P2P Media Streaming

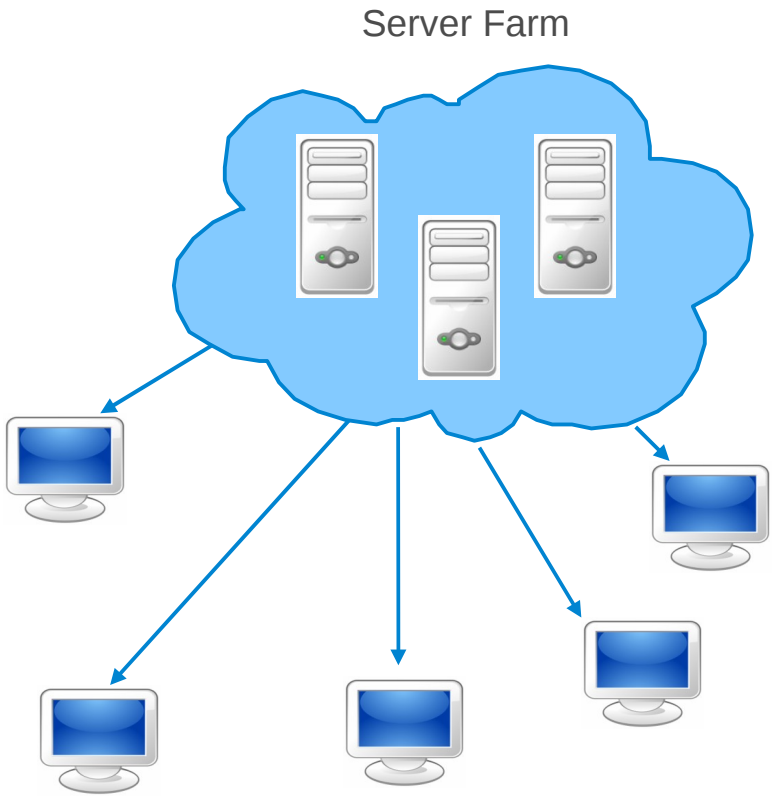
Seif Haridi (haridi@kth.se)
Fatemeh Rahimian (fatemeh@sics.se)
Amir H. Payberah (amir@sics.se)

Content Distribution Network

Content Distribution Network

- CDN is a system of computers, networked together that cooperate transparently to deliver content to end users.

Client – Server



Client – Server

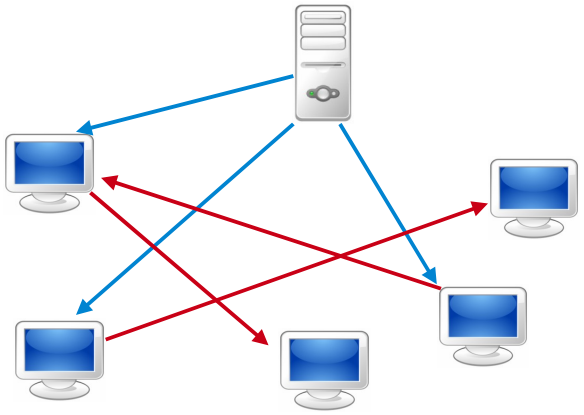
- What is the problem of Client-Server model? [d]

Client – Server

- What is the problem of Client-Server model? [d]

Peer-to-Peer

- The peers can help each other.
- The peers who have parts of the data can forward it to other requesting peers.
- The capacity increases with the number of peers.



P2P Media Streaming

Media Streaming

- Media streaming is a multimedia that is sent over a network and played as it is being received by end users.
- Users do not need to wait to download all the media.
- They can play it while the media is delivered by the provider.

Media Streaming

- Live Media Streaming
  - The streams are only available at one particular time.

- Video on Demand (VoD)
  - The streams are stored on a server and are available to be transmitted at a user's request.
  - It provides a large subset of VCR functionality, e.g. pause, fast forward, fast rewind and ...

P2P Media Streaming Challenges

- Bandwidth intensive.

P2P Media Streaming Challenges

- Bandwidth intensive.

- Data should be received with respect to certain timing constraints.
  - A negligible startup delay
  - Smooth playback
  - A negligible playback latency (only for Live Streaming)

P2P Media Streaming Challenges

- Bandwidth intensive.
- Data should be received with respect to certain timing constraints.
  - A negligible startup delay
  - Smooth playback
  - A negligible playback latency (only for Live Streaming)
- Nodes join, leave and fail continuously.
  - Called churn

P2P Media Streaming Challenges

- Bandwidth intensive.
- Data should be received with respect to certain timing constraints.
  - A negligible startup delay
  - Smooth playback
  - A negligible playback latency (only for Live Streaming)
- Nodes join, leave and fail continuously.
  - Called churn
- Network capacity changes.

Related Work

- SplitStream
- DONet/Coolsteraming
- CoopNet
- Orchard
- Bullet
- Prime
- Pulsar
- NICE
- Zigzag
- DirectStream
- MeshCast

- mtreeBone
- PULSE
- GnuStream
- SAAR
- ChainSaw
- ChunkySpread
- BulkTree
- ForestCast
- AnySee
- DagStream
- Climber

- CollectCast
- HyMoNet
- GridMedia
- Promise
- Yoid
- Zebra
- Tribler
- CliqueStream
- GradienTv
- Sepidar

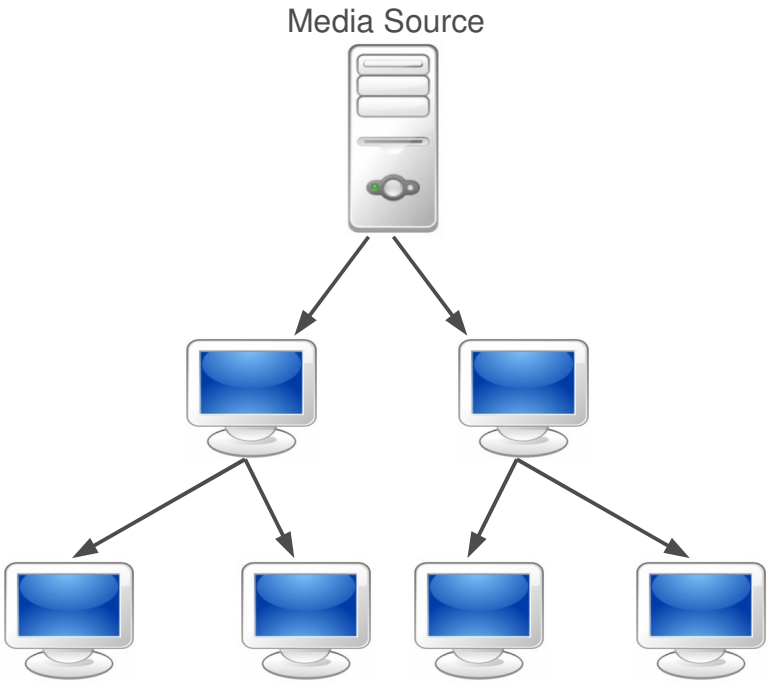
Two Questions

- What overlay topology is built for data dissemination?
- How to construct this overlay?

What overlay topology is built for data dissemination?

Single Tree Structure

- Build a single multicast tree, in which the root is the media source and the interior nodes and leaves are peers.
- The media is pushed from the root to interior nodes to leaf nodes.



Single Tree Advantage/Disadvantage?

- Advantage/Disadvantage [d]

Single Tree Advantage/Disadvantage?

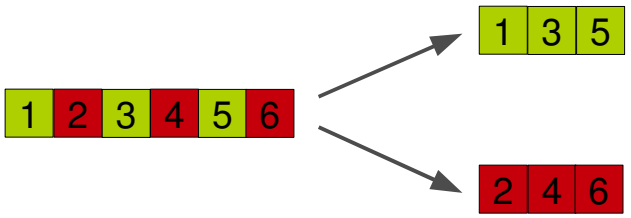
- Advantage/Disadvantage [d]

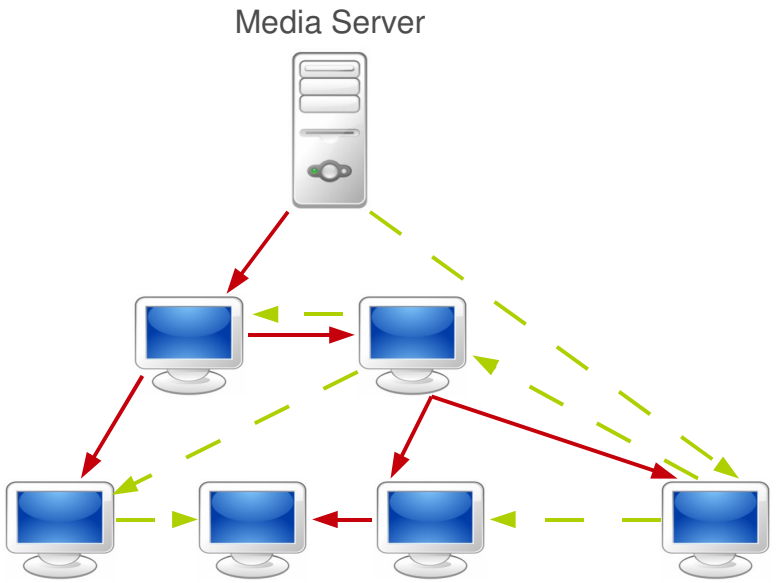
- Advantage
  - The short latency of data delivery.
  - Easy to implement.

- Disadvantage
  - The fragility of the tree structure upon the failure of nodes close to the root.
  - All the traffic is only forwarded by the interior nodes.

