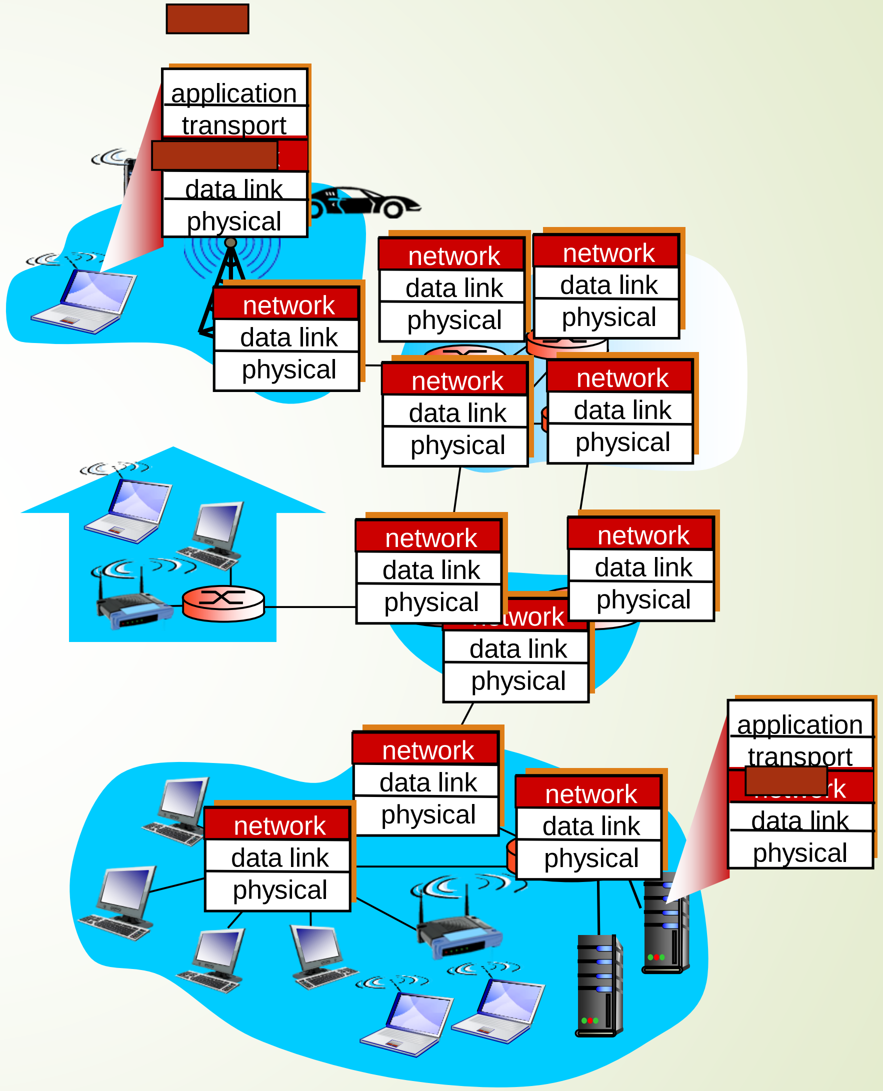
Lecture 7-12: Routing in Vehicular Ad-Hoc Networks(VANETs)

Dr. Debasis Das
Department of CSE
IIT Jodhpur,Rajasthan, India.

Network layer

- transport segment from sending to receiving host
- on sending side encapsulates segments into datagrams
- on receiving side, delivers segments to transport layer
- network layer protocols in *every* host, router



