion. Switch parent.
Case 3	$> L_{th}$	$> L_{th}$	$> L_{th}$	Likely local congestion. Switch parent.
Case 4	$> L_{th}$	$> L_{th}$	$\leq L_{th}$	Minimal local congestion. Wait for parent to switch first. If parent fails to switch, then start to switch parent.

x: don't care

L_{th}: Acceptable loss rate

HPAM (cont.)

Strength in simplicity

Quick in constructing efficient overlay trees

Combination of lightweight, centralized DS with distributed tree building, refinement and repair techniques allow a faster join, swifter partition recovery compared to distributed protocols.

- More responsive to changes in membership dynamics which is achieved at very much lower overheads.

Single Point of Failure in DS

Scaleability concern. DS cannot scale to very large groups.

Hybrid, Proxy-based OMNI

Builds a single-source overlay tree from a set of service nodes called multicast service nodes (MSN) deployed in the network

Targeted at large scale media-streaming applications.

MSNs can leave and join the session like normal peers.

Minimizes latency for overall tree.

Key feature: OMNI gives a dynamic priority to the different MSNs based on the size of its service set (i.e. the number of clients it serves) and iteratively optimizes the overlay tree.

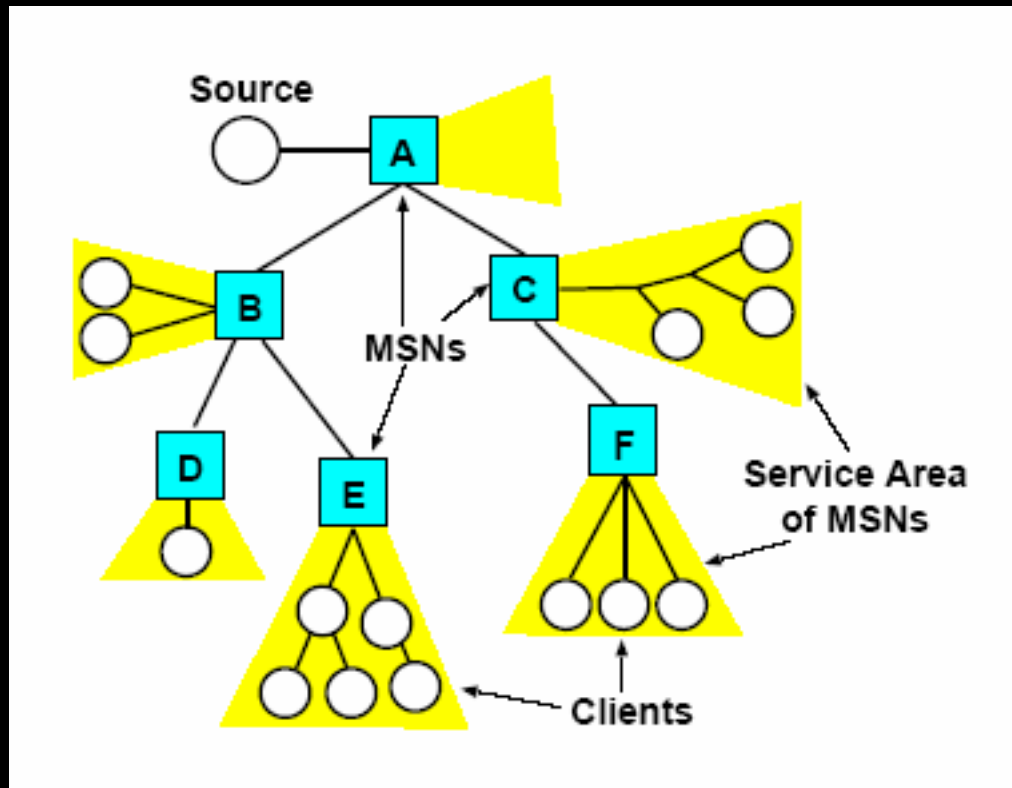
OMNI (cont.)

Two-stage Tree Building Process:

- Stage 1 - Offline Initialization Phase (centralized approach):
  Takes place before data delivery starts and before the dynamic self-organizing process during data delivery.

Root MSN (A) is connected to source. It measures and sorts the list of MSNs in increasing order of unicast latencies from itself.