Multiple-Tree Structure

- The media source splits the stream into a set of sub-streams.
- A single tree is created for each sub-stream.
- A peer to receive the whole media should join all trees.





Multiple-Tree Advantage/Disadvantage?

- Advantage/Disadvantage [d]

Multiple-Tree Advantage/Disadvantage?

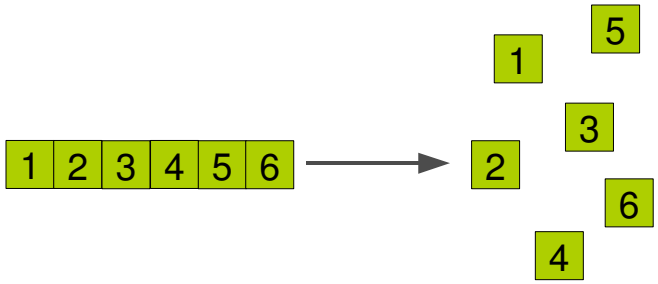
- Advantage/Disadvantage [d]

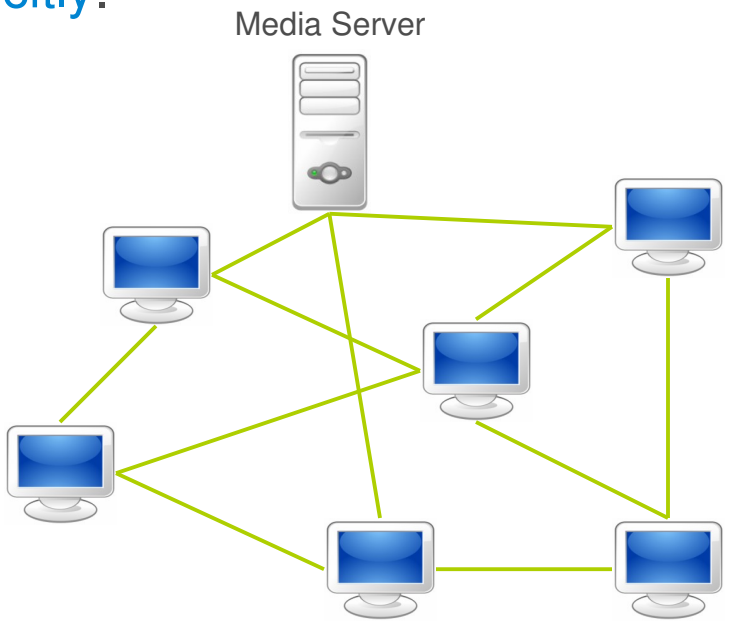
- Advantage
  - Resilient to node failure.
  - Good load balancing

- Disadvantage
  - Difficult to implement.

Mesh-based Structure

- The media source into small blocks.
- Nodes are connected in a mesh-network.
- Nodes pull missing blocks of data explicitly.





Mesh Advantage/Disadvantage?

- Advantage/Disadvantage [d]

Mesh Advantage/Disadvantage?

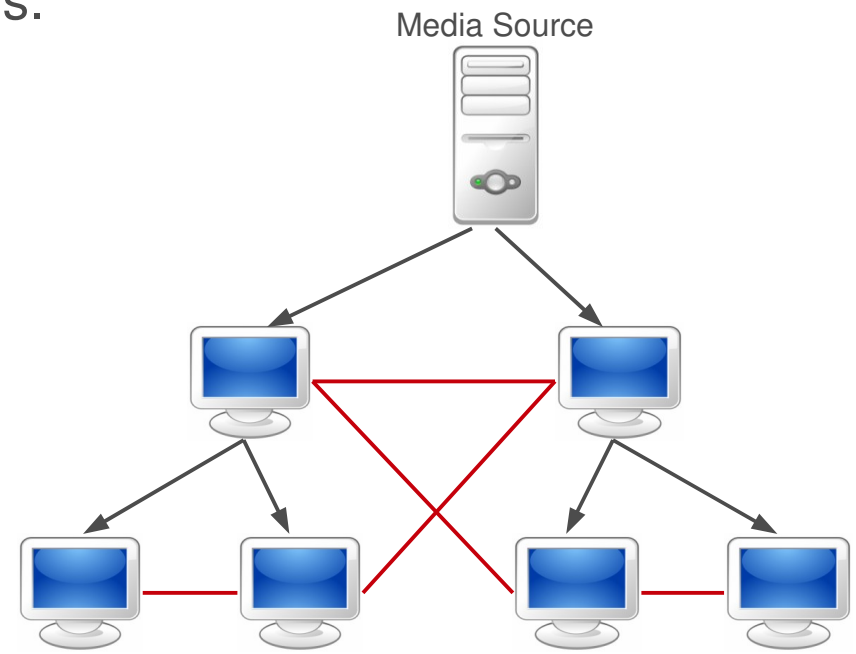
- Advantage/Disadvantage [d]

- Advantage
  - Resilient to node failure
  - Good load balancing
  - Easy to implement

- Disadvantage
  - Unpredictable latencies due to the frequent exchange of notifications and requests.

Mesh-Tree Structure

- Combine tree and mesh structures to construct a data delivery overlay.
- Usually blocks are pushed through the tree and missed blocks are pulled from the mesh neighbours.



Mesh-Tree Advantage/Disadvantage?

- Advantage/Disadvantage [d]

Mesh-Tree Advantage/Disadvantage?

- Advantage/Disadvantage [d]
- Advantage
  - Resilient to node failure
  - Good load balancing
  - Easy to implement
- Disadvantage
  - ?

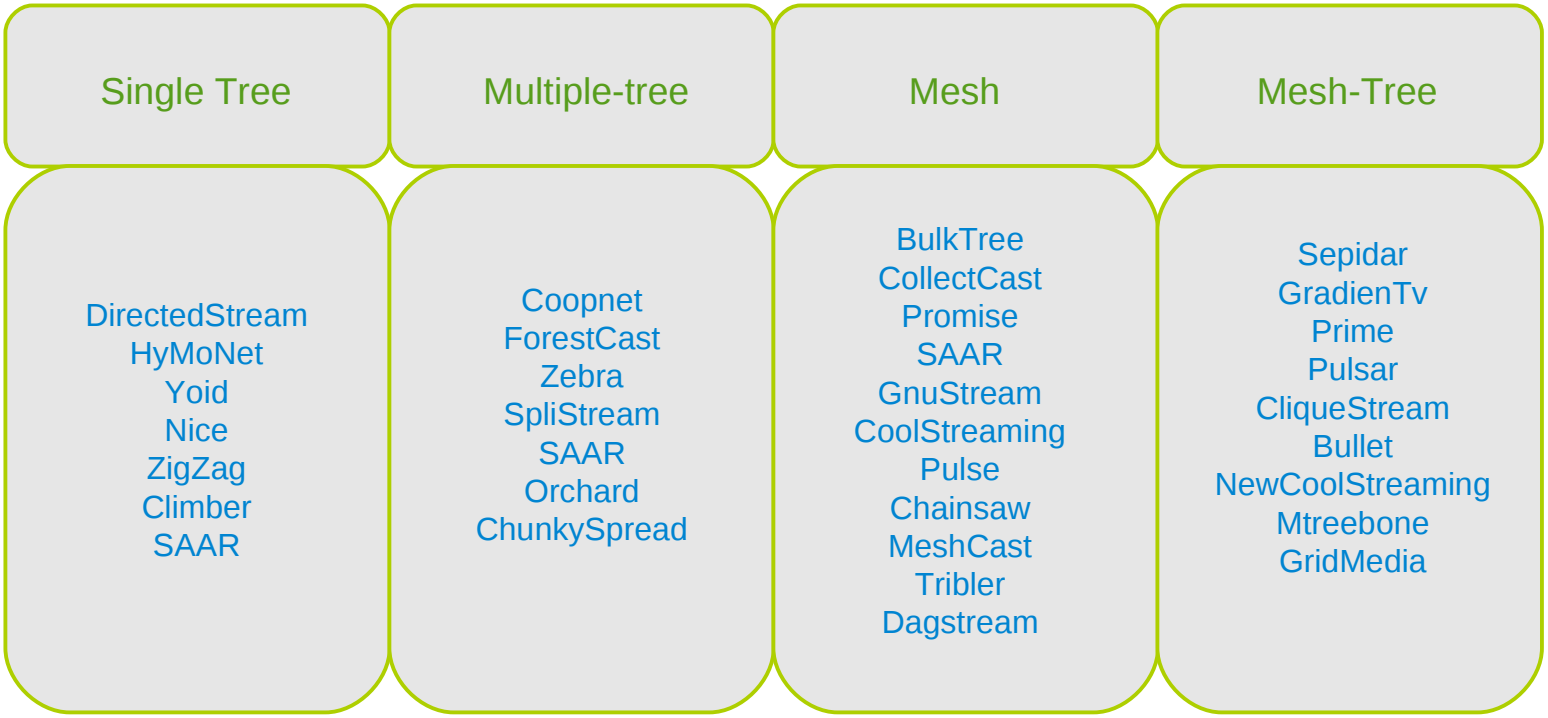
Back to the Related Work

- SplitStream
- DONet/Coolsteraming
- CoopNet
- Orchard
- Bullet
- Prime
- Pulsar
- NICE
- Zigzag
- DirectStream
- MeshCast

- mtreeBone
- PULSE
- GnuStream
- SAAR
- ChainSaw
- ChunkySpread
- BulkTree
- ForestCast
- AnySee
- DagStream
- Climber

