Routing Protocols

- Setting all routes in large size networks is an unfeasible solution
- There exists several routing algorithm developed for Linux-based OSs (BGP, OSPF, OLSR, AODV, etc.)
- In this seminary we will focus on routing protocol for wireless multi-hop networks (specifically for MANETs and WMNs)
  - AODV: reactive routing protocol
  - OLSR: proactive routing protocol

Routing Protocols

- Reactive Protocols (AODV)
  - Generally involve large delays between the request and first packet delivery
  - Incur low overhead in low traffic scenarios
- Proactive Protocols (OLSR)
  - Packets are immediately delivered as paths are already established
  - Results in high path maintenance overhead since the paths are kept regardless of traffic Patterns
- Hybrid Protocols
  - Operate midway of delay and overhead performance

Trade-Off

- Latency of route discovery
  - Proactive protocols may have lower latency since routes are maintained at all times
  - Reactive protocols may have higher latency because a route from X to Y will be found only when X attempts to send to Y
- Overhead of route discovery/maintenance
  - Reactive protocols may have lower overhead since routes are determined only if needed
  - Proactive protocols can (but not necessarily) result in higher overhead due to continuous route updating
- Which approach achieves a better trade-off depends on the traffic and mobility patterns

Reactive Routing Protocols

Route Discovery

- Reactive Routing Protocols discover reactively the route towards a destination D
  - When a packet needs to be sent to D and the route to D is unknown
- When node S wants to send a packet to node D, but does not know a route to D, node S initiates a **route discovery**
  - Source node S floods Route Request (RREQ)
- Destination D on receiving the first RREQ, sends back a Route Reply (RREP)

Route Discovery

- The route used to forward data packets is
  - stored in the packet header (DSR)
  - derived dinamically by using routing tables at each node (AODV)
- Storing routes in packet header results in a large overhead (particularly when data contents of a packet are small)
- AODV improves DSR by maintaining routing tables at the nodes
- Using AODV, routes are maintained only between nodes which need to communicate

Ad Hoc On-Demand Distance Vector Routing (AODV)

- When a source node S desires a route to a destination D for which it does not already have a route, it broadcasts a route request (RREQ) packet across the network
- When a node re-broadcasts a RREQ, it sets up a reverse path pointing towards the source S
  - AODV assumes symmetric (bi-directional) links
- When the intended destination receives a Route Request, it replies by sending a Route Reply (RREP)
- RREP travels along the reverse path set-up when Route Request is forwarded

Route Request (RREQ)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |J|R|G|D|U|   Reserved          |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            RREQ ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination IP Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Destination Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Originator IP Address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Originator Sequence Number                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Route Request (RREQ)

Field	Size (bit)	Description
Type	8	Message Identifier (1 = RREQ)
Flags	5	See Flags description (J, R, G, D, U)
Reserved	11	Ignored
Hop Count	8	The number of hops from the Originator IP Address to the node handling the request
RREQ ID	32	A sequence number uniquely identifying the particular RREQ
Destination IP	32	The IP address of the destination for which a route is desired
Dst Seq Number	32	The latest sequence number received in the past by the originator for any route towards the destination
Originator IP Address	32	The IP address of the node which originated the Route Request
Originator Sequence Number	32	The current sequence number to be used in the route entry pointing towards the originator of the route rquest

RREQ Flags (5)

- **J**: **Join** flag; reserved for multicast.
- **R**: **Repair** flag; reserved for multicast.
- **G**: **Gratuitous** RREP flag; indicates whether a gratuitous RREP should be unicast to the node specified in the Destination IP Address field
- **D**: **Destination** only flag; indicates only the destination may respond to this RREQ
- **U**: **Unknown** sequence number; indicates the destination sequence number is unknown

Route Reply (RREP)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |R|A|    Reserved     |Prefix Sz|   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Destination IP address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Destination Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Originator IP address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Lifetime                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Route Reply (RREP)

Field	Size (bit)	Description
Type	8	Message Identifier (2 = RREP)
Flags	2	See Flags description (R, A)
Reserved	9	Ignored
Prefix Sz	5	refix Size specifies that the indicated next hop may be used for any nodes with the same routing prefix
Hop Count	8	The number of hops from the Originator IP Address to the Destination IP Address
Destination IP Address	32	The IP address of the destination for which a route is supplied
Dst Seq Number	32	The destination sequence number associated to the route
Originator IP Address	32	The IP address of the node which originated the RREQ for which the route is supplied
Lifetime	32	The time in milliseconds for which nodes receiving the RREP consider the route to be valid

RRKEP Flags (2)

- Flags of Route Replay:
  - **R**: **Repair** flag; used for multicast.
  - **A**: **Acknowledgment** required.