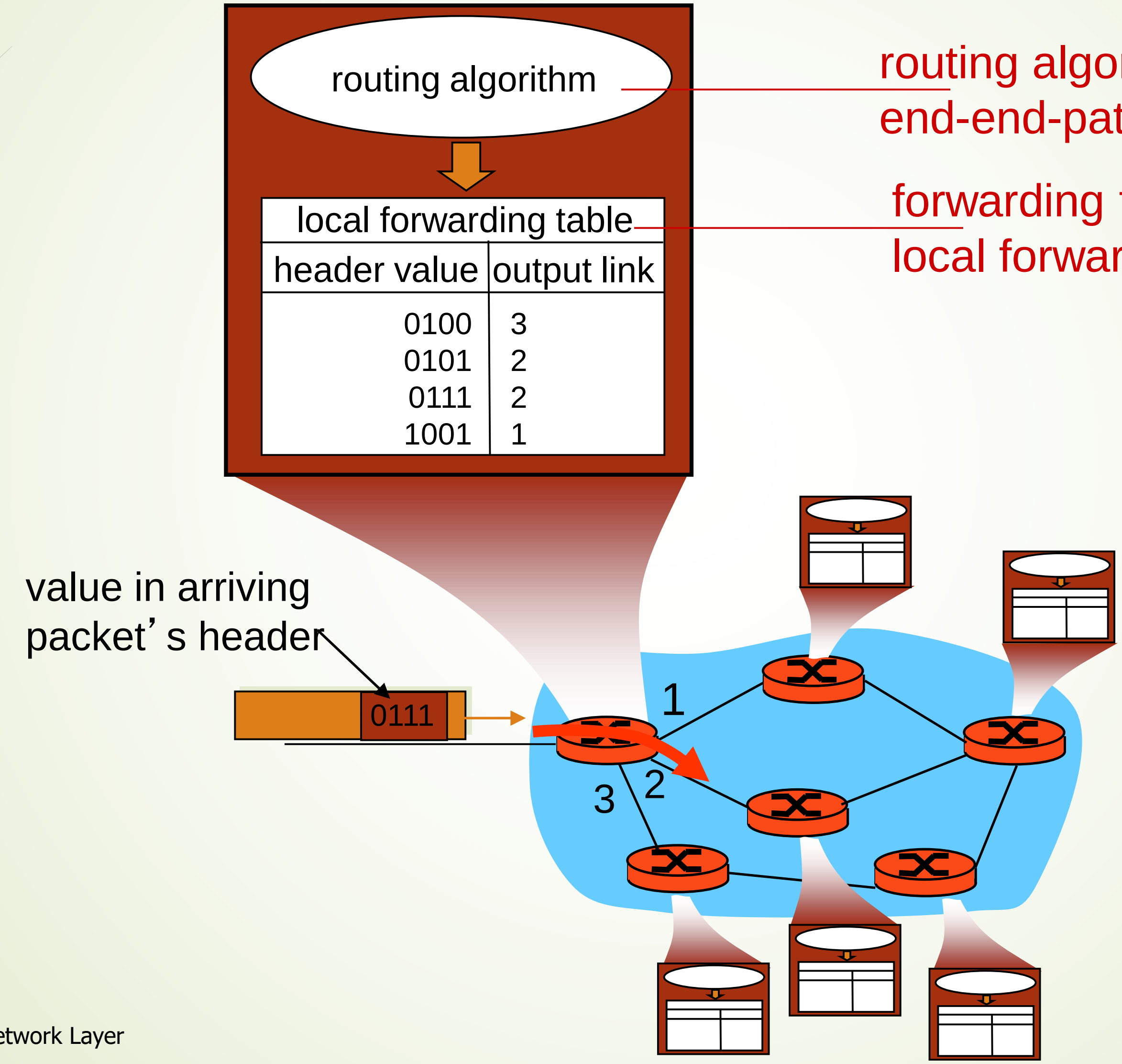
Two key network-layer functions

- *forwarding:* move packets from router's input to appropriate router output
- *routing:* determine route taken by packets from source to dest.
  - *routing algorithms*

analogy:

- *routing:* process of planning trip from source to dest
- *forwarding:* process of getting through single interchange

Interplay between routing and forwarding



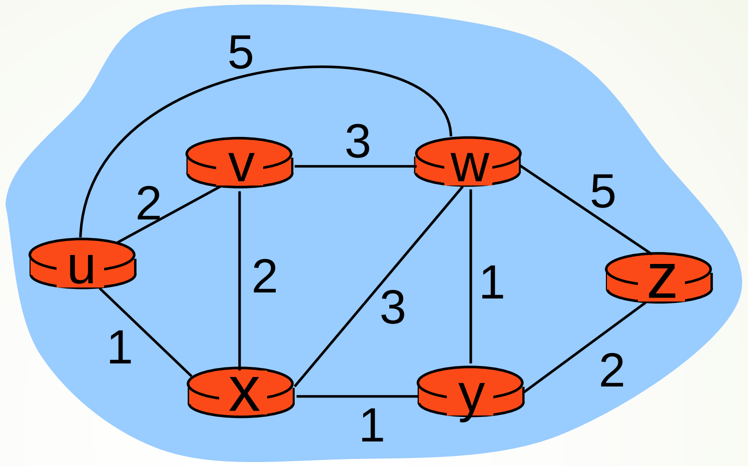
local forwarding table

header value	output link
0100	3
0101	2
0111	2
1001	1

routing algorithm determines end-end-path through network

forwarding table determines local forwarding at this router

Graph abstraction



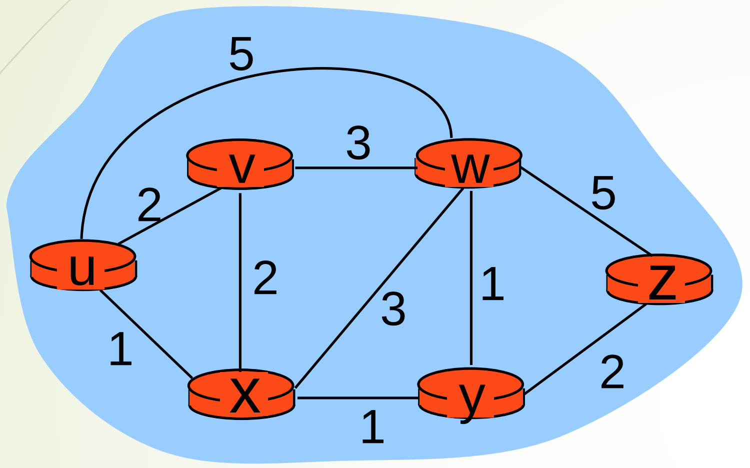
graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

aside: graph abstraction is useful in other network contexts, e.g., P2P, where *N* is set of peers and *E* is set of TCP connections

Graph abstraction: costs



c(x,x') = cost of link (x,x')
e.g., c(w,z) = 5

cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

cost of path $(x_1, x_2, x_3, \dots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \dots + c(x_{p-1},x_p)$

key question: what is the least-cost path between u and z ?
routing algorithm: algorithm that finds that least cost path

Routing algorithm classification

Q: global or decentralized information?

global:

- all routers have complete topology, link cost info
- "link state" algorithms

decentralized:

- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms

Q: static or dynamic?

static:

- ❖ routes change slowly over time

dynamic:

- ❖ routes change more quickly
  - ▪ periodic update
  - ▪ in response to link cost changes

Routing protocols

1. Link state
2. Distance Vector
3. Hierarchical Routing

Distance-vector & Link-state Routing

- **Link state** - router knows entire network topology and computes shortest path
- **Distance vector** - router knows cost to each destination

A Link-State Routing Algorithm

Dijkstra 's algorithm

- network topology, link costs known to all nodes
  - ✓ accomplished via "link state broadcast"
  - ✓ all nodes have same info
- computes least cost paths from one node ('source") to all other nodes
  - ✓ gives *forwarding table* for that node

notation:

- ❖ c(x,y): link cost from node x to y; = ∞ if not direct neighbors
- ❖ D(v): current value of cost of path from source to dest. v
- ❖ p(v): predecessor node along path from source to v
- ❖ N': set of nodes whose least cost path definitively known

Dijsktra’ s Algorithm

- ❑ update D(v) for all v adjacent to w and not in N' :
- ❑ **D(v) = min(D(v), D(w) + c(w,v)) ,**

(new cost to v is either old cost to v or known
 shortest path cost to w plus cost from w to v)