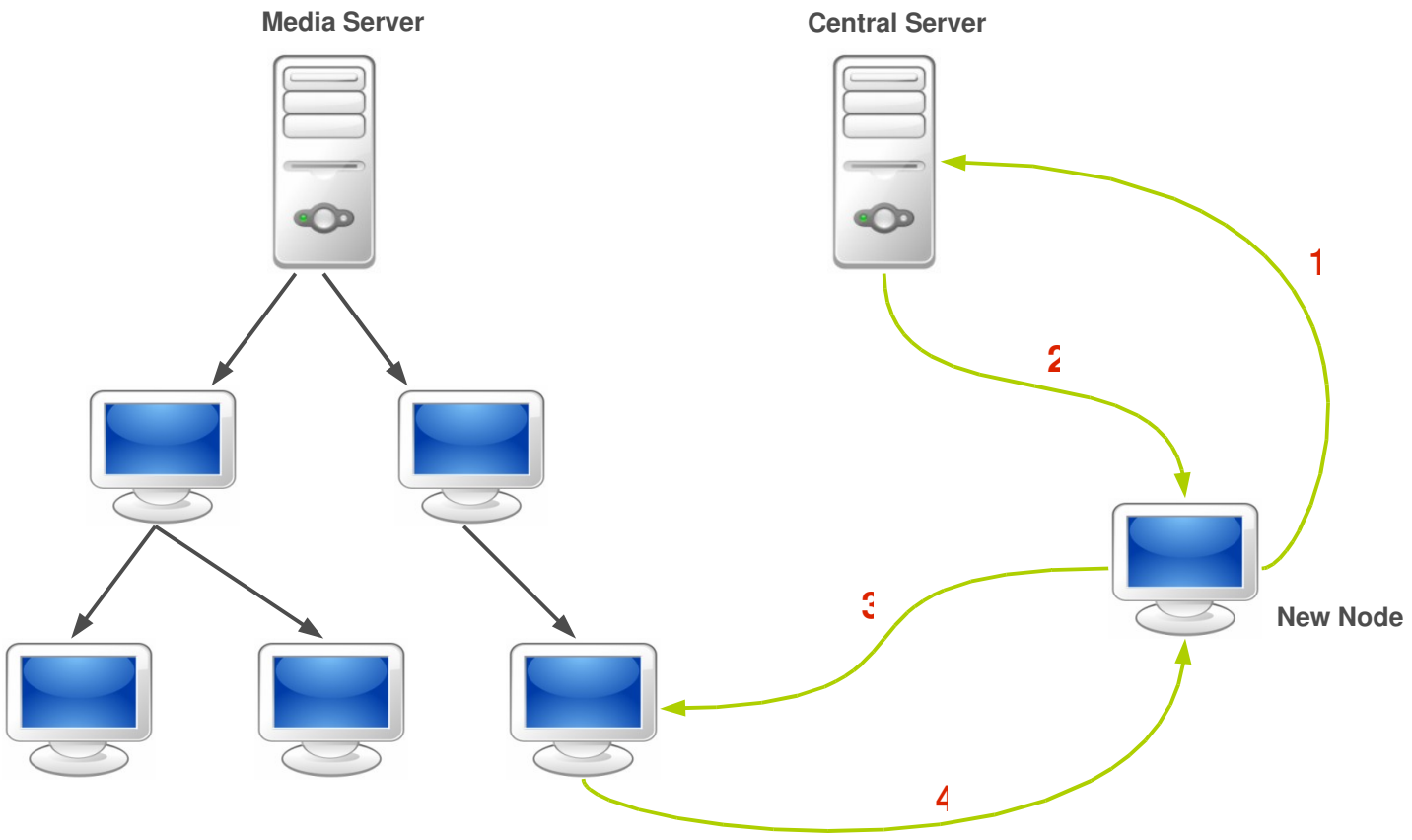
- CollectCast
- HyMoNet
- GridMedia
- Promise
- Yoid
- Zebra
- Tribler
- CliqueStream
- GradienTv
- Sepidar

Data Dissemination Topology



How to construct the overlay?

Centralized Method



Centralized Advantage/Disadvantage?

- Advantage/Disadvantage [d]

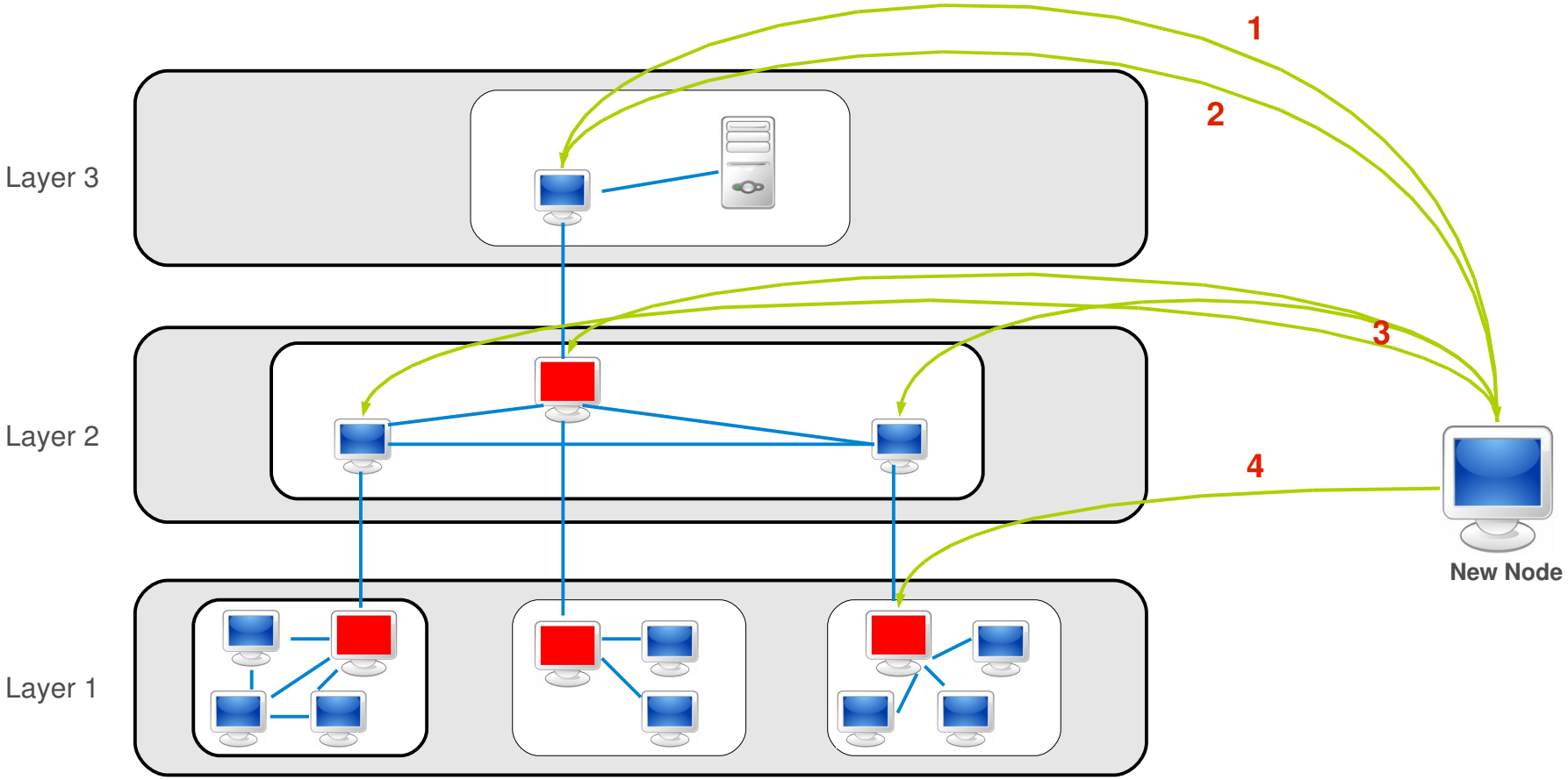
Centralized Advantage/Disadvantage?

- Advantage/Disadvantage [d]

- Advantage
  - Fast
  - Easy to apply optimization methods.
  - Easy to implement.

- Disadvantage
  - Not scalable
  - Single point of failure

Hierarchical Method



Hierarchical Advantage/Disadvantage?