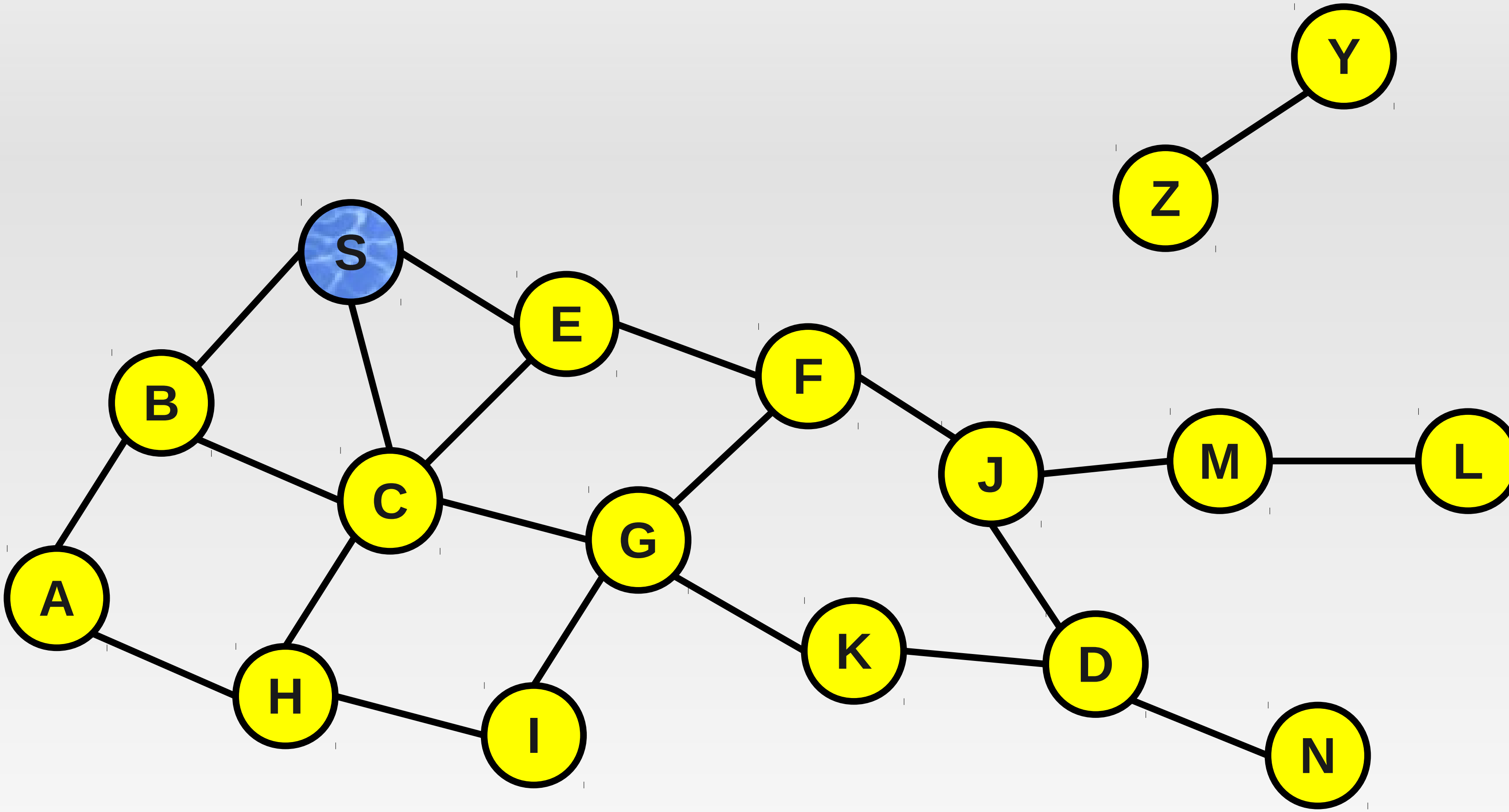
Route Error (RERR)

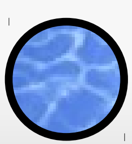
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |N|          Reserved           |   DestCount   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Unreachable Destination IP Address (1)             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Unreachable Destination Sequence Number (1)           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
|  Additional Unreachable Destination IP Addresses (if needed)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Additional Unreachable Destination Sequence Numbers (if needed)|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Route Error (RREP)

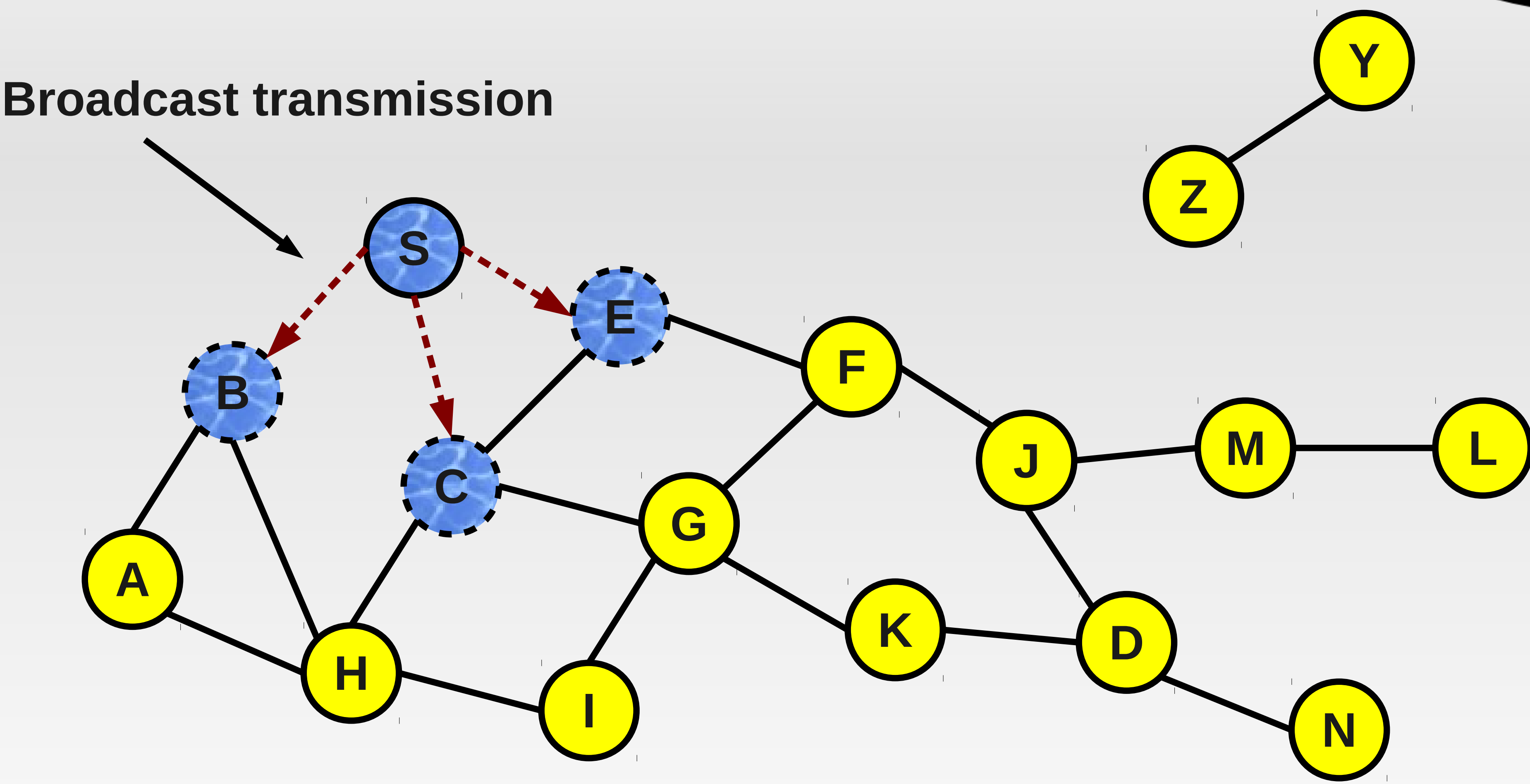
Field	Size (bit)	Description
Type	8	Message Identifier (3 = RERR)
N flag	1	No delete flag; set when a node has performed a local repair of a link
Reserved	15	Ignored
DestCount	8	The number of unreachable destinations included in the message; MUST be at least 1
Unreachable Destination IP	32	The IP address of the destination that has become unreachable due to a link break
Unreachable Dst Seq Number	32	The sequence number in the route table entry for the destination listed in the previous Unreachable Destination IP Address field