It then builds a tree always choosing the nearest MSN to itself at each level of the tree. (D, E and F are further from A in latency terms than B and C.)



OMNI (cont.)

- Stage 2 - Dynamic Self-Organizing Phase (distributed approach)

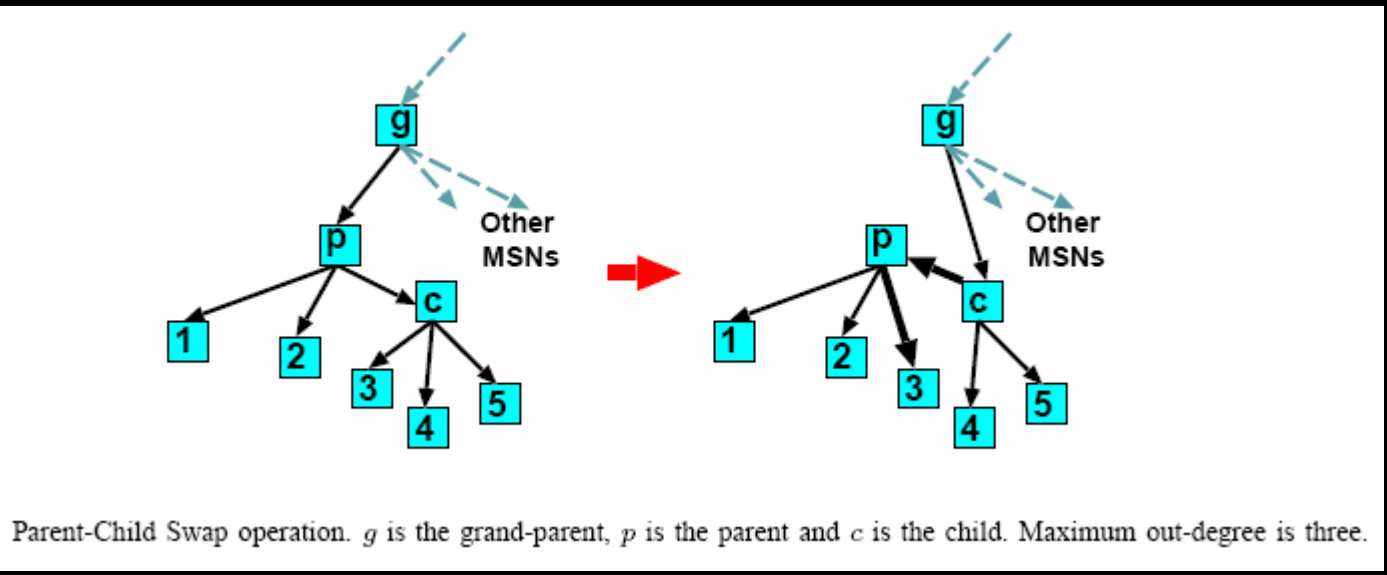
Takes place after data delivery starts

a. To adapt the overlay to changing latencies, each MSN performs periodic swapping among themselves if such swapping can reduce their current average latency.

This swapping is strictly local in the sense that it is confined to within two levels of each other.

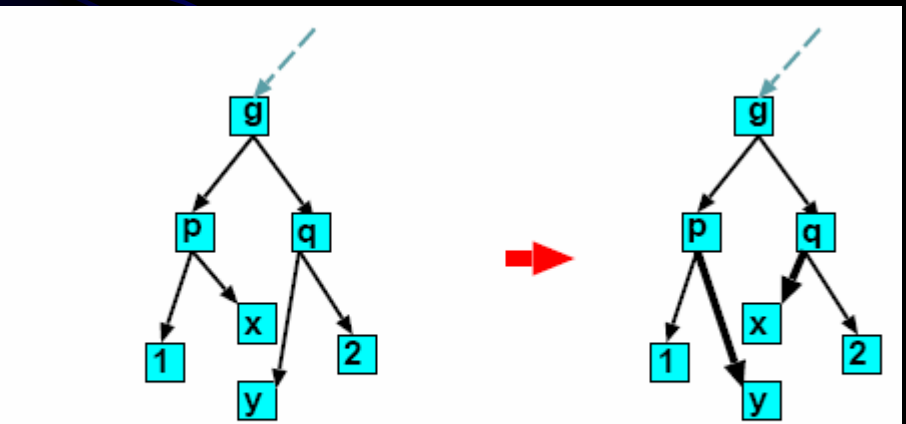
b. OMNI also uses simulated annealing to probabilistically swap one MSN with a random member not within the two levels to allow the overlay to reach a global minima in terms of latency.