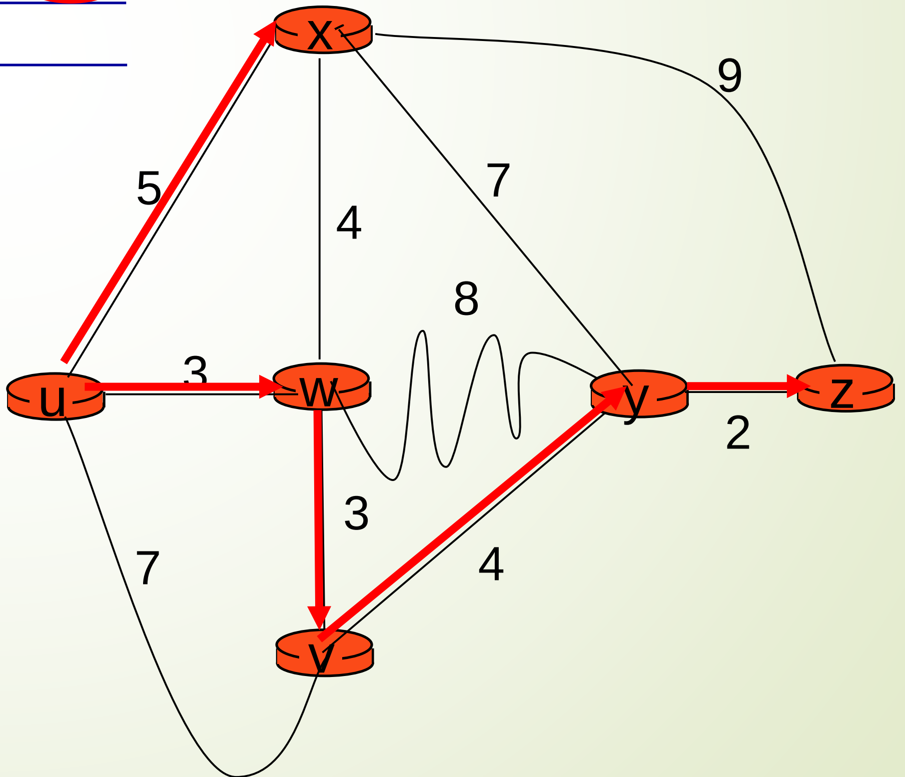
- ❑ ***until all nodes in N'***

Dijkstra's algorithm: example

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

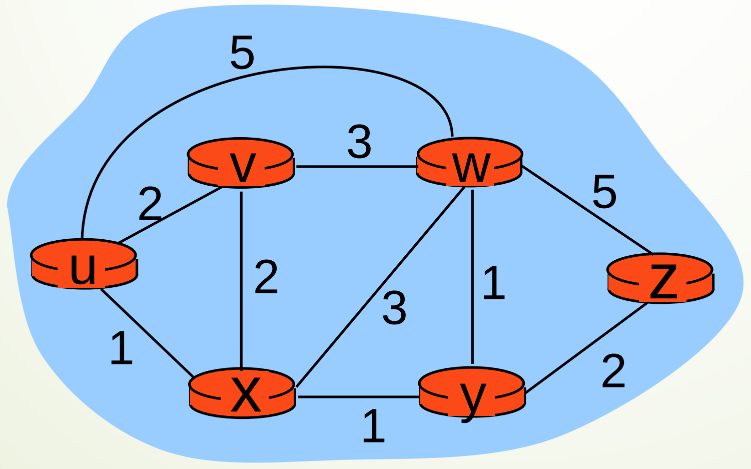
notes:

- construct shortest path tree by tracing predecessor nodes



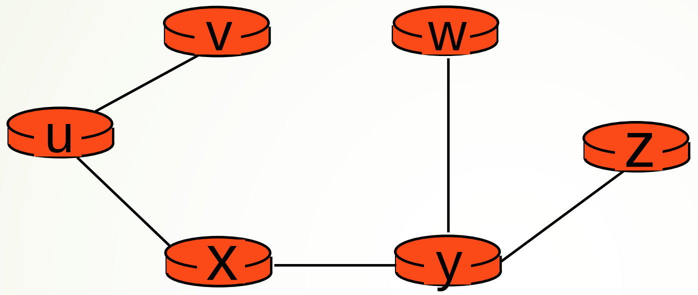
Dijkstra's algorithm: another example

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Dijkstra's algorithm: example (2)

resulting shortest-path tree from u:



resulting forwarding table in u:

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

Dijkstra' s algorithm, discussion

algorithm complexity: n nodes

- each iteration: need to check all nodes,
- n(n+1)/2 comparisons: O(n^2)
- more efficient implementations possible: O(nlogn)

- Given a graph and a source vertex *src* in graph, find shortest paths from *src* to all vertices in the given graph. The graph may contain negative weight edges.

- We have discussed Dijkstra's algorithm for this problem. Dijksra's algorithm is a Greedy algorithm and time complexity is O(VLogV) (with the use of Fibonacci heap). *Dijkstra doesn't work for Graphs with negative weight edges, Bellman-Ford works for such graphs.*

Distance vector algorithm

Bellman-Ford equation (dynamic programming)

let

$d_x(y)$:= cost of least-cost path from x to y

then

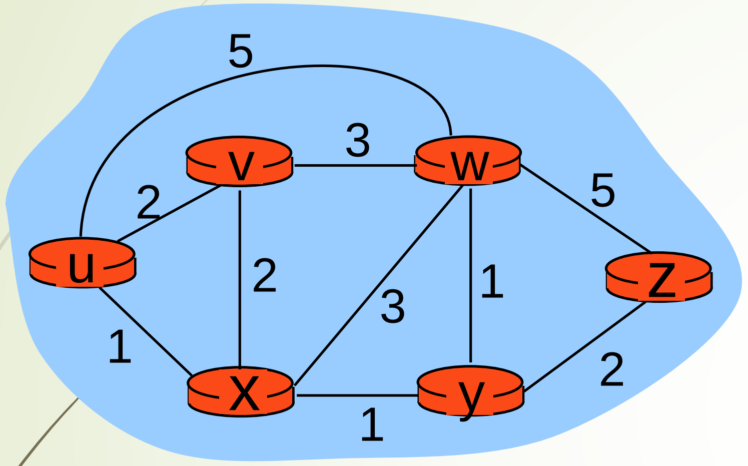
$$d_x(y) = \min_{v} \{c(x,v) + d_v(y)\}$$

cost from neighbor v to destination y

cost to neighbor v

min taken over all neighbors v of x

Bellman-Ford example



clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$\begin{aligned} d_u(z) &= \min \{ c(u,v) + d_v(z), \\ &\qquad c(u,x) + d_x(z), \\ &\qquad c(u,w) + d_w(z) \} \\ &= \min \{2 + 5, \\ &\qquad 1 + 3, \\ &\qquad 5 + 3\} = 4 \end{aligned}$$

