Ad-Hoc Wireless Networks: Architectures and Protocols
Ch. 8.6: Tree-Based Multicast Routing Protocols

C. Siva Ram Murthy, B. S. Manoj

Antti Hyvärinen

April 4., 2007

Practical Significance

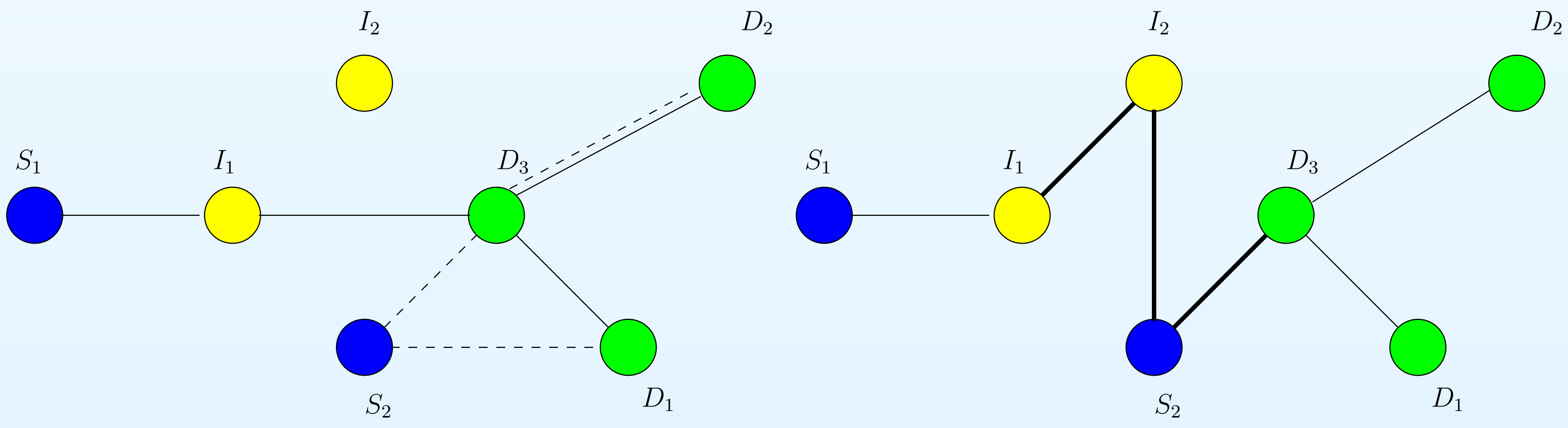
- Multicast trees in ad-hoc networks emerge naturally from domains where
  - Group of collaborators move in a new environment
  - Collaboration is directed by one or more coordinators
- Such environments include
  - Search and rescue missions
  - Military campaigns
  - Law enforcement
  - Classrooms
  - Conferences
- Requirements vary, for example QoS
  - Military
  - Multimedia

Outline

- Cover the main types of multicast trees
  - source-tree-based
  - shared-tree-based
- Cover the main design criteria
  - Optimize for memory
  - Optimize for bandwidth
  - Optimize for robustness
- Describe representative examples of tree-based multicast protocols