Route Requests in AODV



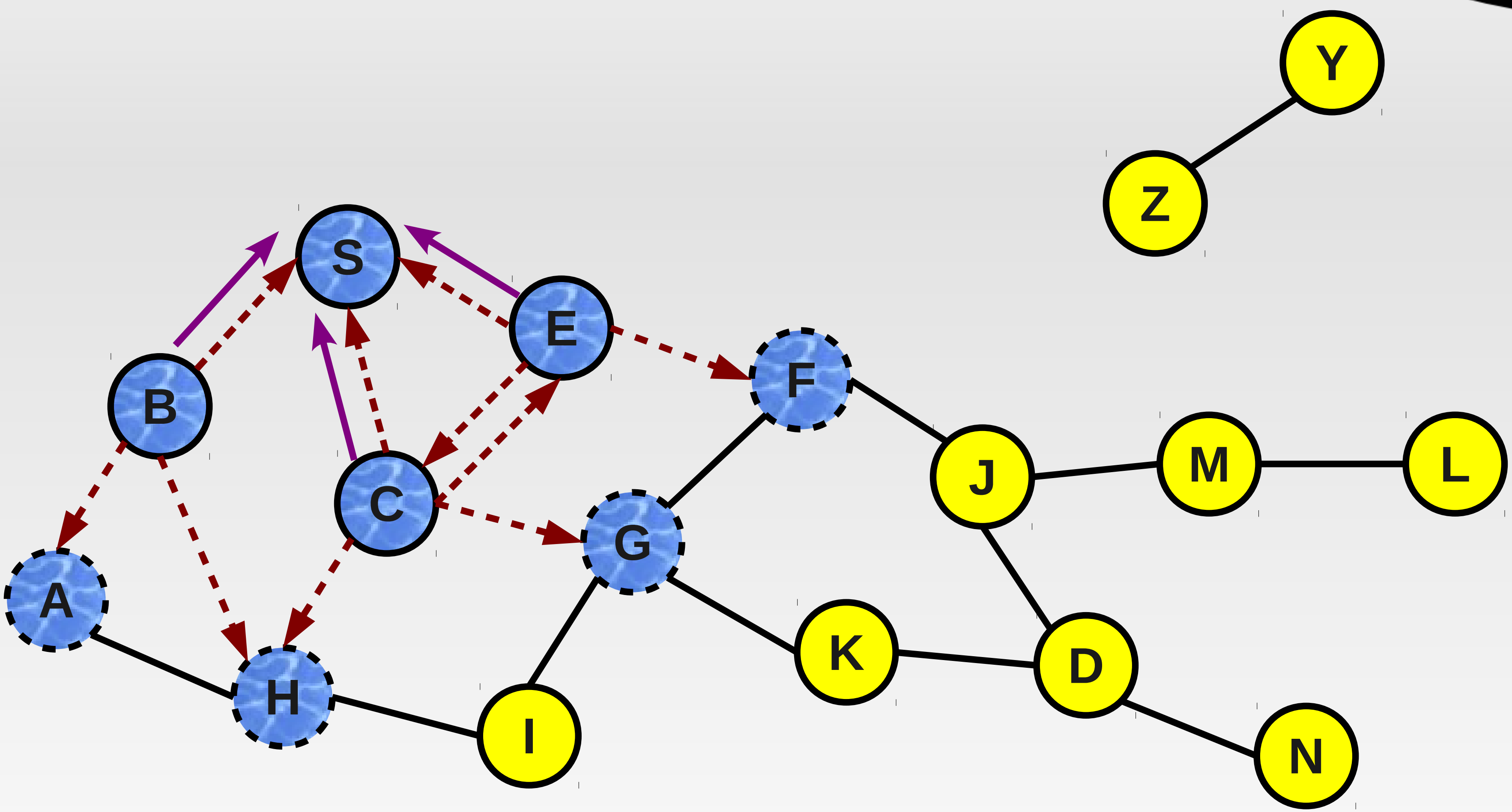

Represents a node that has received RREQ for D from S

Route Requests in AODV



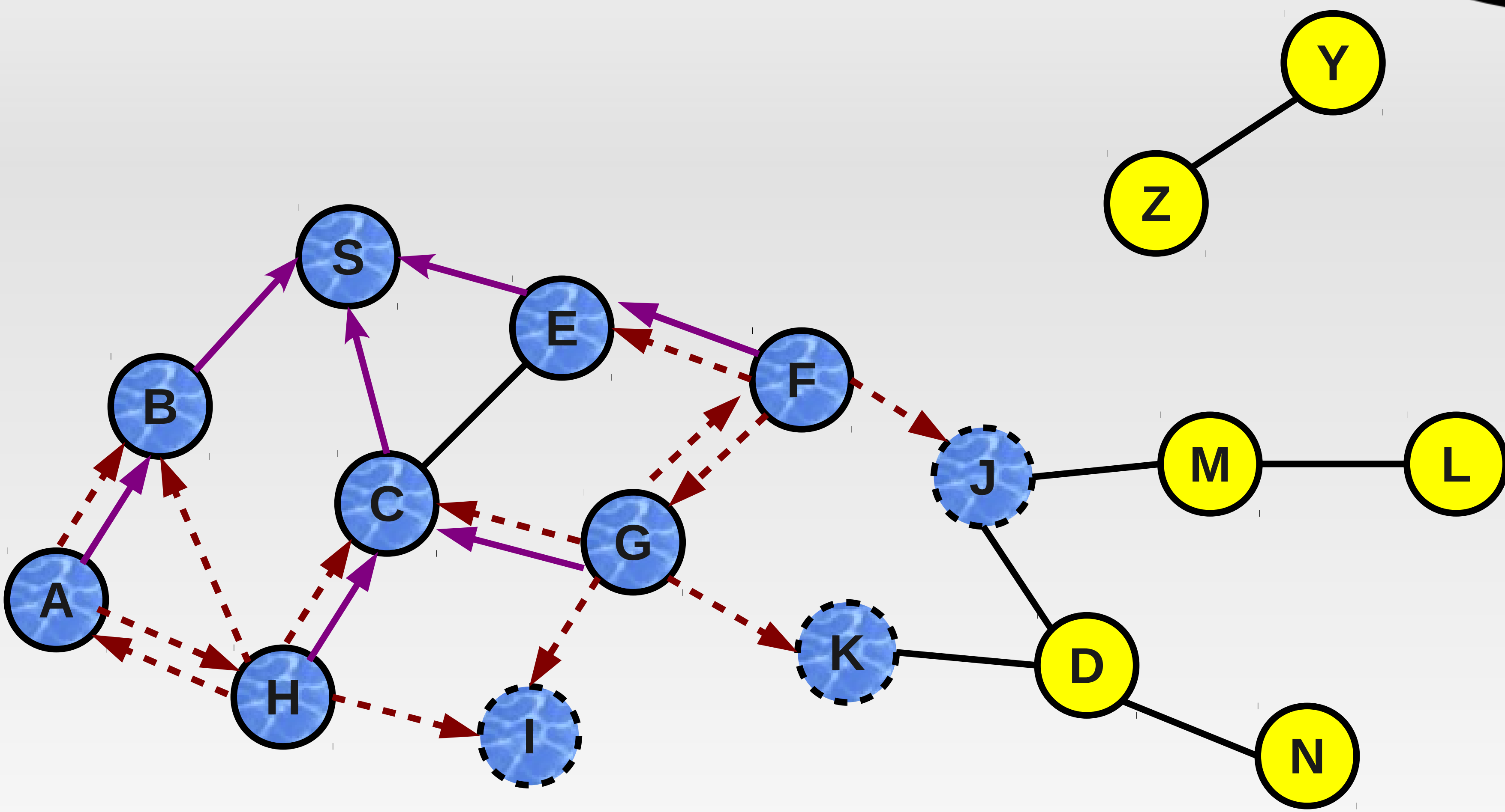
Represents transmission of RREQ