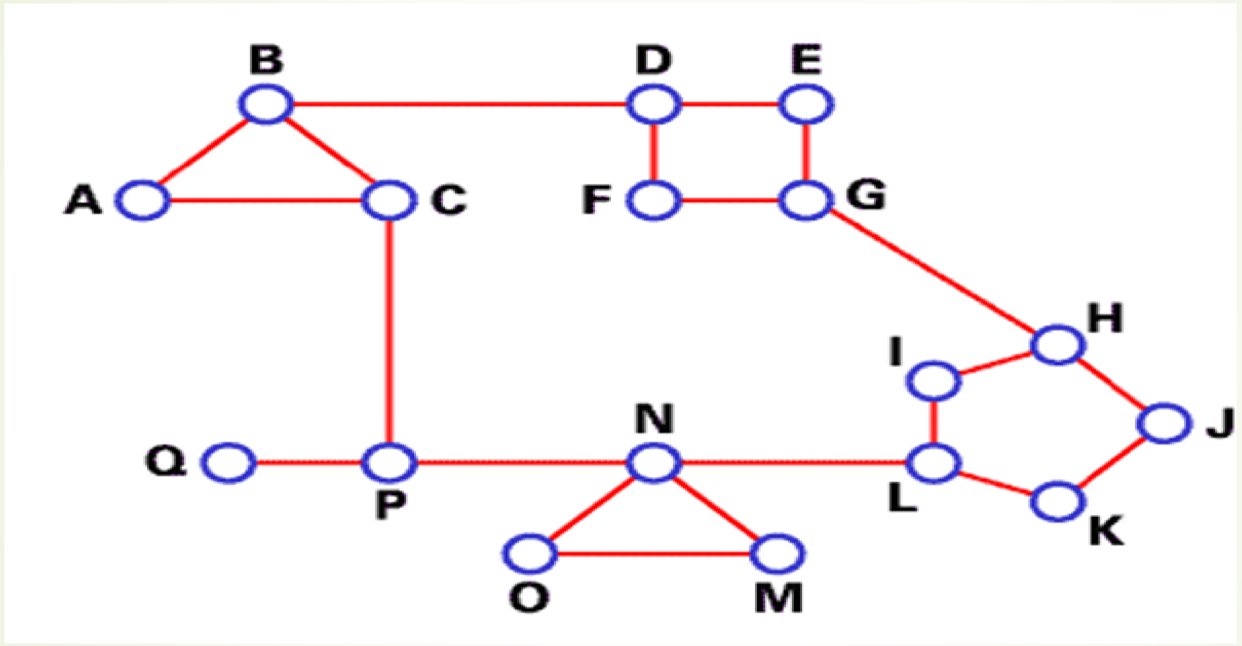
Distance vector Complexity

- Bellman-Ford is also simpler than Dijkstra and suites well for distributed systems.
- But time complexity of Bellman-Ford is O(VE), which is more than Dijkstra.

Hierarchical routing

- As you see, in both LS and DV algorithms, every router has to save some information about other routers. When the network size grows, the number of routers in the network increases. Consequently, the size of routing tables increases, as well, and routers can't handle network traffic as efficiently. We use hierarchical routing to overcome this problem.

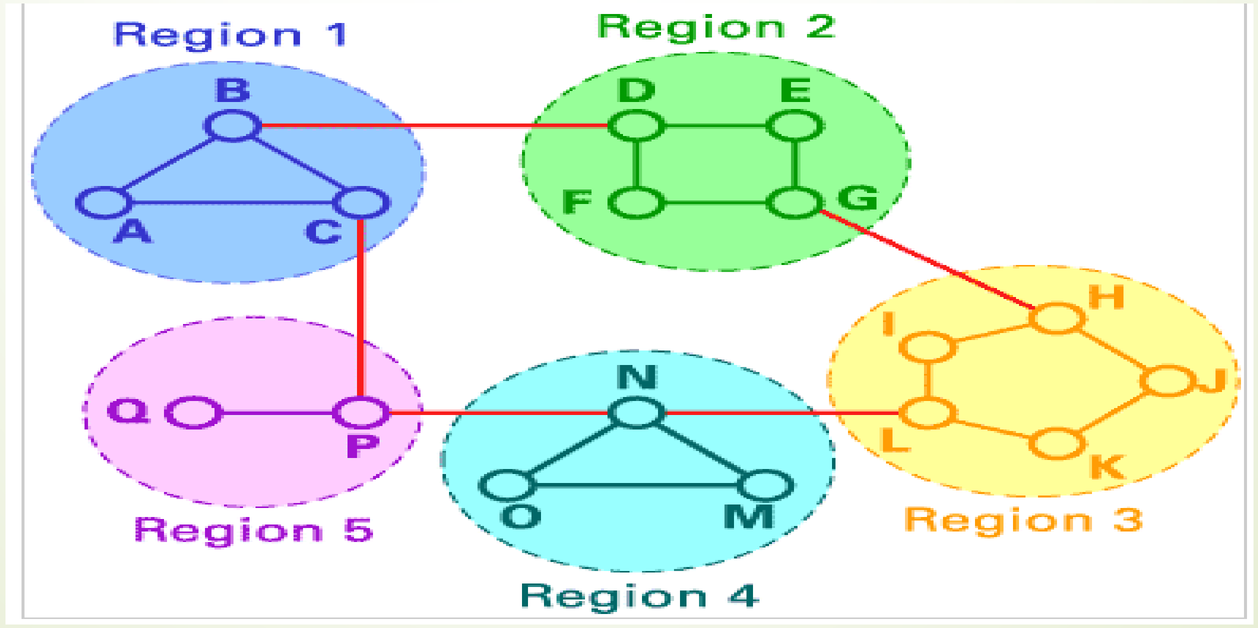
Let's examine this subject with an example



- **We use DV algorithms to find best routes between nodes. In the situation depicted below, every node of the network has to save a routing table with 17 records. Here is a typical graph**

Hierarchical routing

- In hierarchical routing, routers are classified in groups known as regions. Each router has only the information about the routers in its own region and has no information about routers in other regions. So routers just save one record in their table for every other region. In this example, we have classified our network into five regions (see below).