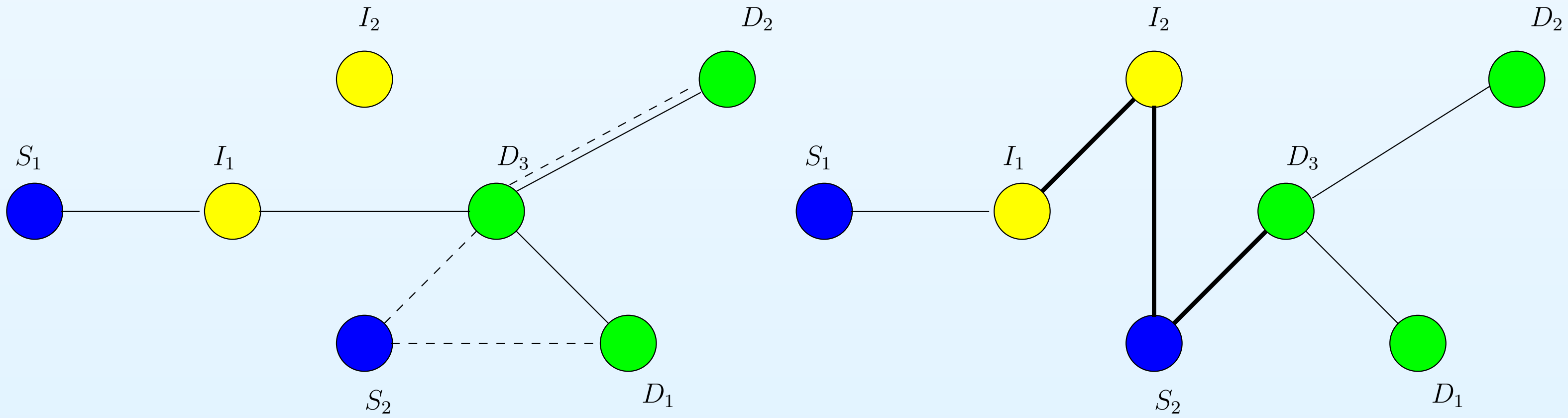
Source-tree-based and shared-tree-based

- Source-tree-based: tree rooted at the source
  - Performs well at heavy loads, due to efficient traffic distribution
- shared-tree-based: tree rooted at a rendez-vous point
  - Scales well for multiple sources
  - Tree links get overloaded with traffic
  - Heavy dependence on the shared tree nodes



Memory, Bandwidth, Energy and Robustness

- Memory: each node having routing information infeasible in large networks
- Bandwidth: certain protocols overload some network connections causing slowdown
- Energy: certain use excessive amounts of intermediate nodes causing unreliable multi-hop links and battery usage
- Added robustness: (e.g. link count) usually increases administrative overhead



General: Building the multicast tree

Tree construction performed by

- source or
- destination
- A new destination might be able to join after tree creation

Route discovery by

- Flooding the full network or
- Sending to neighbors only
- Use caution to avoid loops

Using possibly some existing infrastructure

General: Recovering from link failure

Link failure identified by

- Periodic querying by beacons (proactive)
- Timeouts (reactive)
- RTS/CTS information (hw-assisted)

Link recovery initiated by

- The destination
- The upstream node

Tree-based Routing Protocols: Examples

- Multicast Zone Routing Protocol
- Multicast Core-Extraction Distributed Ad-Hoc Routing
- Differential Destination Multicast Routing Protocol
- Weight-Based Multicast Protocol
- Preferred Link-Based Multicast Protocol
- Ad-Hoc Multicast Routing Protocol

Not covered in this presentation are Bandwidth-Efficient Multicast Routing Protocol, Associativity-Based Ad-Hoc Multicast Routing, Multicast Ad–Hoc On-Demand Distance Vector Routing, Ad-Hoc Multicast Routing Using Increased ID-Numbers and Adaptive Shared-Tree Multicast Routing

Multicast Zone Routing Protocol

- Shared-tree source-initiated
- Each node is associated with a zone
  - Inside the zone, node knows the topology
  - Outside the zone, let border nodes do routing
- Source constructs a tree in two phases

A_1 Send TREE-CREATE to all nodes in the zone

A_2 Willing nodes reply with TREE-CREATE-ACK

B_1 Send TREE-PROPAGATE to all nodes

B_2 Border nodes send TREE-CREATE to respective zones

MZRP (cont'd)

- Source maintains tree by periodic TREE-REFRESH.
  - If a node in the tree does not receive TREE-REFRESH, it removes the stale multicast
- Receiver R disconnected because of failing intermediate node I
  - R sends TREE-JOIN to the zone and connects
  - R sends JOIN-PROPAGATE to border nodes

Multicast Core-Extraction Distributed Ad-Hoc Routing

Assume there is an underlying mesh of core nodes which form a minimum dominating set for all nodes.