Route Requests in AODV



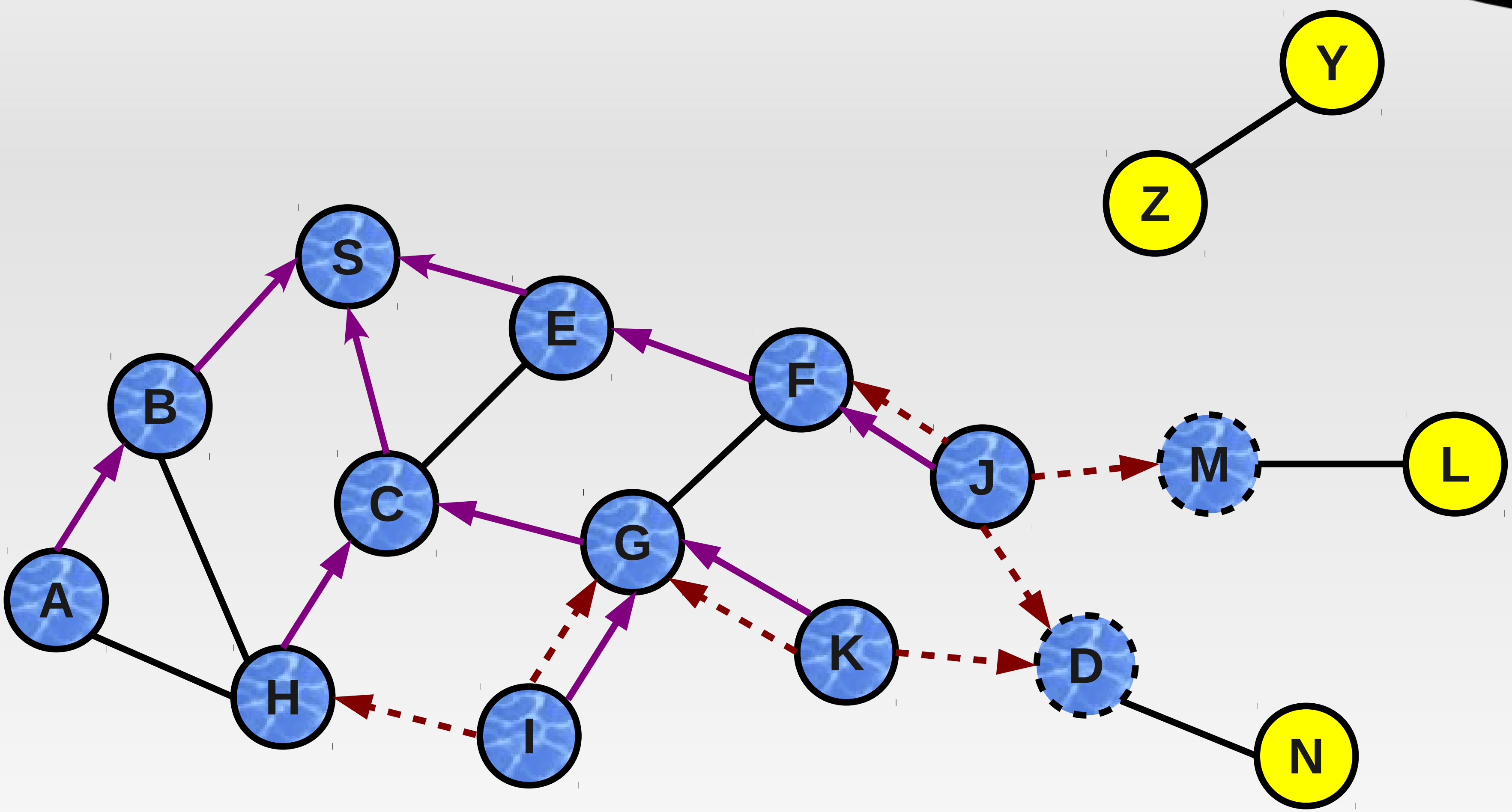
Represents links on Reverse Path

Route Requests in AODV



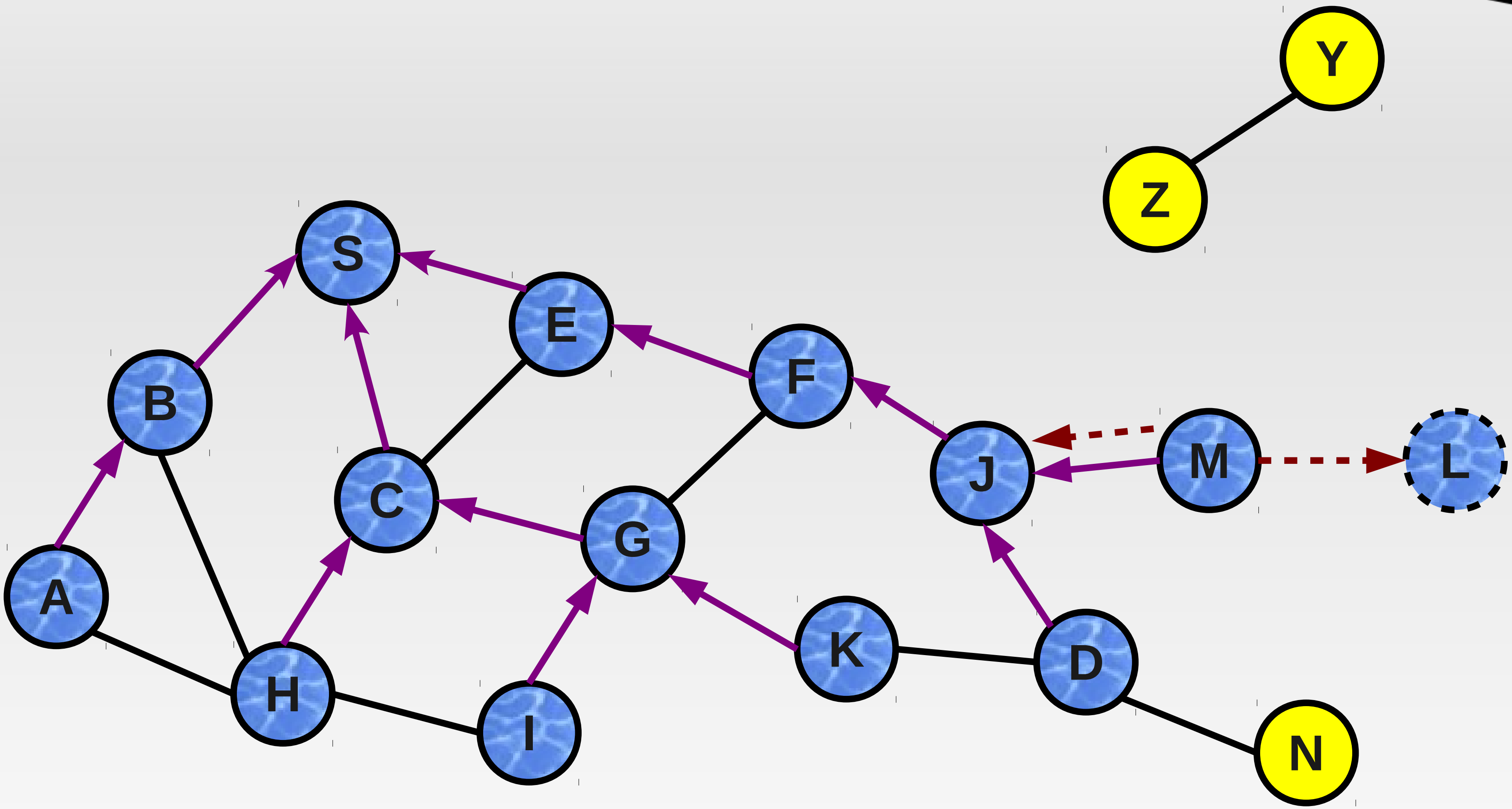
Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once