Three-level hierarchical routing

- In three-level hierarchical routing, the network is classified into a number of clusters. Each cluster is made up of a number of regions, and each region contains a number or routers.
- Hierarchical routing is widely used in Internet routing and makes use of several routing protocols

Routing Protocols

- Reactive protocols
  - Determine route if and when needed
  - Source initiates route discovery
  - Example:
  - DSR (Dynamic Source Routing)
  - AODV (Ad hoc On-Demand Distance Vector)
- Proactive protocols
  - Traditional distributed shortest-path protocols
  - Maintain routes between every host pair at all times
  - Based on periodic updates; High routing overhead
  - Example: DSDV (destination sequenced distance vector)
  - Optimized Link State Routing (OLSR)

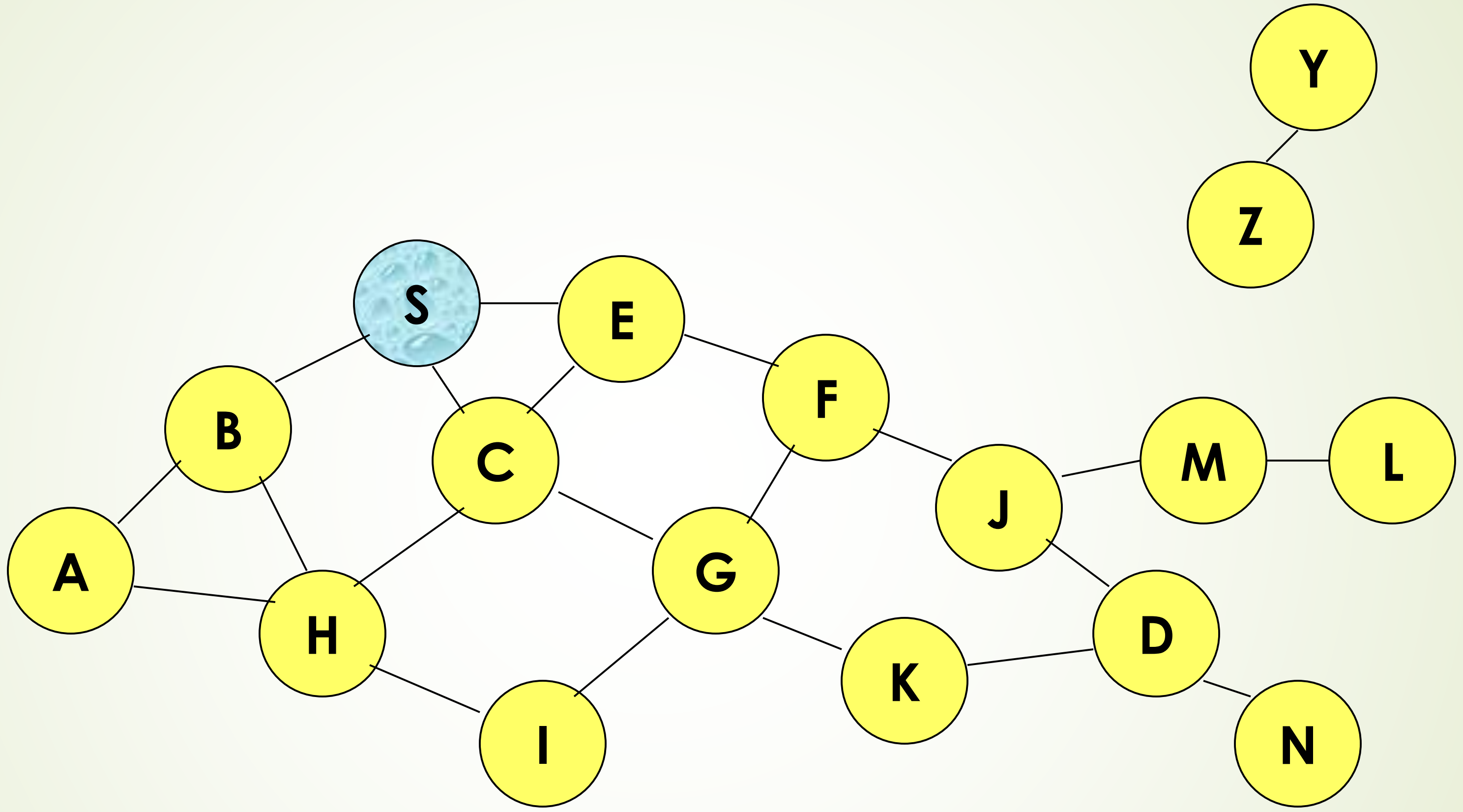
- Hybrid protocols
  - Adaptive; Combination of proactive and reactive
  - Example : ZRP (zone routing protocol)

Dynamic Source Routing

- J. Broch, D. Johnson, and D. Maltz, “The dynamic source routing protocol for mobile ad hoc networks,” *Internet-Draft Version 03, IETF, October 1999.*