- The mesh, called *mgraph*, is used as a robust infrastructure for forwarding data
- Resulting multicast tree is a source-tree
- Core nodes
  - are selected by election approximating the NP-complete problem
  - have complete knowledge on dominated neighbors
  - know the nearest core nodes in three-hop radius
- Multicast is based on reliable unicast

MCEDAR (cont'd)

- A new node C sends *JoinReq* to its dominator
  - Loops are prevented by a decreasing identity in *JoinReq*
  - A tree-node replies with *JoinAck* containing tree-node's identity
  - The new node C accepts multiple *JoinAck*'s depending on the *robustness factor*
- Node C might have downstream core nodes, for which communication is forwarded
- Link quality is measured and bad quality is propagated faster than good quality

The protocol uses redundant links, combining the strengths of tree-based and mesh-based protocols

Differential Destination Multicast Routing Protocol

- Source nodes manage the multicast group membership
  - Destinations join the source by unicast
  - Source piggy-backs queries periodically to refresh list of destinations
- Each node independently decides to operate in
  - Stateless mode or
  - Soft-state mode
- explained in the following slide

DDM states (cont'd)

- Stateless mode
  - The route of the packet is coded in the data
  - No need for complicated routing
  - In large networks, overhead is high
- Soft-state mode
  - Every node may cache the routing information
  - The protocol needs not to list all destinations in every data packet
  - When routes change, upstream node sends a *difference* to the destinations
  - Significantly reduces the overhead

Weight-Based Multicast

Joining node R minimizes the cost to source by considering

- Number of required intermediate nodes
- Distance of R from source

The joining is done by flooding a *JoinReq* with a TTL

- Tree nodes reply with a distance from source
- Distance is increased by each hop to the joining node
- Collect several of alternate routes
- The objective is to minimize function

$$Q = (1 - joinWeight) \times (n_h - 1) + joinWeight \times (n_h + n_s)$$

- n_h is the hop distance from joining node to tree node
- n_s is the hop distance from tree node to source
- $0 \leq joinWeight \leq 1$

WBM (cont'd)

To maintain the tree with high packet delivery capability, link failures are predicted in the following way

- Neighbor nodes listen to communication in promiscuous mode
- When receiving a packet piggy-backed with *TriggerHandoff*, and
    - node has information on the multicast tree, and
    - Distance from the tree is less than the node's requesting handoff,
- the node sends *HandoffConf* to requesting node

Requesting node selects the node nearest to the tree. If the handoff fails, rejoin.

Preferred Link-Based Multicast

A receiver-initiated protocol with local and tree-level topology, limiting forwarding nodes to preferred ones

- Neighbor-Neighbor Table (NNT)
  - A list of neighbors of the node in two steps
  - Used for quick repair of broken links
- Connect Table (CT)
  - Tree information

PLBM (cont'd)

- No flooding of network required if NNT contains tree node
  - Each node periodically sends a beacon with TTL
  - Nearby nodes will know of a tree node by the beacon
- Otherwise, use algorithm to determine candidates for connection
  - List potential nodes in the flood-packet
  - Only listed nodes will respond to flooding (by forwarding or replying)
  - Hopefully several good candidates, which receiver can choose from

Ad-Hoc Multicast Routing

A robust algorithm for high-mobility environment using underlying mesh

- Based on underlying IP network using IP unicast
- Builds a higher-level IP network tunnelled over the lower level IP
- Network is divided to groups having
  - At least one logical core
    - Selected by an election in case of multiple cores
    - Can thus change dynamically
  - zero or more normal nodes
- Network is periodically flooded by CREATE-TREE messages from cores

AMRoute (cont'd)

- Removing multiple cores
  - Different segments may be joined by new nodes
  - Two-core segment can be noticed by multiple tree creation messages
  - In this case, a distributed core election algorithm is run by all nodes
- Adding a new core
  - A disappeared core because of movement
  - In a no-core segment, one of the nodes will announce itself as a core after a random period

Summary

- Several different approaches exits for tree-based Ad-Hoc routing protocols
  - Mesh-assisted
  - Source-initiated
  - Destination-initiated
  - Stateless
  - Semi-stateless
  - And others
- Most reports on the use seem to be simulated
  - The applicability of the methods are not known on practical domains, such as rescue missions, military campaigns and other domains