- Advantage/Disadvantage [d]

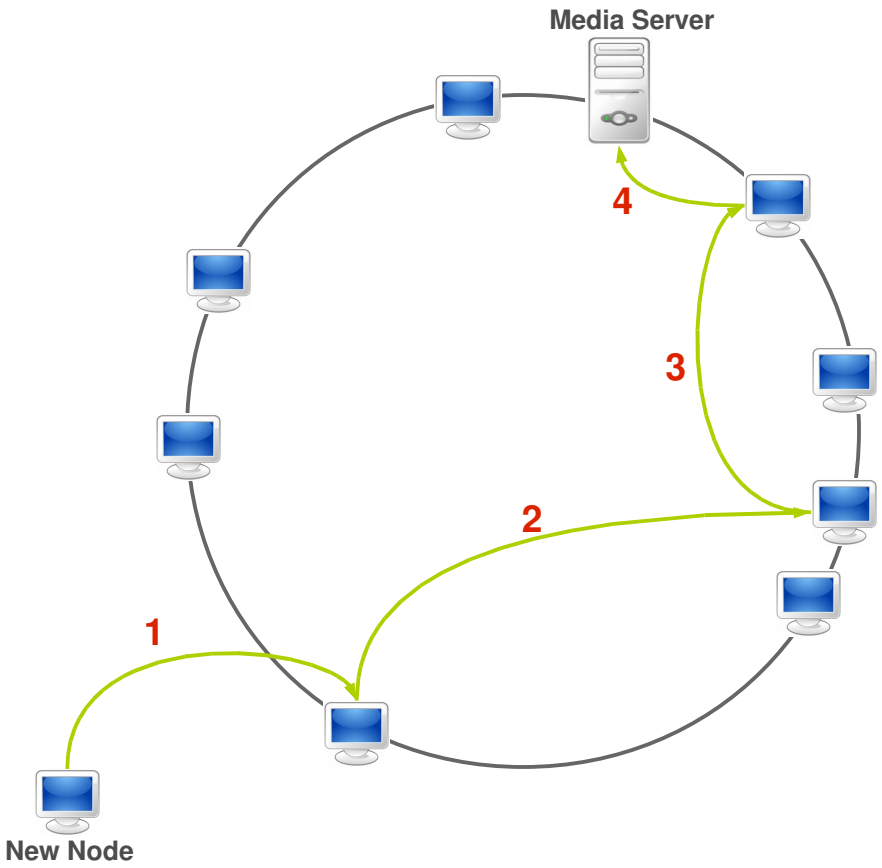
Hierarchical Advantage/Disadvantage?

- Advantage/Disadvantage [d]

- Advantage
  - Scalable.
  - No single point of failure.

- Disadvantage
  - Slow convergence
  - Difficult to implement

DHT-based Method



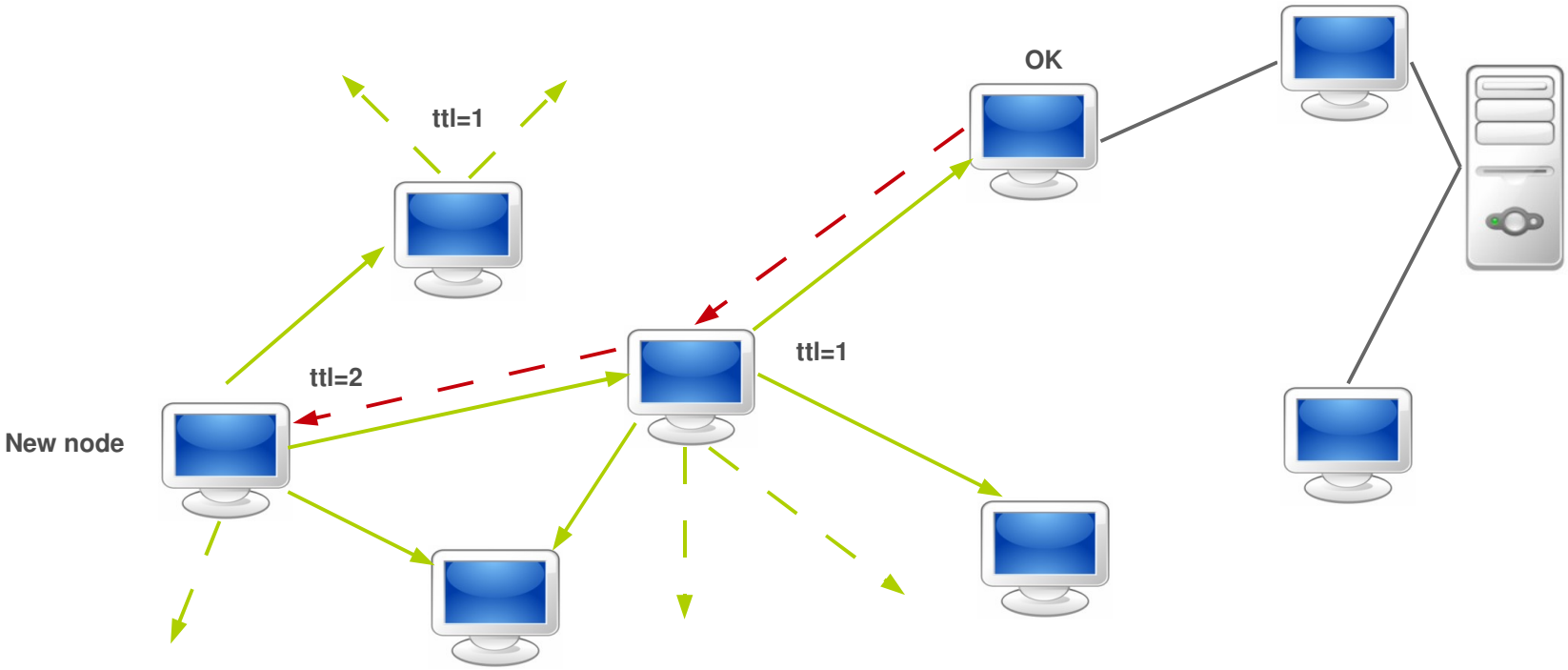
DHT-based Advantage/Disadvantage?

- Advantage/Disadvantage [d]

DHT-based Advantage/Disadvantage?

- Advantage/Disadvantage [d]
- Advantage
  - Scalable.
  - No single point of failure.
- Disadvantage
  - Difficult to implement

Controlled Flooding Method



Flooding Advantage/Disadvantage?

- Advantage/Disadvantage [d]

Flooding Advantage/Disadvantage?

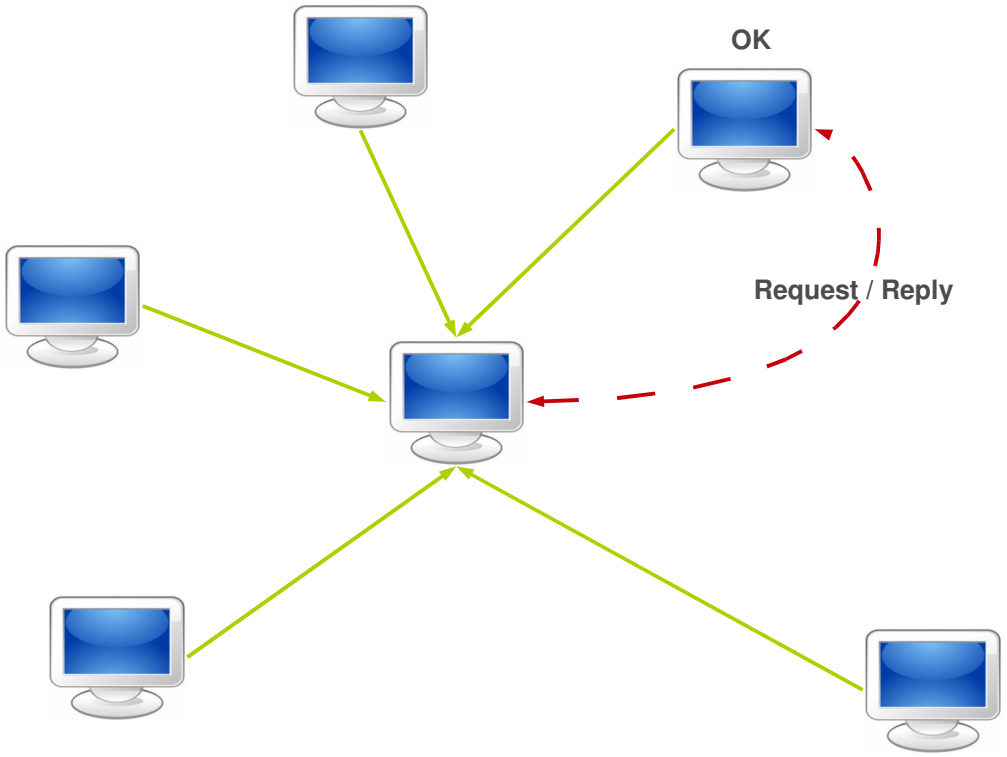
- Advantage/Disadvantage [d]

- Advantage
  - Scalable.
  - No single point of failure.

- Disadvantage
  - No guarantee to find supplier node
  - Slow convergence

Gossip-based Method

- Peers periodically send their data availability to their neighbours.



Gossip-based Advantage/Disadvantage?

- Advantage/Disadvantage [d]

Gossip-based Advantage/Disadvantage?