OMNI (cont.)

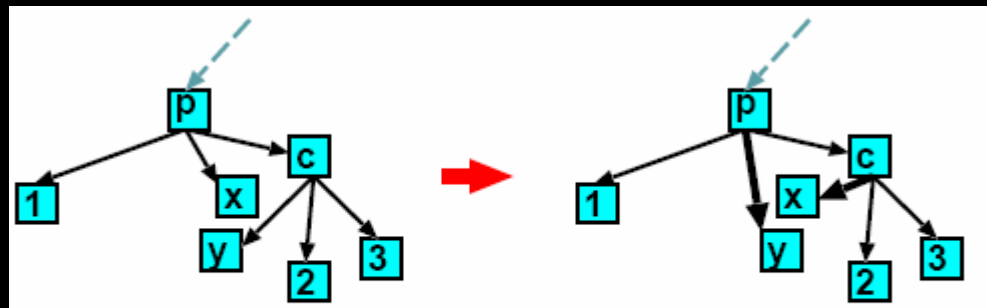


Parent-Child Swap operation. g is the grand-parent, p is the parent and c is the child. Maximum out-degree is three.

Periodic transformations performe
to reduce aggregate sub-tree
latency for subtree under node g



Iso-level-2 Swap operation. g is the grand-parent, p and q are siblings. x and y are swapped.

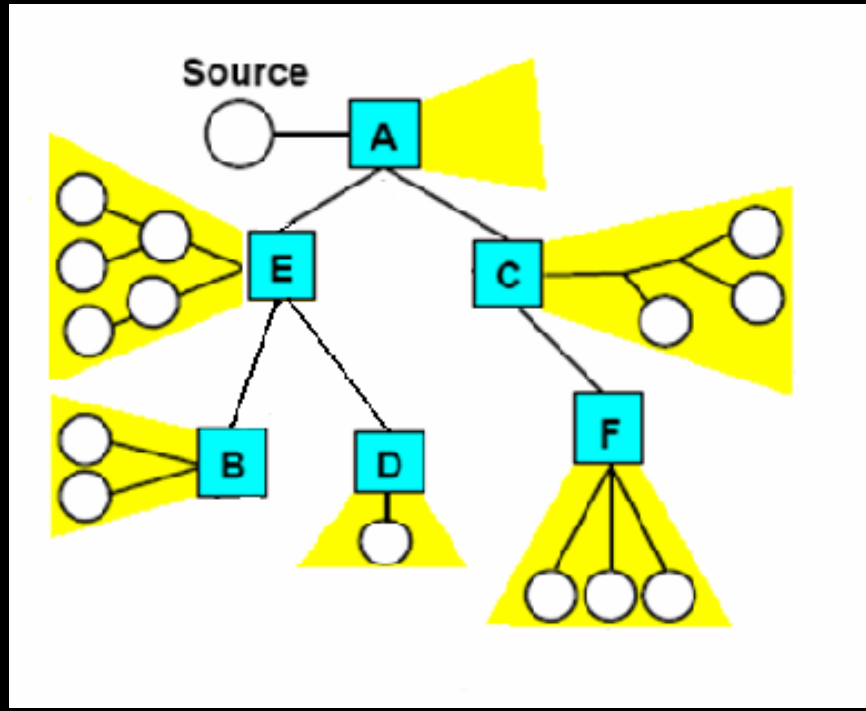


Aniso-level-1-2 Swap operation. p is the parent of c. x and y are swapped.

OMNI (cont.)

c. Swapping based on increase in importance of MSN as its peer distribution increases.

   As the number of peers under an MSN increases, it will check if it can migrate up the tree by swapping to improve its latencies from the root MSN.