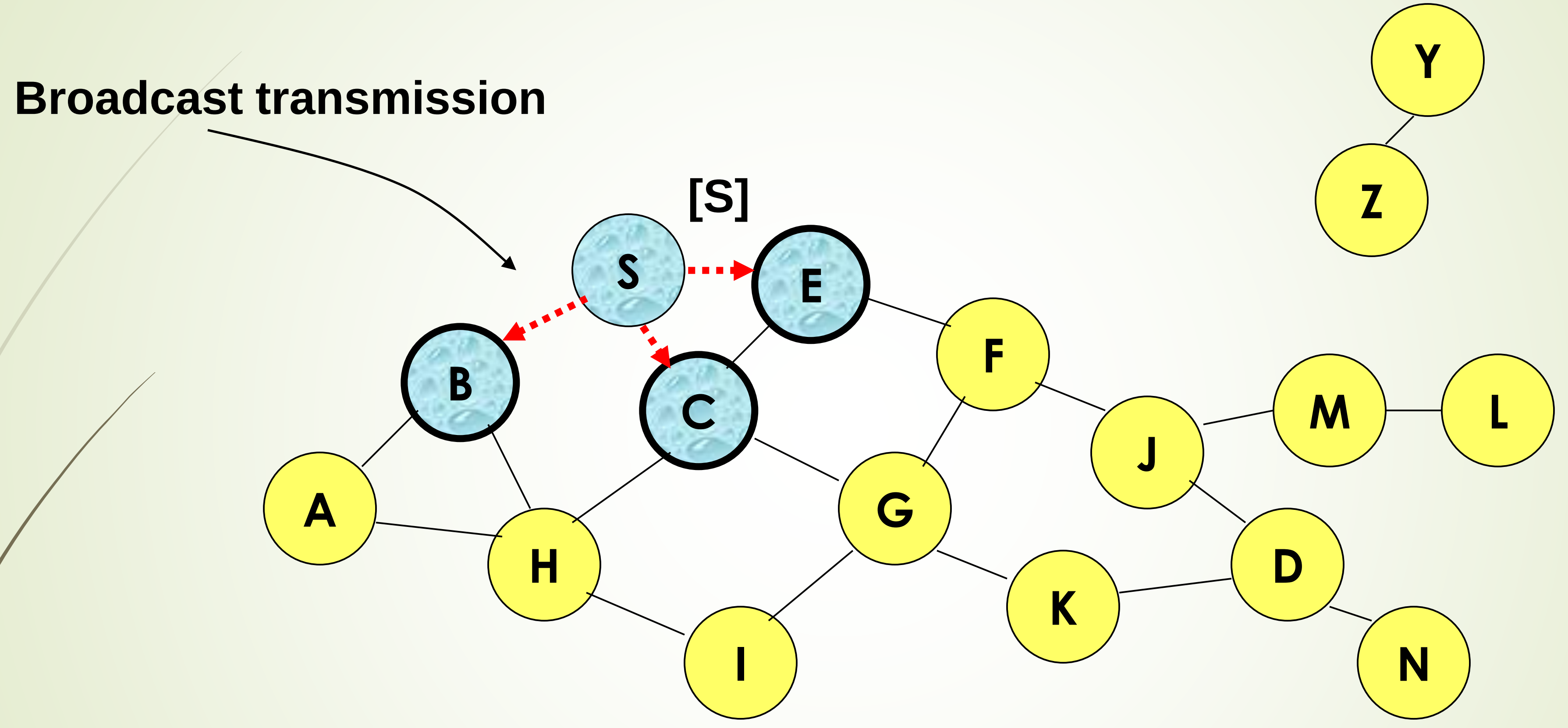
- When node S wants to send a packet to node D, but does not know a route to D, node S initiates a routing process
- Runs in three phases
  - Route Discovery → Route Reply → Path Establishment
- Route Discovery
  - Source node S floods Route Request (RREQ)
  - Each node appends own identifier when forwarding RREQ