- Advantage/Disadvantage [d]

- Advantage
  - Scalable.
  - No single point of failure.
  - Easy to implement

- Disadvantage
  - No guarantee to find supplier node in time

Back to the Related Work

- SplitStream
- DONet/Coolsteraming
- CoopNet
- Orchard
- Bullet
- Prime
- Pulsar
- NICE
- Zigzag
- DirectStream
- MeshCast

- mtreeBone
- PULSE
- GnuStream
- SAAR
- ChainSaw
- ChunkySpread
- BulkTree
- ForestCast
- AnySee
- DagStream
- Climber

- CollectCast
- HyMoNet
- GridMedia
- Promise
- Yoid
- Zebra
- Tribler
- CliqueStream
- GradienTv
- Sepidar

Node Discovery Methods

Centralized	DirectedStream, HyMoNet, Yoid, CoopNet ForestCast, Zebra, Prime
Hierarchical	NICE, ZigZag, Climber, BulkTree, Prime
DHT-based	SAAR, SplitStream, CollectCast, Promise, CliqueStream, Pulsar
Flooding	GnuStream
Gossip-based	Sepidar, GradienTv, Orchard, ChunkySpread, CoolStreaming, Pulse, Chainsaw MeshCast, Tribler, DagStream, Bullet, mTreebone, GridMedia

All Together

	Single tree	Multiple-tree	Mesh	Mesh-Tree
Centralized	DirectedStream HyMoNet Yoid	Coopnet ForestCast Zebra		Prime
Hierarchical	NICE ZigZag Climber		BulkTree	Prime
DHT-based	SAAR	SAAR SplitStream	SAAR CollectCast Promise	Pulsar CliqueStream
Flooding			GnuStream	
Gossip-based		Orchard ChunkySpread	CoolStreaming - Pulse Chainsaw - MeshCast Tribler - DagStream	Sepidar - GradienTv Bullet - mTreebone GridMedia

All Together

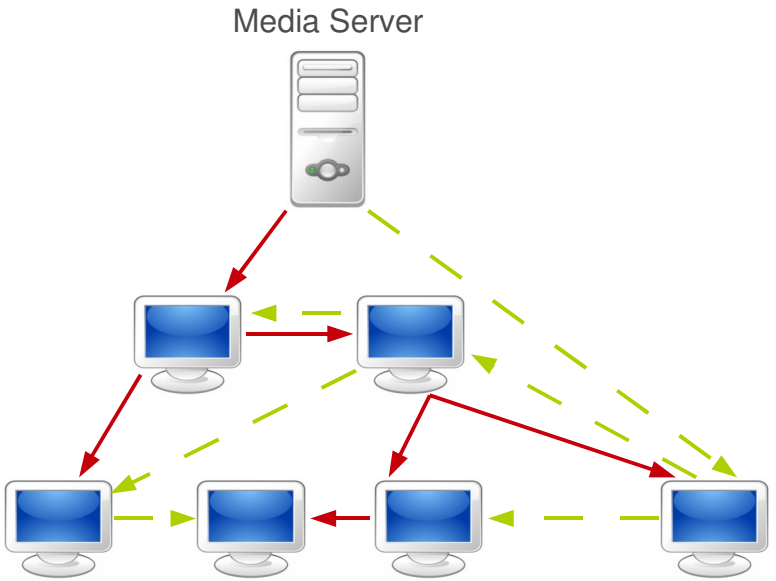
	Single tree	Multiple-tree	Mesh	Mesh-Tree
Centralized	DirectedStream HyMoNet Yoid	Coopnet ForestCast Zebra		Prime
Hierarchical	NICE ZigZag Climber		BulkTree	Prime
DHT-based	SAAR	SAAR SplitStream	SAAR CollectCast Promise	Pulsar CliqueStream
Flooding			GnuStream	
Gossip-based		Orchard ChunkySpread	CoolStreaming - Pulse Chainsaw - MeshCast Tribler - DagStream	Sepidar - GradienTv Bullet - mTreebone GridMedia

SplitStream

(DHT-based – Multiple-Tree)

SplitStream

- Splitting data into stripes, each is sent over its own tree.
- Build on Pastry and Scribe.

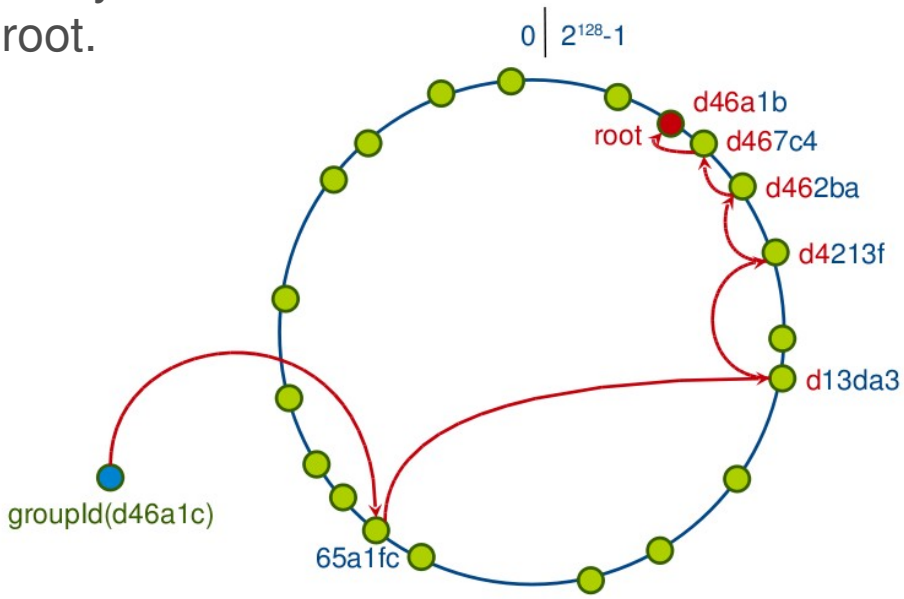


Scribe

- Application-level multicast structure.
- Build on top of Pastry.

Scribe

- Any Scribe node may create a group with a groupId.
- Node with nodeId numerically closest to groupId is the root of multicast tree.
- Group is formed by the union of the Pastry routes from each group member to the groupId's root.

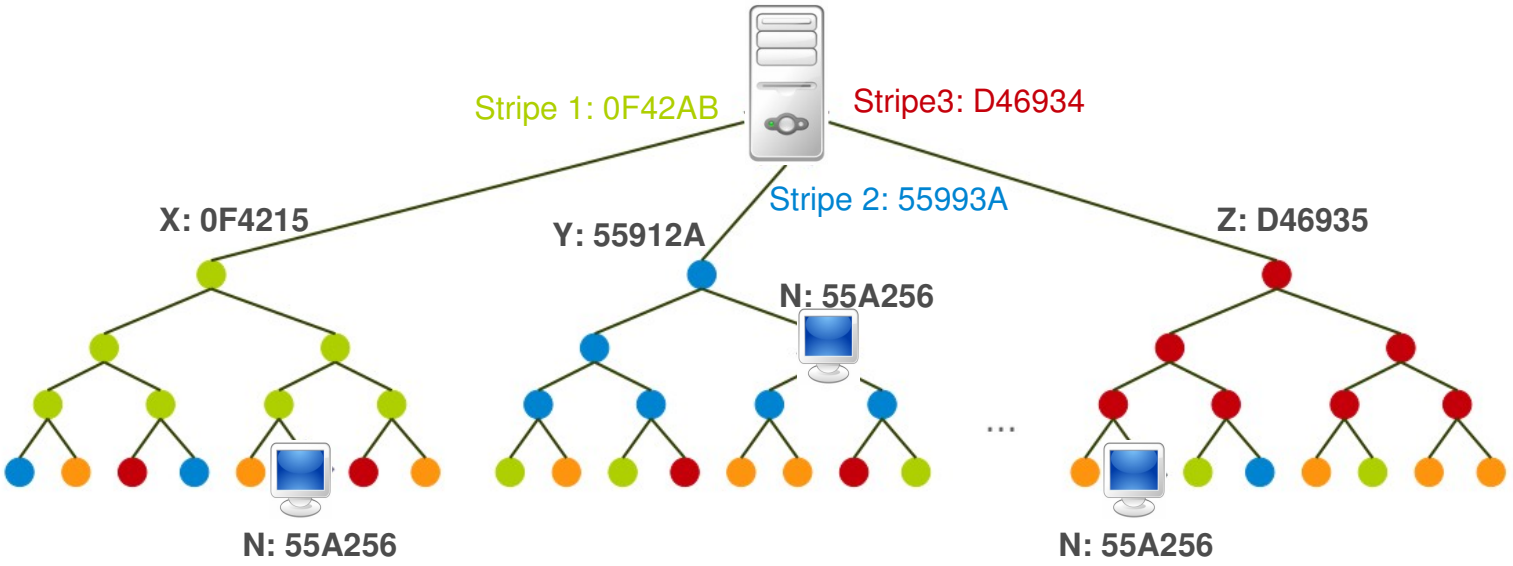


SplitStream Trees

- Create multiple trees, such that a node is interior node in at most one tree, and a leaf node in the other trees.
  - Interior-node-disjoint
- Each stripe is assigned a groupID.
  - The groupIDs differ in the most significant digit.
  - Creates one Scribe multicast tree for each stripe.
  - Prefix routing ensures the interior-node-disjoint property.