Route Requests in AODV



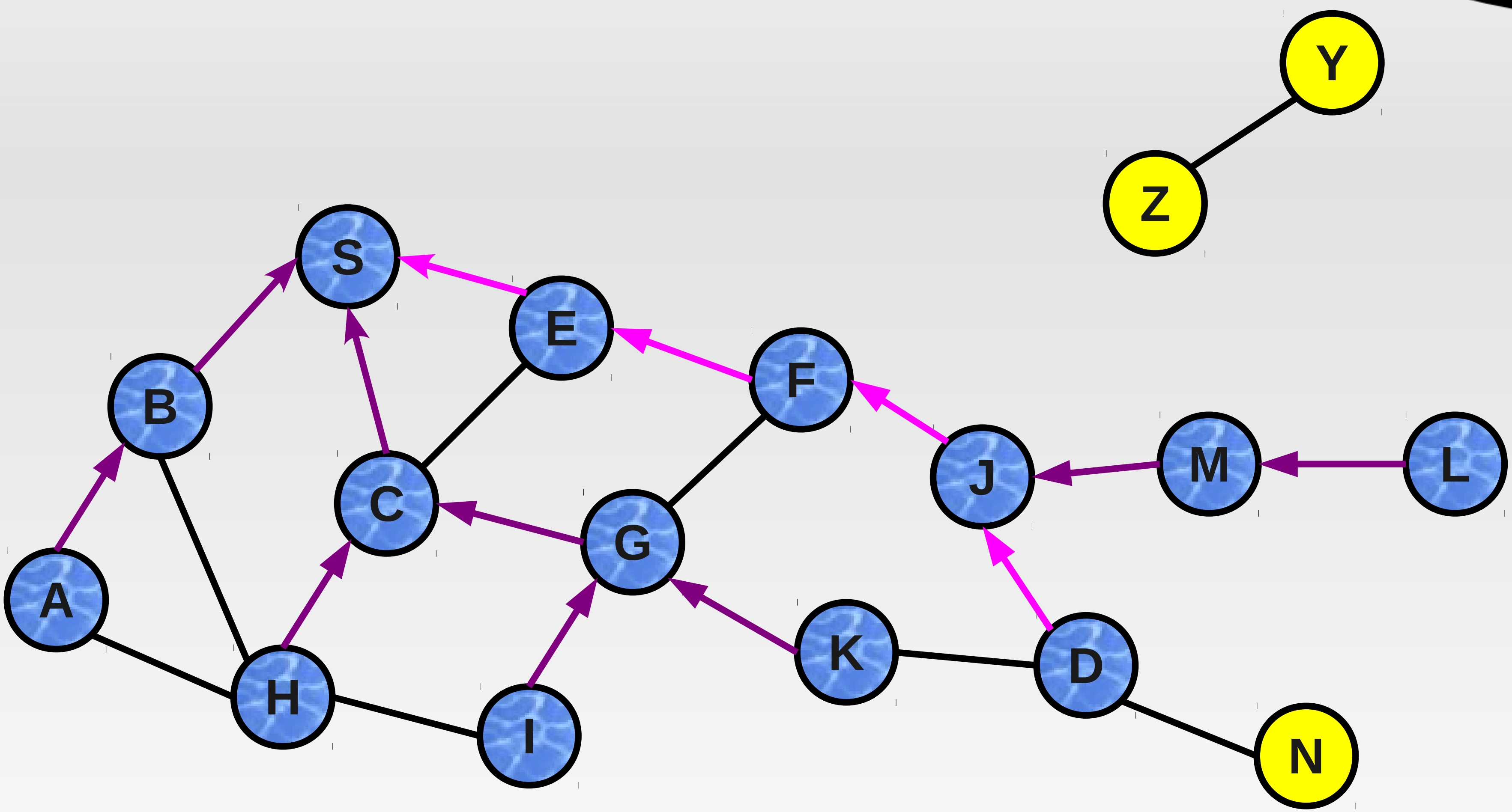
ubuntu

Route Replies in AODV



Node D **does not forward RREQ**, because node D is the **intended target** of the RREQ