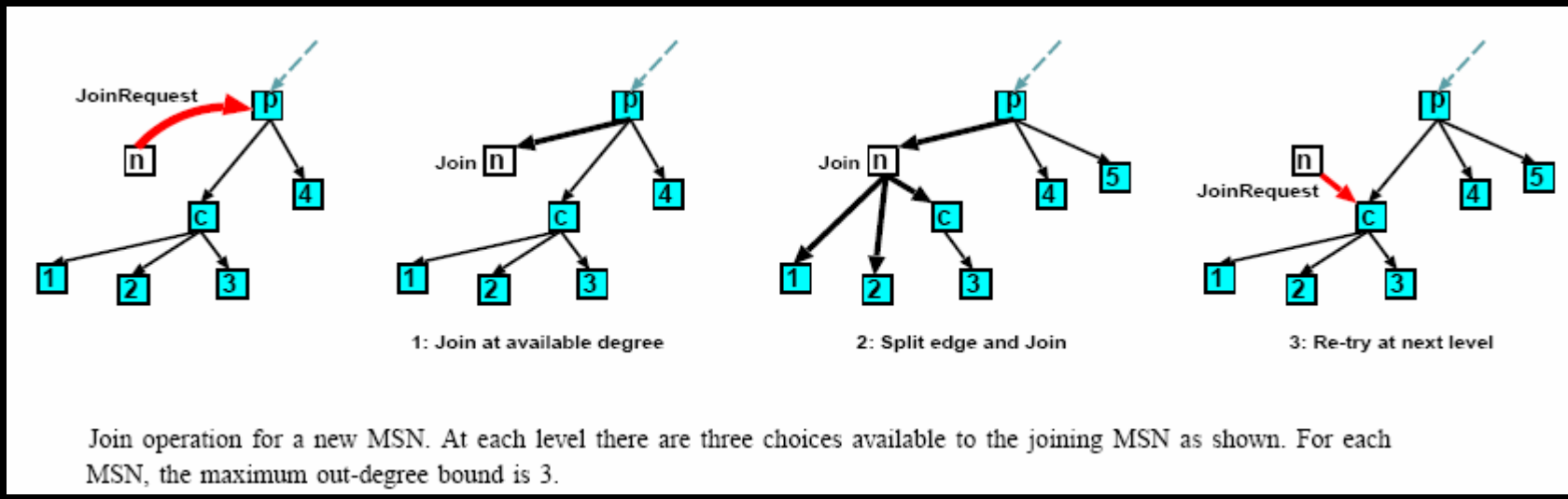


Each MSN maintains state for all its tree neighbours, its ancestors and the overlay path from the root to itself.

The root path is also used for loop detection during tree optimizations.

OMNI (cont.)

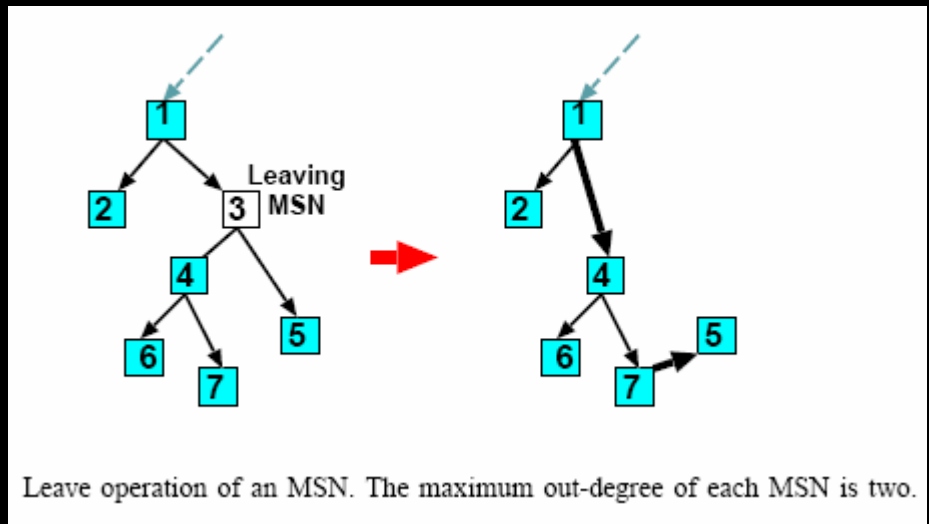
Join Operation



Join operation for a new MSN. At each level there are three choices available to the joining MSN as shown. For each MSN, the maximum out-degree bound is 3.