SplitStream Trees

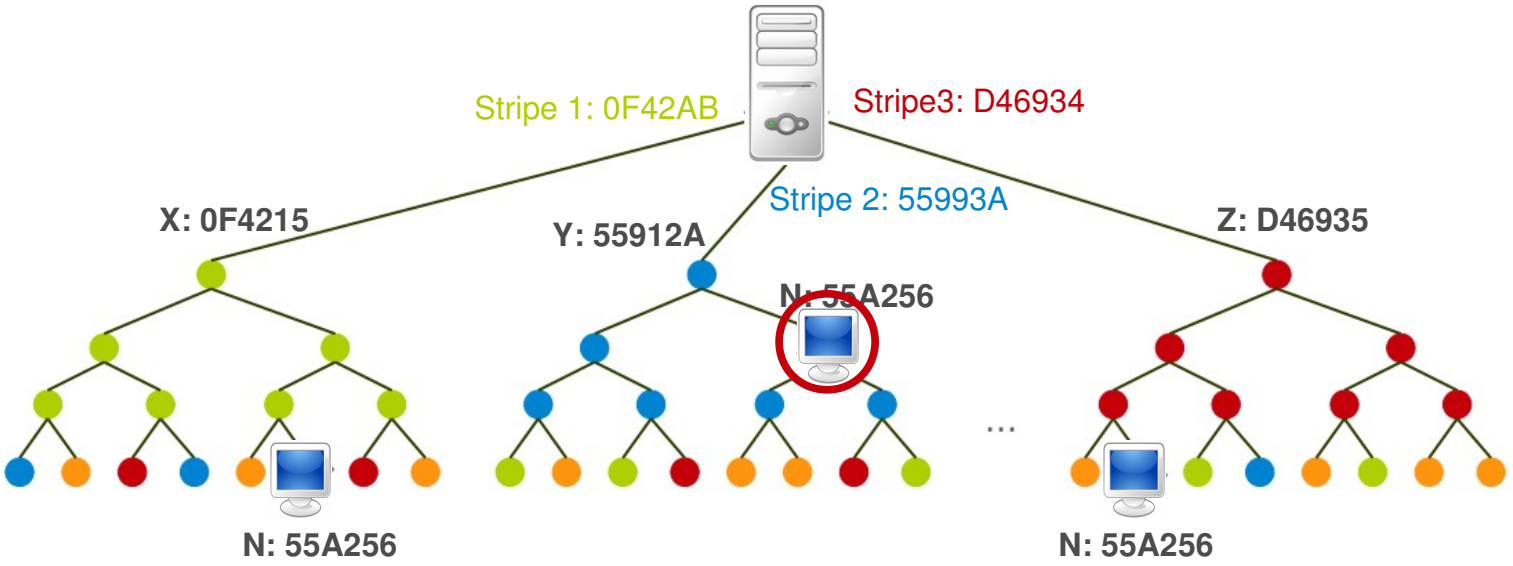
- For example:
  - Stripe1: groupID = 0F42AB
  - Stripe2: groupID = 55993A
  - Stripe3: groupID = D46934
- Node N is only internal in stripe tree 2.



SplitStream Trees

- For example:
  - Stripe1: groupID = 0F42AB
  - Stripe2: groupID = 55993A
  - Stripe3: groupID = D46934
- Node N is only internal in stripe tree 2.

What happens if N does not have enough upload bandwidth to support its children?

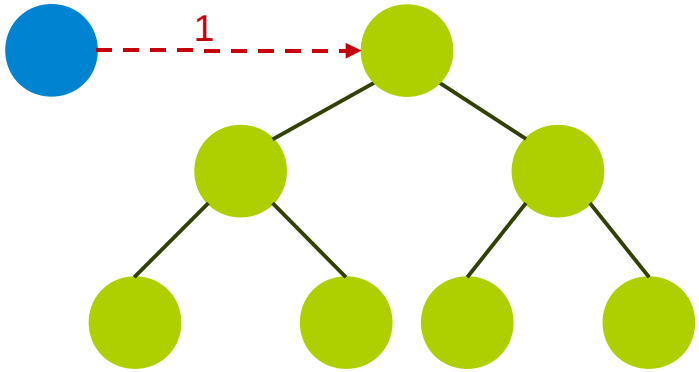


Scribe Solution for Bandwidth Problem

- Scribe has a built-in mechanism to limit a node's outdegree.
  - Push-down

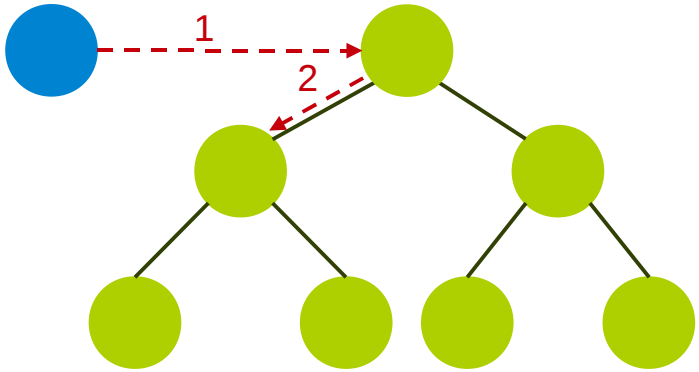
Push-Down (1/4)

- When a node that has reached its outdegree limit receives a join request:
  - It provides the new node a list of its current children.
  - The new node then seeks to be adopted by the child with lowest delay.
  - This procedure continues recursively down the tree until a node is found that take another child.



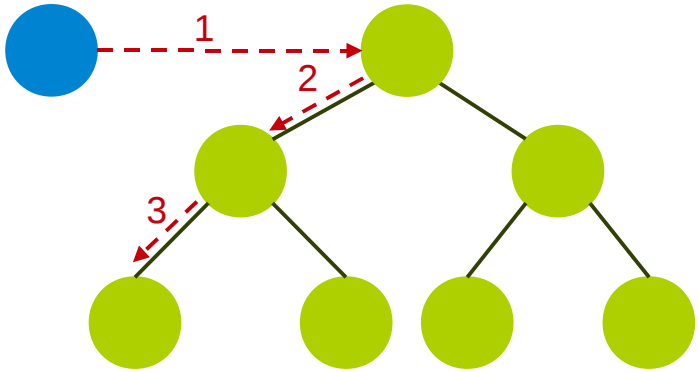
Push-Down (2/4)

- When a node that has reached its outdegree limit receives a join request:
  - It provides the new node a list of its current children.
  - The new node then seeks to be adopted by the child with lowest delay.
  - This procedure continues recursively down the tree until a node is found that take another child.



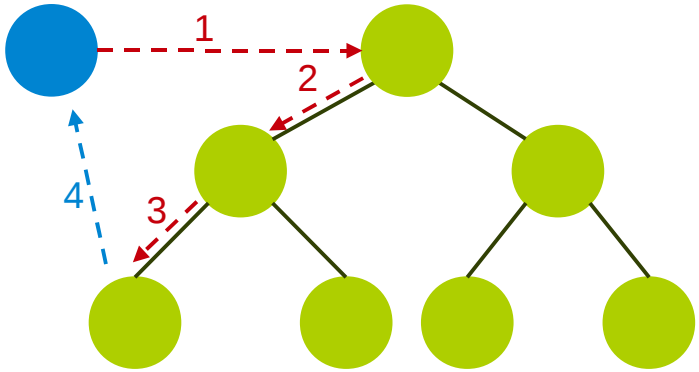
Push-Down (3/4)

- When a node that has reached its outdegree limit receives a join request:
  - It provides the new node a list of its current children.
  - The new node then seeks to be adopted by the child with lowest delay.
  - This procedure continues recursively down the tree until a node is found that take another child.



Push-Down (4/4)

- When a node that has reached its outdegree limit receives a join request:
  - It provides the new node a list of its current children.
  - The new node then seeks to be adopted by the child with lowest delay.
  - This procedure continues recursively down the tree until a node is found that take another child.

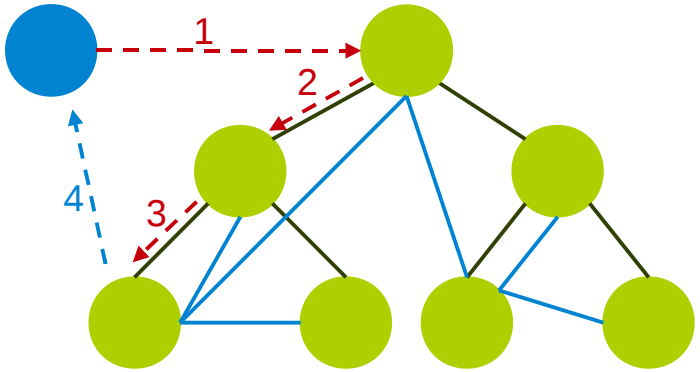


The Bandwidth Problem

- Does this procedure work in SplitStream? [d]

The Bandwidth Problem

- Does this procedure work in SplitStream? [d]
  - No!
  - A leaf node in one tree may be an interior node in another tree, and it may have already reached its outdegree limit with children in this other tree.



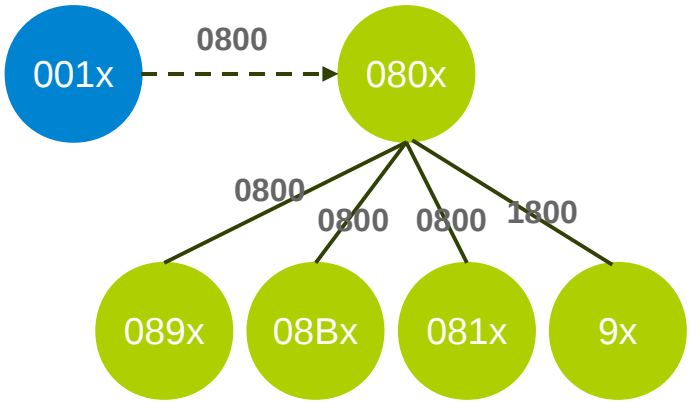
SplitStream Solution for Bandwidth Problem

- The algorithm for the case where a node that has reached its outdegree limit receives a join request:
- First, the node adopts the new child regardless of the outdegree limit.
- Then, it evaluates its new set of children to select a child to reject.
- Called locating parent.

Locating Parent (1/2)

- First, the node looks for children to reject in stripes whose stripeIds do not share a prefix with the local node's nodeId or has the shortest prefix match with that stripeId.