Route Replies in AODV

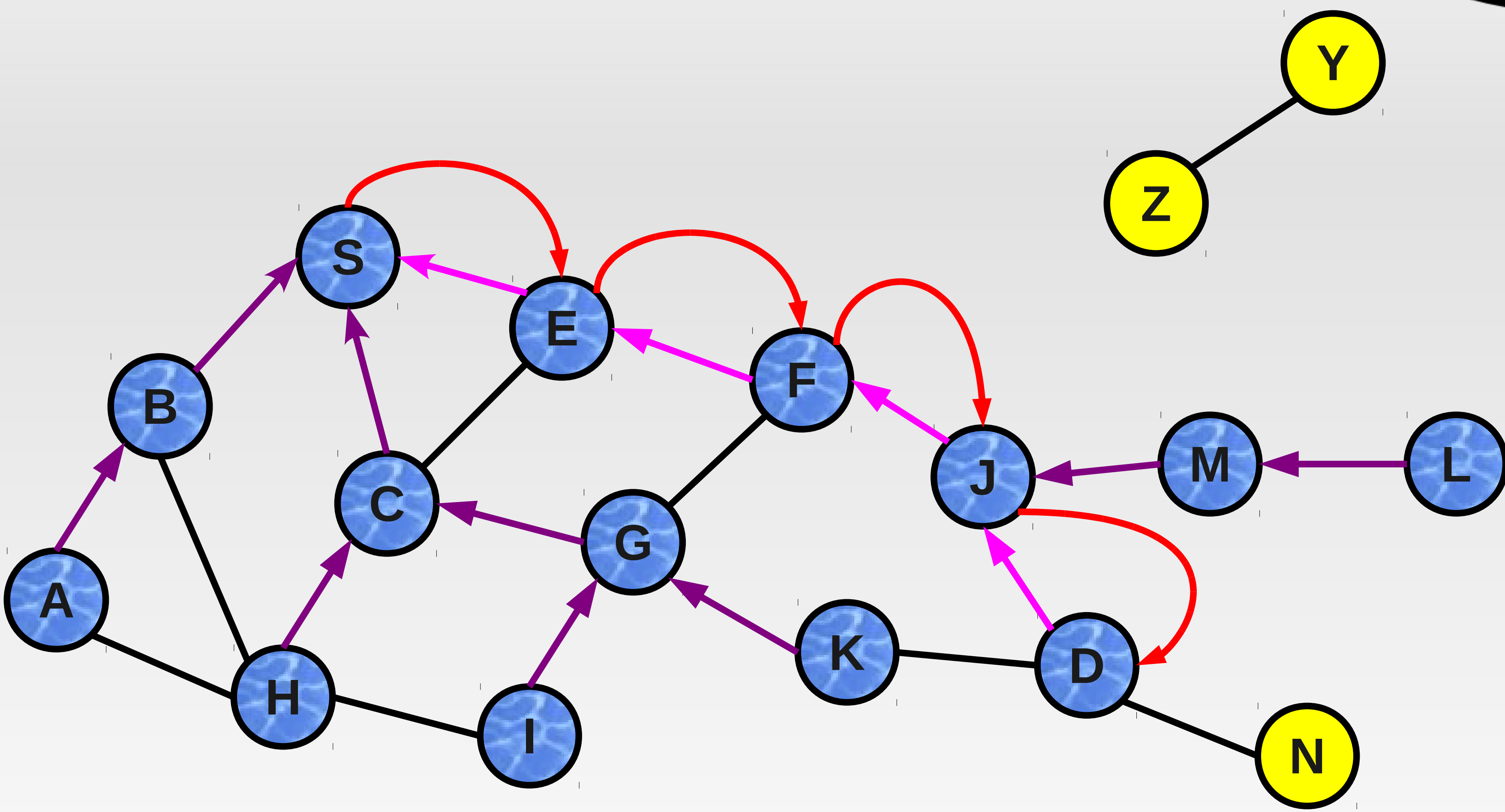


Represents links on path taken by RREP

Route Reply in AODV

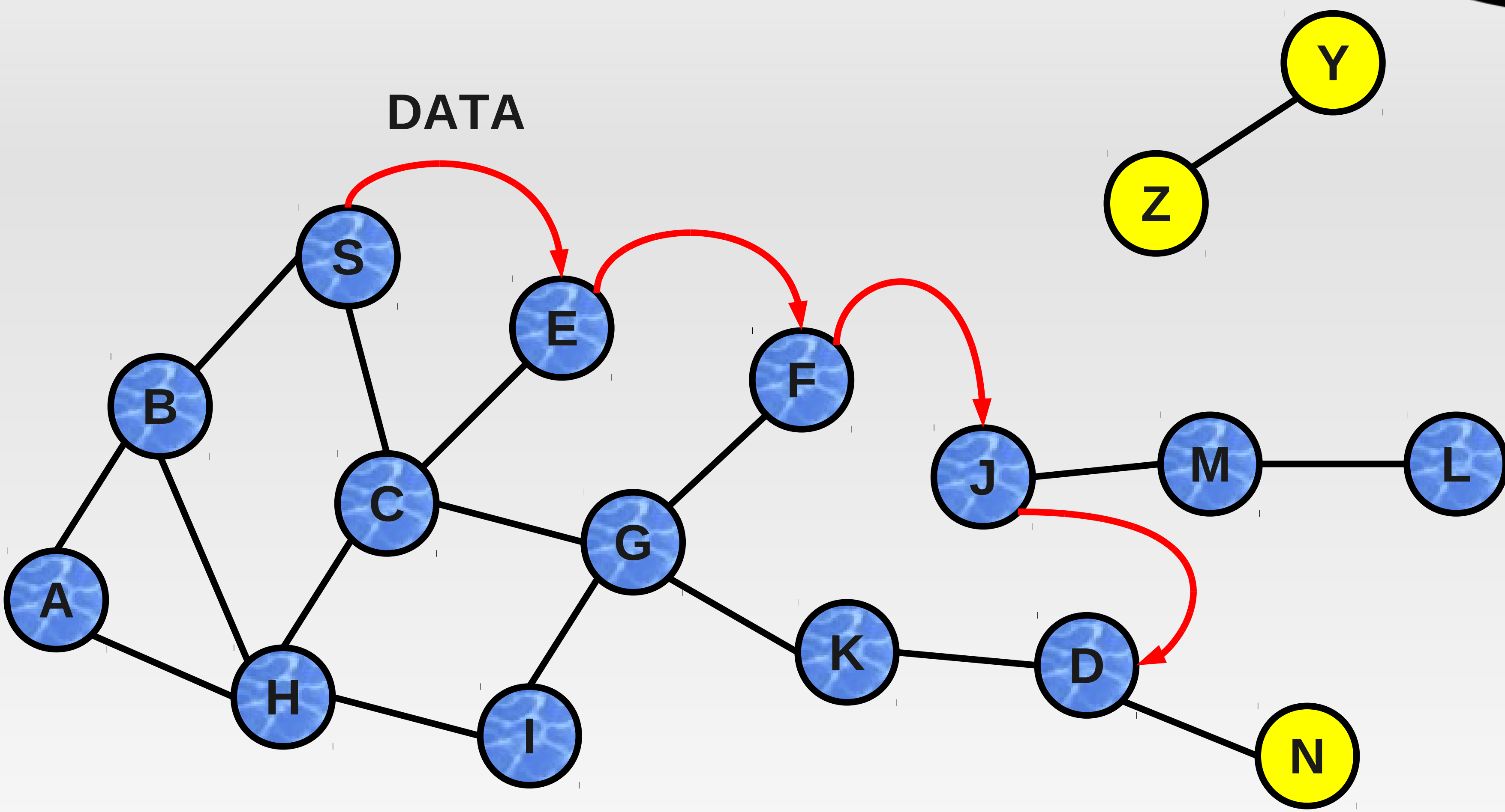
- An intermediate node (not the destination) may also send a Route Reply (RREP) provided that it knows a more recent path than the one previously known to sender S
- To determine whether the path known to an intermediate node is more recent, destination sequence numbers are used
  - A new RREQ by node S for a destination is assigned a higher destination sequence number. An intermediate node which knows a route, but with a smaller sequence number, cannot send RREP