Leave Operation

Assumes failure is rare as MSNs are specially managed infrastructure entities. MSN always leave gracefully.



Leave operation of an MSN. The maximum out-degree of each MSN is two.

OMNI (cont.)

Hybrid approach with off-line optimized initial tree building and distributed real-time swapping

Assumes that most MSNs are available before start of streaming session

No handling of MSN failure

Peers will join the MSNs closest to them

Peer under one MSN cannot migrate to another MSN to improve its latency but has to rely on its MSN to act on its behalf.

Conclusion

Application Level Multicast is a fast and economical means to overcome deployment barriers to network-level solutions for multipoint communication.

Advantages include flexibility, adaptivity, enhanced functionalities and ease of deployment.

Performance penalty include less bandwidth efficiency and increased latency over IP multicast as well as increased protocol overheads.

Future Directions

1. Tree or no tree, that is the question.

   - Tree is efficient but complex to build and is fragile

   - Non-tree systems such as embedded system (eg. Content Addressable Network CAN [6]), swam-style, P2P multicast (coolstreaming [7]) have a lot of overheads

   Embedded approach assigns to members of the overlay network logical addresses from some abstract coordinate space and builds an overlay with the help of these logical addresses.

   By embedding neighbour mappings in member addresses, next-hop routing of messages can be performed without the need for full fledged routing protocols

Future Directions (cont.)

Swarm-style P2P multicast has node advertises to its neighbors the data it has received, and the neighbors explicitly request blocks as needed. No tree is built.

Interest triggered by popularity of live IPTV.

Coolstreaming (Cooperative Overlay Streaming) [7]uses gossip for membership management

New member joins source and source appoints a deputy to provide a list of gossip candidates

Future Directions (cont.)

2. Hierarchical approach with clustering of peers to reduce protocol overhead and increase scaleability. Each cluster of peers has a leader (eg. NICE [16])

3. Security issues in ALM

4. Detection and prevention of dishonest peers (freeloaders, freeriders, cheats) who are willing to take but unwilling to share

5. Pricing models and issues to render ALM to be commercially viable

References

1. D. Pendarakis, S. Shi, D. Verma and M. Waldvogel, “ALMI: An Application Level Multicast Infrastructure,” *Proc. 3rd Usenix Symposium on Internet Technologies & Systems (USITS),* March 2001.
2. Y. Chu, S. G. Rao, S. Seshan and H. Zhang, “A Case for End System Multicast,” *IEEE Journal on Selected Areas in Communications*, 20(8), October 2002.
3. B. Zhang, S. Jamin and L. Zhang, “Host Multicast: A Framework for Delivering Multicast to End Users,” *Proc. INFOCOM*, June 2002.
4. C. K. Yeo, B. S. Lee and M. H. Er, “A Framework for Multicast Video Streaming over IP Networks,” *Journal of Networks and Computer Applications*, 26(3), pp. 273-289, August 2003.
5. Banerjee, C, Kommareddy, K. Kar, B. Bhattacharjee and S. Khulle, “Construction of an Efficient Overlay Multicast Infrastructure for Real-time Applications”, *Proc. INFOCOM*, April 2003.
6. S. Ratnasamy, P. Francis, M. Handley and R. Karp, “A Scaleable Content-Addressable Network”, *Proc. ACM SIGCOMM*, August 2001.
7. X. Zhang, J. Liu, B. Li, and T.-S. P. Yum. DONet/CoolStreaming: A Data-driven Overlay Network for Live Media Streaming. In *Proceedings of IEEE Infocom*, Miami 2005.
8. S. Banerjee, B. Bhattacharjee and C. Kommareddy, “Scaleable Application Layer Multicast,” *Proc. ACM SIGCOMM*, August 2002.
9. A. El-Sayed, V. Roca, L. Mathy, “A Survey of Proposals for an Alternative Group Communication Service”, *IEEE Network,* January/February 2003.
10. C. K. Yeo, B. S. Lee, M. H. Er, “A Survey of Application Level Multicast Techniques,” *Computer Communications,* 27(15), pp. 1547-1568, September 2004.

Thank you for your attention

Hope to see you in Singapore

Good-bye