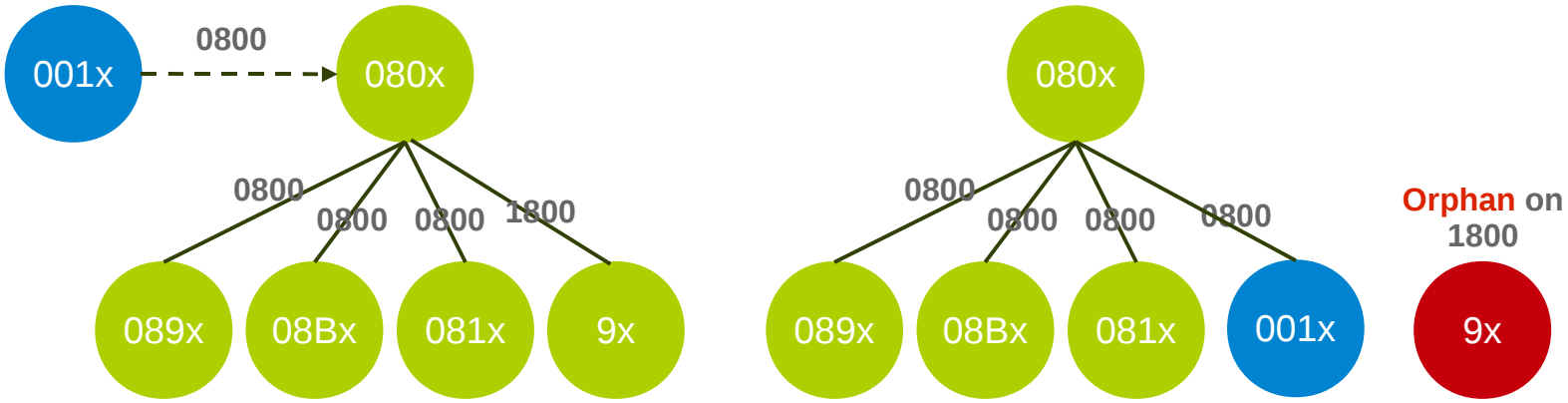
- If the new node is among them, it is selected; otherwise, one is chosen randomly from the set.



Locating Parent (2/2)

- First, the node looks for children to reject in stripes whose stripeIds do not share a prefix with the local node's nodeId or has the shortest prefix match with that stripeId.

- If the new node is among them, it is selected; otherwise, one is chosen randomly from the set.

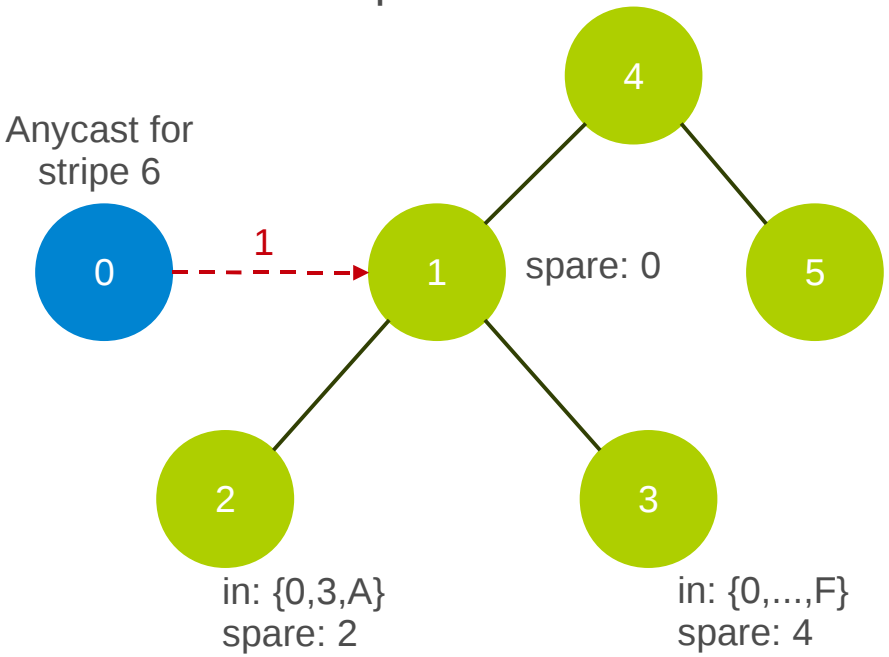


The Orphan Child

- Locate a parent amongst former siblings with the proper prefix.
  - Push-down
- Search the Spare Capacity Group.

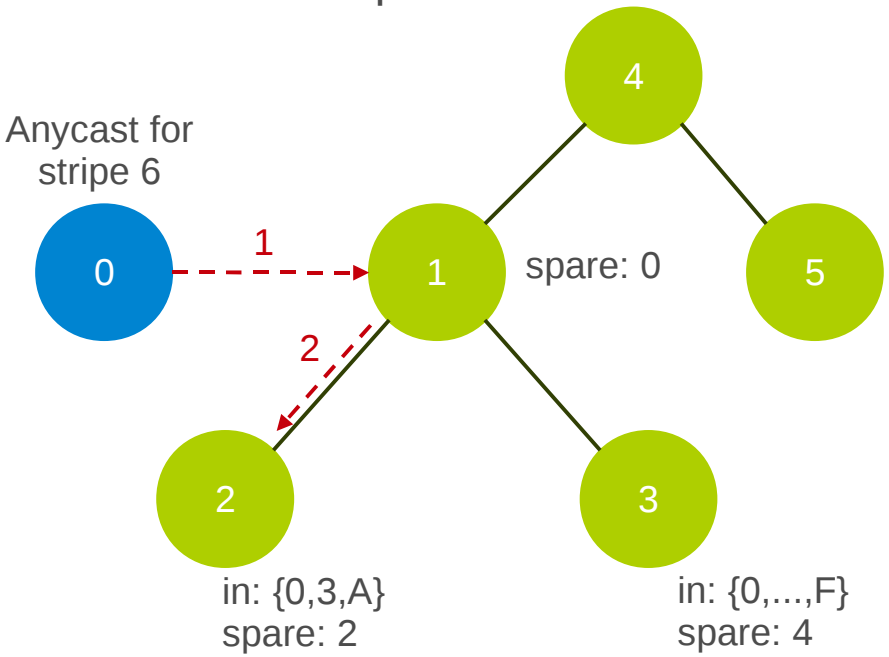
Spare Capacity Group (1/5)

- All nodes that have less children than their forwarding capacity limit.
- The orphan it sends an anycast message to the spare capacity group.
- Perform a depth-first search for a parent.



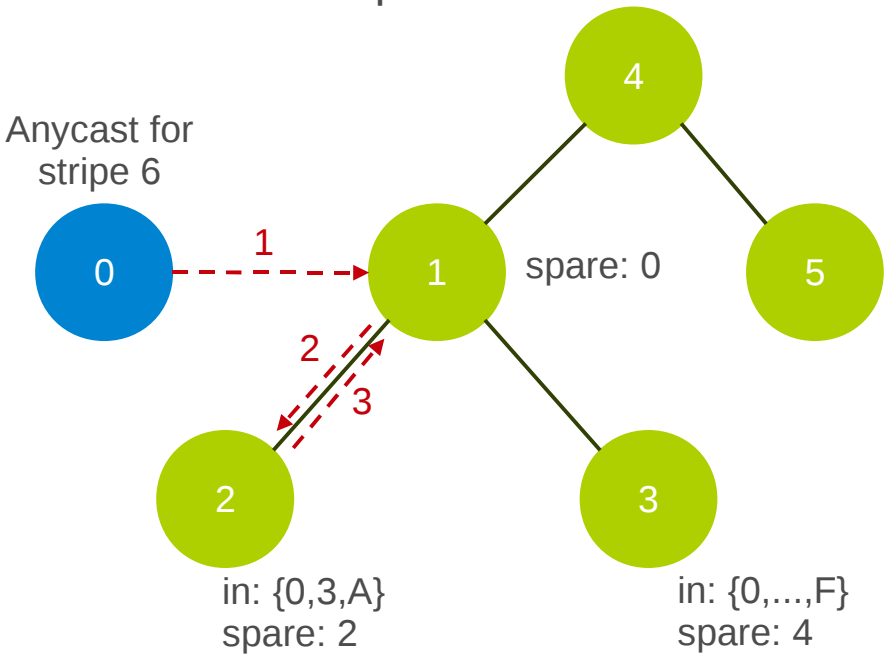
Spare Capacity Group (2/5)

- All nodes that have less children than their forwarding capacity limit.
- The orphan it sends an anycast message to the spare capacity group.
- Perform a depth-first search for a parent.



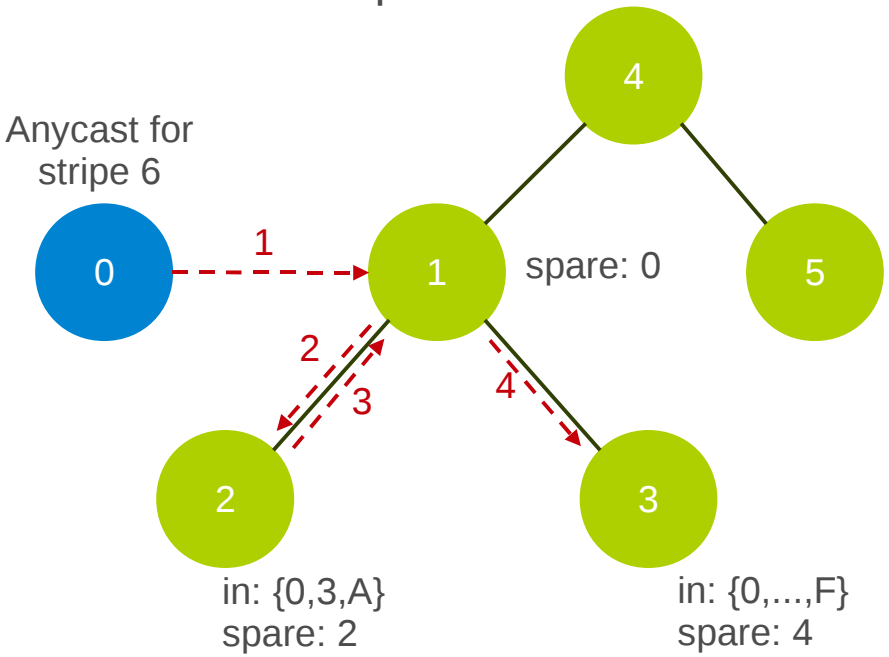
Spare Capacity Group (3/5)

- All nodes that have less children than their forwarding capacity limit.
- The orphan it sends an anycast message to the spare capacity group.
- Perform a depth-first search for a parent.