Route Requests in AODV



Forward links are setup when RREP travels along the reverse path
Represents a link on the forward path

Route Requests in AODV



Routing table entries used to forward data packet.
Route is not included in packet header

Timeout

- A routing table entry maintaining a reverse path is purged after a timeout interval
  - timeout should be long enough to allow RREP to come back
- A routing table entry maintaining a forward path is purged if not used for a active_route_timeout interval
  - if no data is being sent using a particular routing table entry, that entry will be deleted from the routing table (even if the route may actually still be valid)

Link Failure Reporting

- A neighbor of node X is considered active for a routing table entry, if the neighbor sent a packet within active_route_timeout interval which was forwarded using that entry
- When the next hop link in a routing table entry breaks, all active neighbors are informed
- Link failures are propagated by means of Route Error (RERR) messages, which also update destination sequence numbers

Route Error

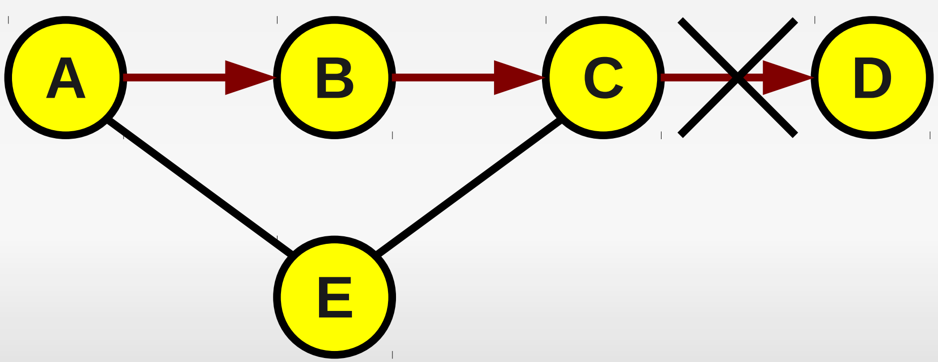
- When node X is unable to forward packet P (from node S to node D) on link (X,Y), it generates a RERR message
- Node X increments the destination sequence number for D cached at node X
- The incremented sequence number N is included in the RERR
- When node S receives the RERR, it initiates a new route discovery for D using destination sequence number at least as large as N
- When node D receives the RREQ with destination sequence number N, node D will set its sequence number to N, unless it is already larger than N

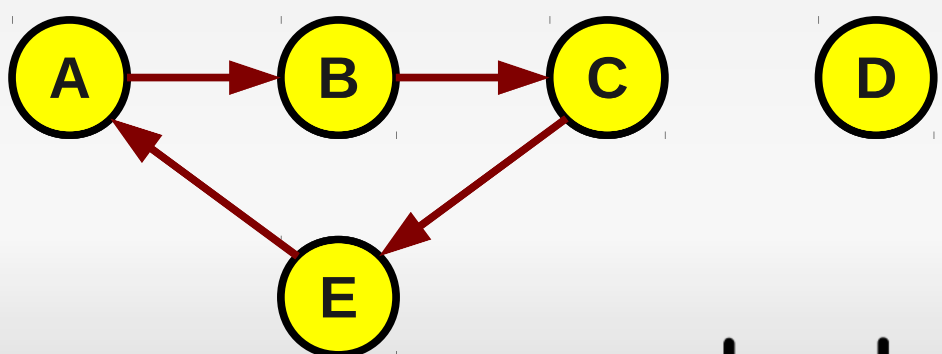
Link Failure Detection

- ***Hello*** messages: Neighboring nodes periodically exchange hello message
- Absence of hello message is used as an indication of link failure
- Alternatively, failure to receive several MAC-level acknowledgement may be used as an indication of link failure

Why Sequence Numbers in AODV

- To avoid using old/broken routes
  - To determine which route is newer
- To prevent formation of loops
  - Assume that A does not know about failure of link CD because RERR sent by C is lost
  - Now C performs a route discovery for D. Node A receives the RREQ (say, via path C-E-A)
  - Node A will reply since A knows a route to D via node B
  - Results in a loop (for instance, C-E-A-B-C)





Optimization: Expanding Ring Search

- RREQs are initially sent with small Time-to-Live (TTL) field, to limit their propagation
  - DSR also includes a similar optimization
- If no Route Reply is received, then larger TTL tried

AODV for Linux

AODV for Linux

- There exists several Linux implementations of the AODV routing protocols
- In this course we use AODV-UU, the version developed at Uppsala University
  - Most stable version
  - Available at http://sourceforge.net/projects/aodvuu/
- Drawback: it does not support correctly multi-interface devices
- We will force multi-hop communication setting filtering rules with iptables

ubuntu

AODV-UU

- AODV-UU implements the version 13 of the RFC 3561 (AODV)
- AODV-UU is composed of
  - A user-space process (aodvd) which implements the routing algorithm
  - A kernel-space loadable module (kaodv) which captures the data packets
- The software must be cross-compiled on the host machine
- Pre-compiled modules are available on the website of the course