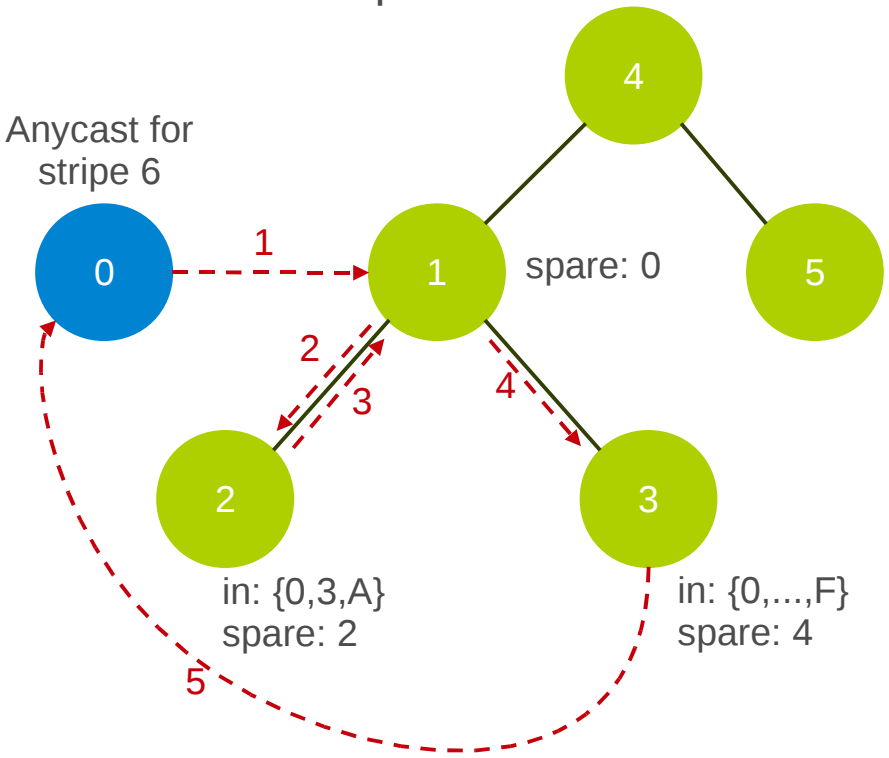
Spare Capacity Group (4/5)

- All nodes that have less children than their forwarding capacity limit.
- The orphan it sends an anycast message to the spare capacity group.
- Perform a depth-first search for a parent.



Spare Capacity Group (5/5)

- All nodes that have less children than their forwarding capacity limit.
- The orphan it sends an anycast message to the spare capacity group.
- Perform a depth-first search for a parent.



SplitStream Summary

- Multiple trees
  - Interior-node-disjoint
- Each stripe is assigned a groupID.
  - The groupIDs differ in the most significant digit.
- Bandwidth problem
  - Push-down solution (scribe)
    - Not enough for splitstream
  - Locating parents
  - Spare capacity group
    - Used by orphan nodes to rejoin

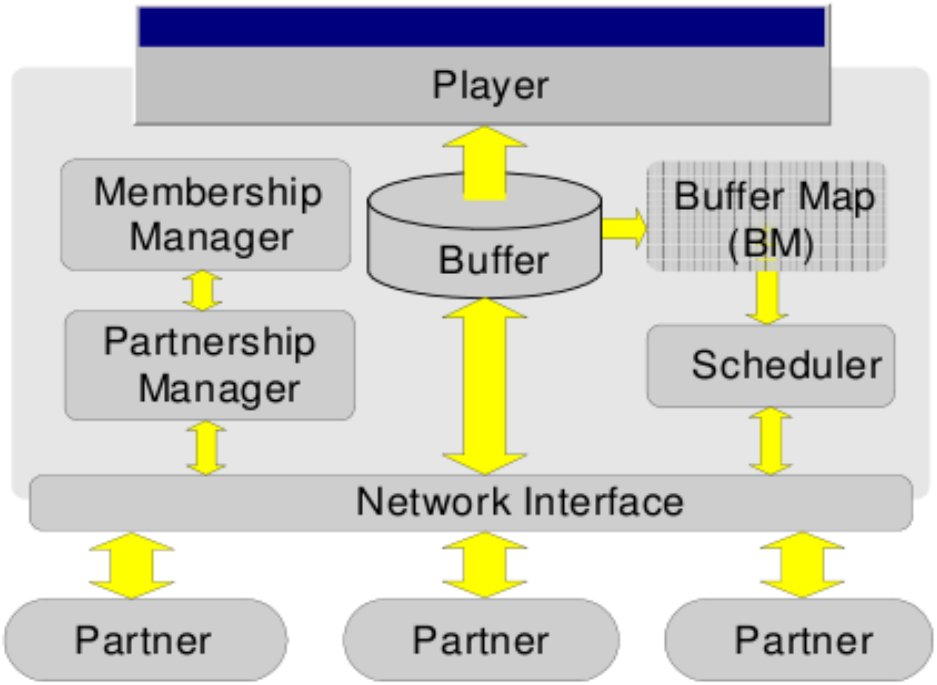
DONet/CoolStreaming

(Gossip-based – Mesh)

DONet/Coolstreaming

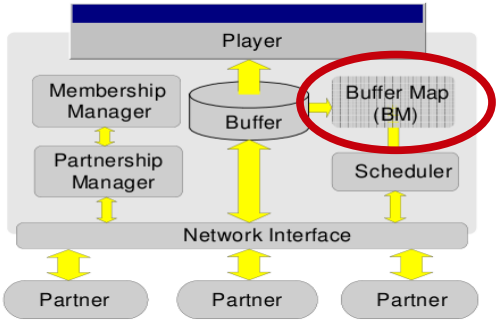
- Uses gossip algorithm to disseminate messages.
- The media stream is divided into blocks or segments.
- For each segment, a node can be receiver or supplier.
- The source node is always supplier.

Node System Diagram



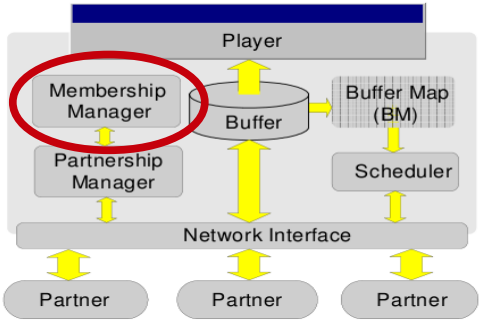
Buffer Map

- Shows the availability of the segments in the buffer of a node.
- Each node continuously exchange its BM with its partners.



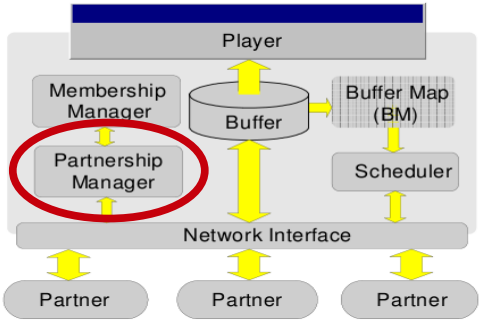
Membership Management

- Each node has a partial list of the ID for the active nodes.
  - mCache
- A node uses a peer sampling service to update its mCache.
  - Coolstreaming uses SCAMP.



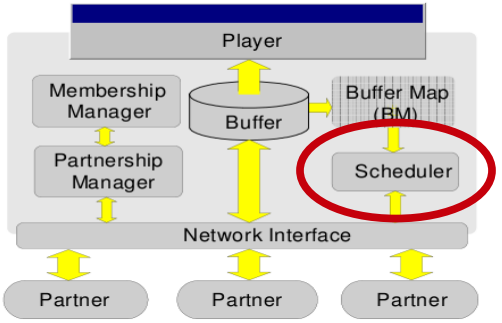
Partnership Management

- Each node periodically exchanges their BM with its partners in mCache.
- A node retrieves unavailable data from one or more partners, or supplies available data to partners.



Scheduler

- For a homogeneous and static network a simple round-robin scheduler may work well.



Scheduler

- For a heterogeneous and dynamic network.
- Two constraints:
  - The playback deadline for each segment.
  - The heterogeneous streaming bandwidth from the partners.
- If the first constraint cannot be satisfied, then the number of segments missing deadlines should be kept minimum.

Scheduler

- First calculates the number of potential suppliers for each segment.
- A segment with less potential suppliers is more difficult to meet the deadline constraints.
  - Starting from those with only one potential supplier, then those with two, and so forth.
- Among the multiple potential suppliers, the one with the highest bandwidth.

DONE!

A Page To Remember

- Media Streaming
  - Live
  - VoD
- Client-Server model
  - Expensive

- P2P model
  - The peers can help each other and the capacity increases with the number of peers
- Challenges
  - Bandwidth
  - Time constraint
  - Churn
- Two questions
  - What overlay topology
  - How to construct the topology

Question?