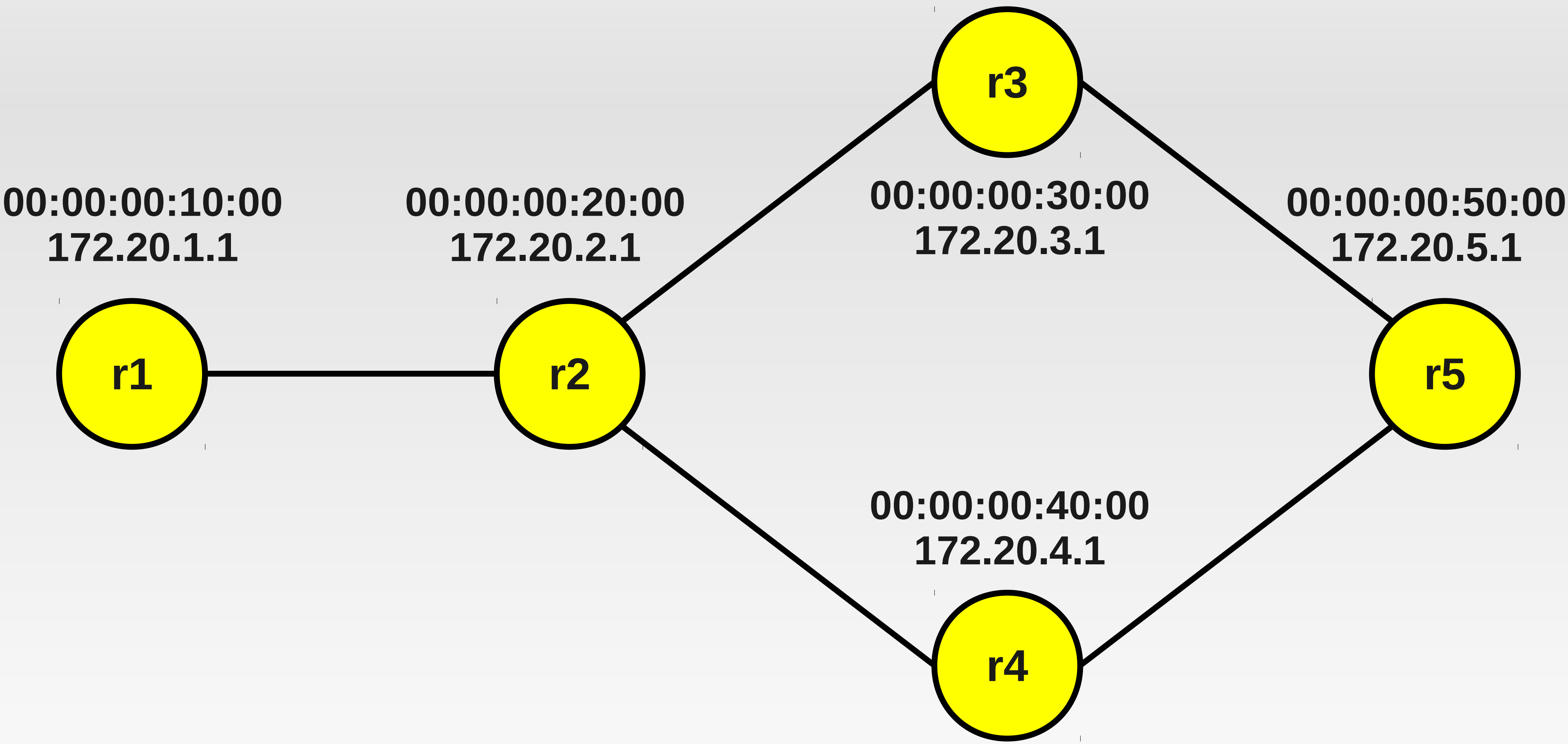
AODV-UU

AODV-UU

File Edit View Terminal Help

```
stefano@bender:~$ ./aodvd -h

Usage: aodvd [-dghjlouwxLDRV] [-i if0,if1,..] [-r N] [-n N] [-q THR]

-d, --daemon             Daemon mode, i.e. detach from the console.
-g, --force-gratuitous   Force the gratuitous flag to be set on all RREQ's.
-h, --help               This information.
-i, --interface          Network interfaces to attach to. Defaults to first
                         wireless interface.
-j, --hello-jitter       Toggle hello jittering (default ON).
-l, --log                Log debug output to /var/log/aodvd.log.
-o, --opt-hellos         Send HELLOs only when forwarding data (experimental).
-r, --log-rt-table       Log routing table to /var/log/aodvd.rtlog every N secs.
-n, --n-hellos           Receive N hellos from host before treating as neighbor.
-u, --unidir-hack        Detect and avoid unidirectional links (experimental).
-w, --gateway-mode       Enable experimental Internet gateway support.
-x, --no-expanding-ring  Disable expanding ring search for RREQs.
-D, --no-worb            Disable 15 seconds wait on reboot delay.
-L, --local-repair       Enable local repair.
-f, --llfeedback         Enable link layer feedback.
-R, --rate-limit         Toggle rate limiting of RREQs and RERRs (default ON).
-q, --quality-threshold  Set a minimum signal quality threshold for control packets.
-V, --version            Show version.

Erik Nordstr�m, <erik.nordstrom@it.uu.se>

stefano@bender:~$ 
```

Network Topology



Network Topology

- Multi-hop communication is forced definig a white-list:
  - List of MAC addresses from which frames are accepted
  - Any other frame containing a different src MAC address is discarded
- The routing algorithm, which operates at the network layer, does not see routing messages

Network Topology

- Example of white-list configuration
  - Accept all frames containig as source MAC address 00:00:00:20:00 (r1)

    iptables -t mangle -A PREROUTING -m mac \\

    --mac-source 00:00:00:20:00 -j ACCEPT
  - Drop all frames

    iptables -t mangle -A PREROUTING -i eth0 -j DROP
- The order of the two commands is important, since it defines the triggering order of the rules

Run AODV protocol

- After having set up the network interface, we start the sw daemon implementing the AODV routing algorithm

    aodvd -l -r 3 -i eth0 -d

- Log both the aodvd messages (-l) and the routing table updates (-r)
    - The routing table is sampled every 3 seconds
- Attach the daemon on te network interface eth0
- Run in background (-d)