Route Discovery in DSR



Represents a node that has received RREQ for D from S

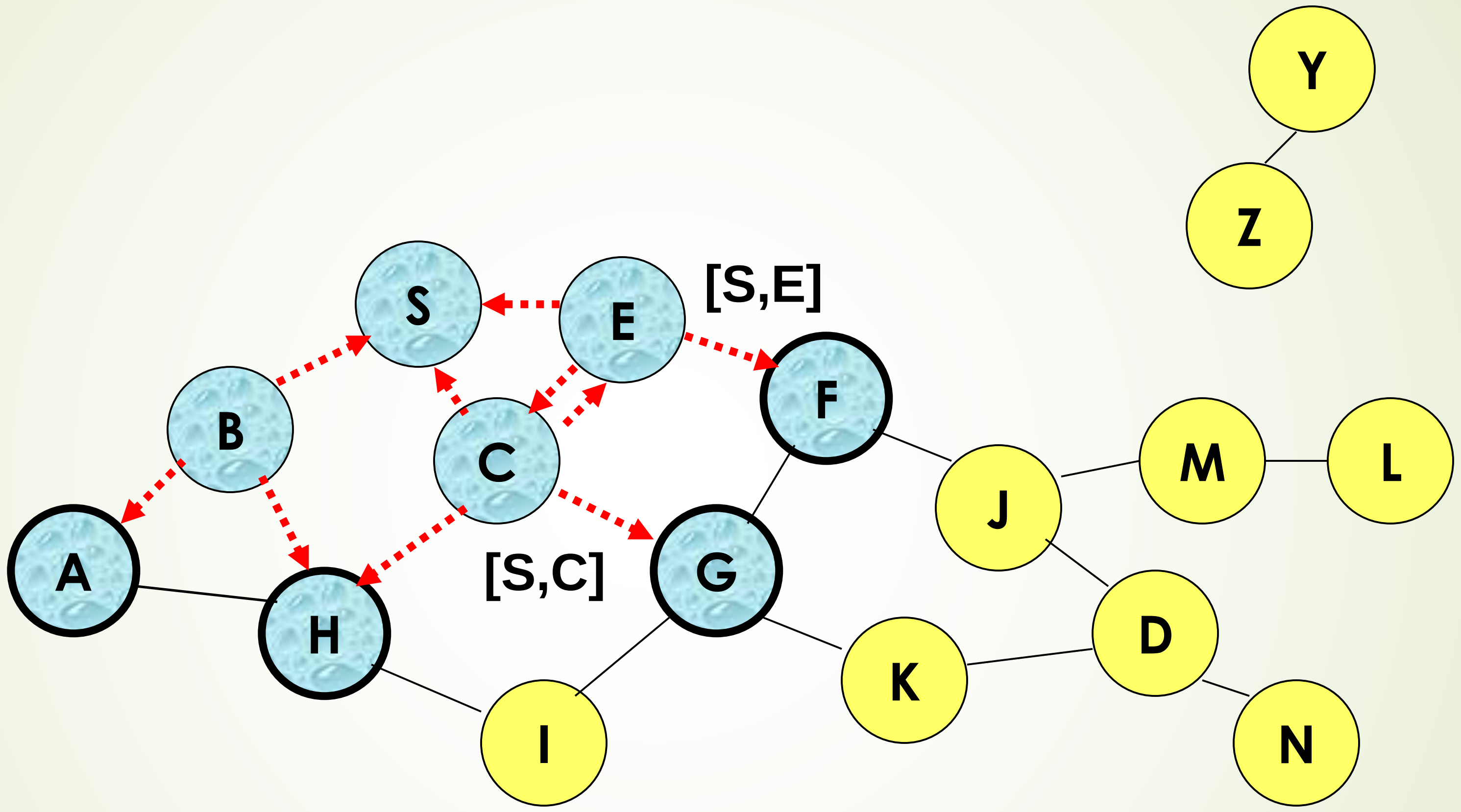
Route Discovery in DSR



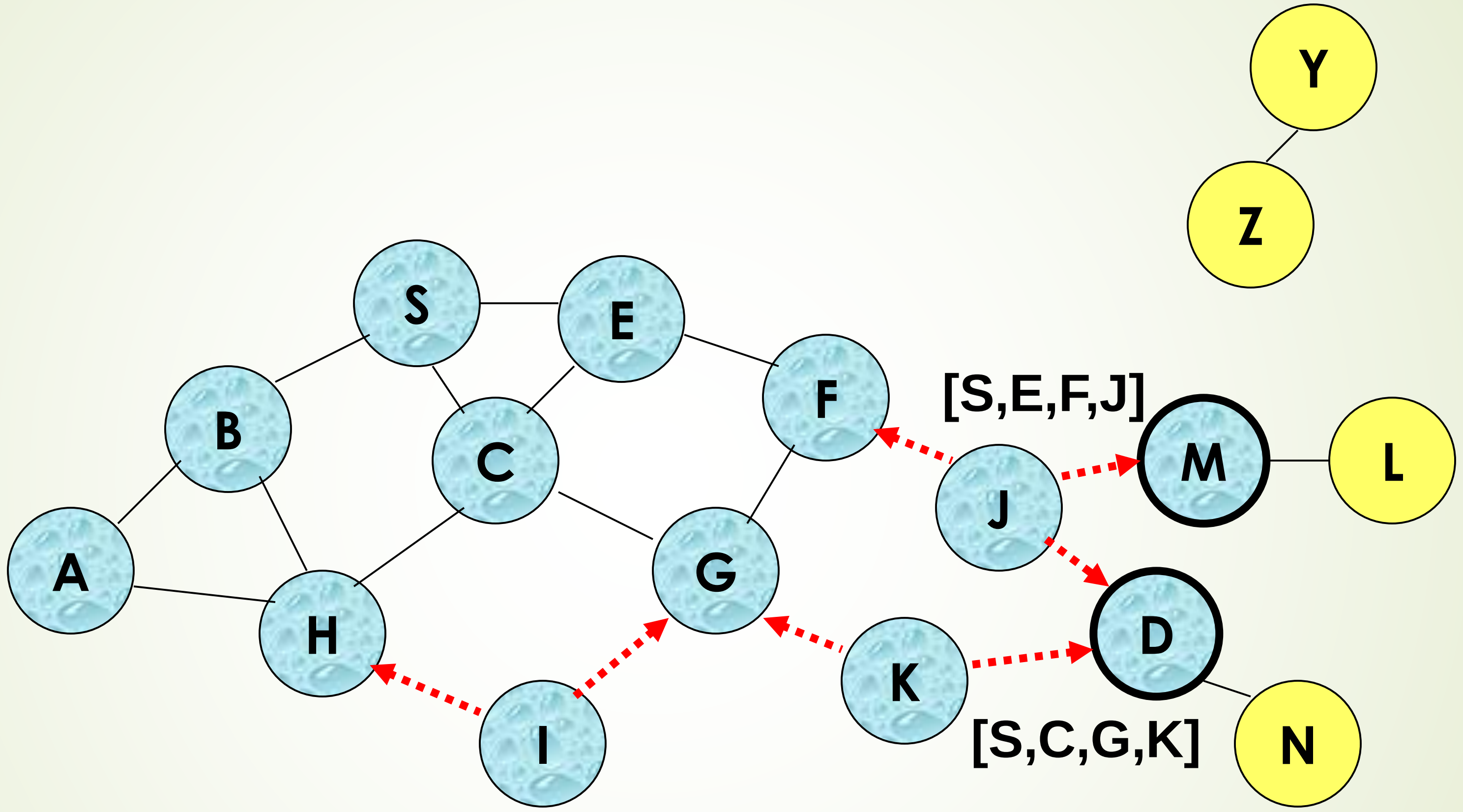
Represents transmission of RREQ

[X,Y] Represents list of identifiers appended to RREQ

Route Discovery in DSR



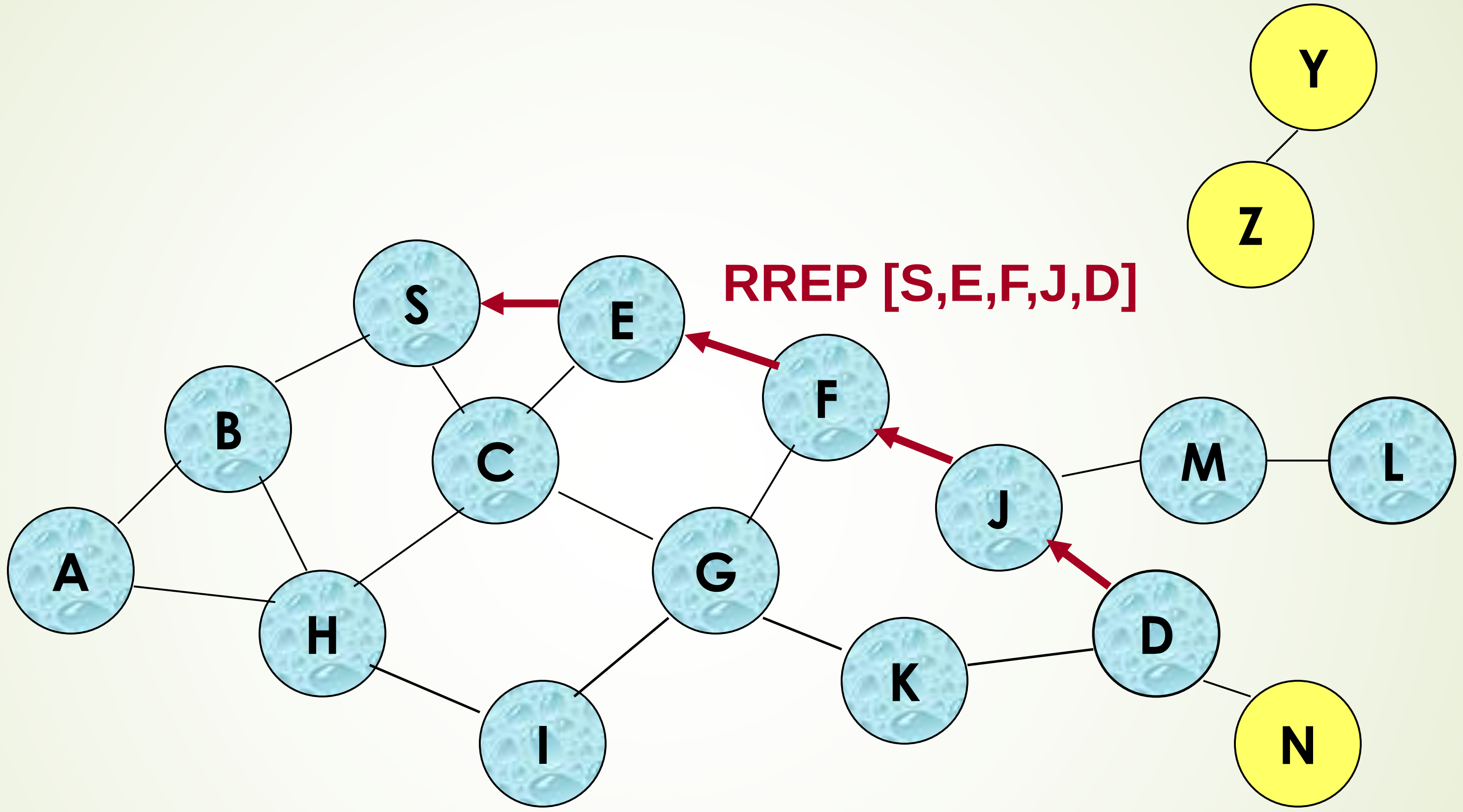
Route Discovery in DSR



Route Reply in DSR

- Destination D on receiving the first RREQ, sends a Route Reply (RREP)
- RREP is sent on a route obtained by reversing the route appended to received RREQ
- RREP includes the route from S to D on which RREQ was received by node D

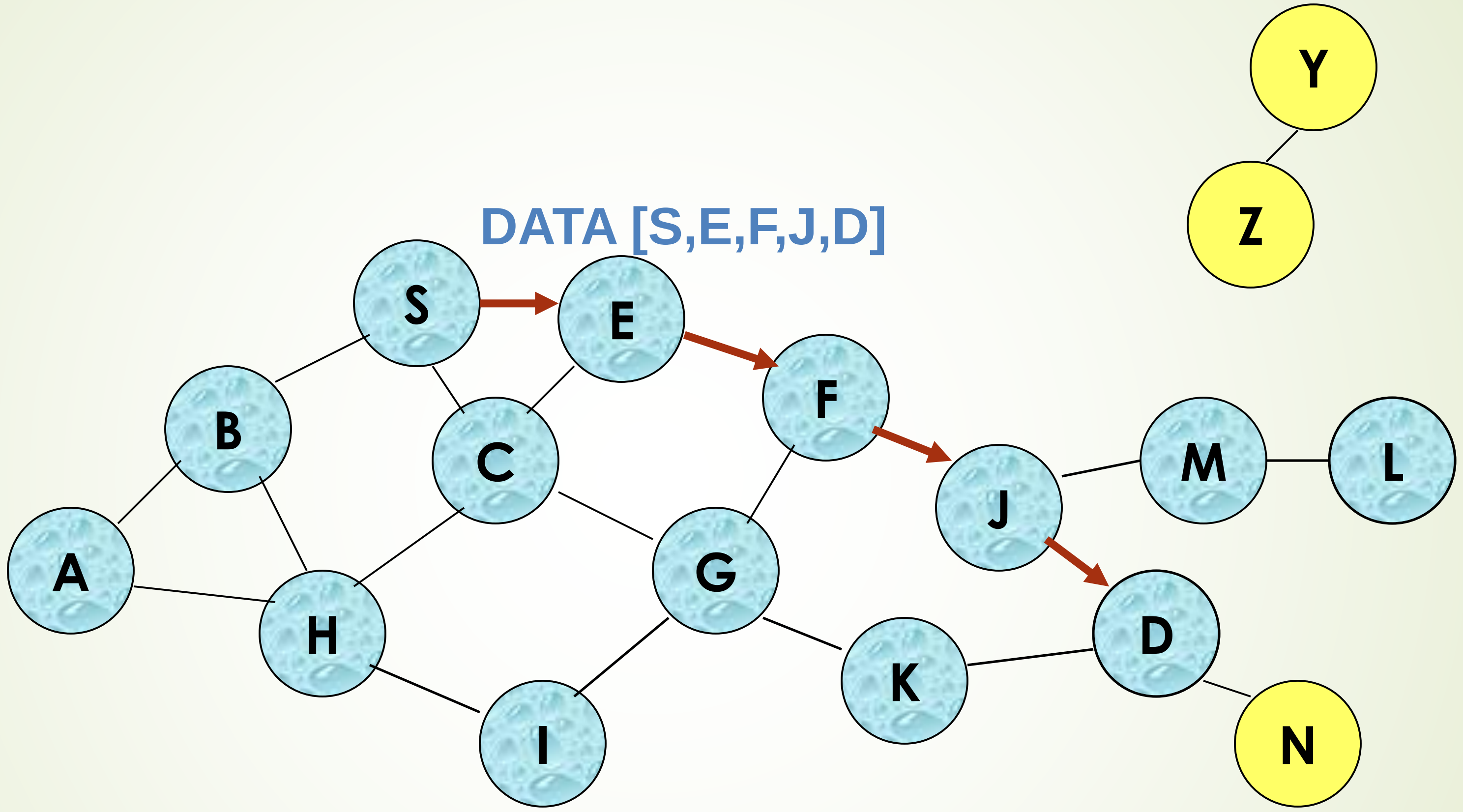
Route Reply in DSR



Route Reply in DSR

- Node S on receiving RREP, caches the route included in the RREP
- When node S sends a data packet to D, the entire route is included in the packet header
  - Hence the name source routing
- Intermediate nodes use the source route included in a packet to determine to whom a packet should be forwarded

Data Delivery in DSR



Packet header size grows with route length

Dynamic Source Routing: Advantages

- Routes maintained only between nodes who need to communicate
  - reduces overhead of route maintenance

Dynamic Source Routing: Disadvantages

- Packet header size grows with route length due to source routing
- Flood of route requests may potentially reach all nodes in the network
- Potential collisions between route requests propagated by neighboring nodes
  - insertion of random delays before forwarding RREQ
- Increased contention if too many route replies come back due to nodes replying using their local cache
  - Route Reply *Storm* problem

Ad Hoc On-Demand Distance Vector Routing (AODV) [Perkins99Wmcsa]

- DSR includes source routes in packet headers resulting large headers can sometimes degrade performance
  - particularly when data contents of a packet are small

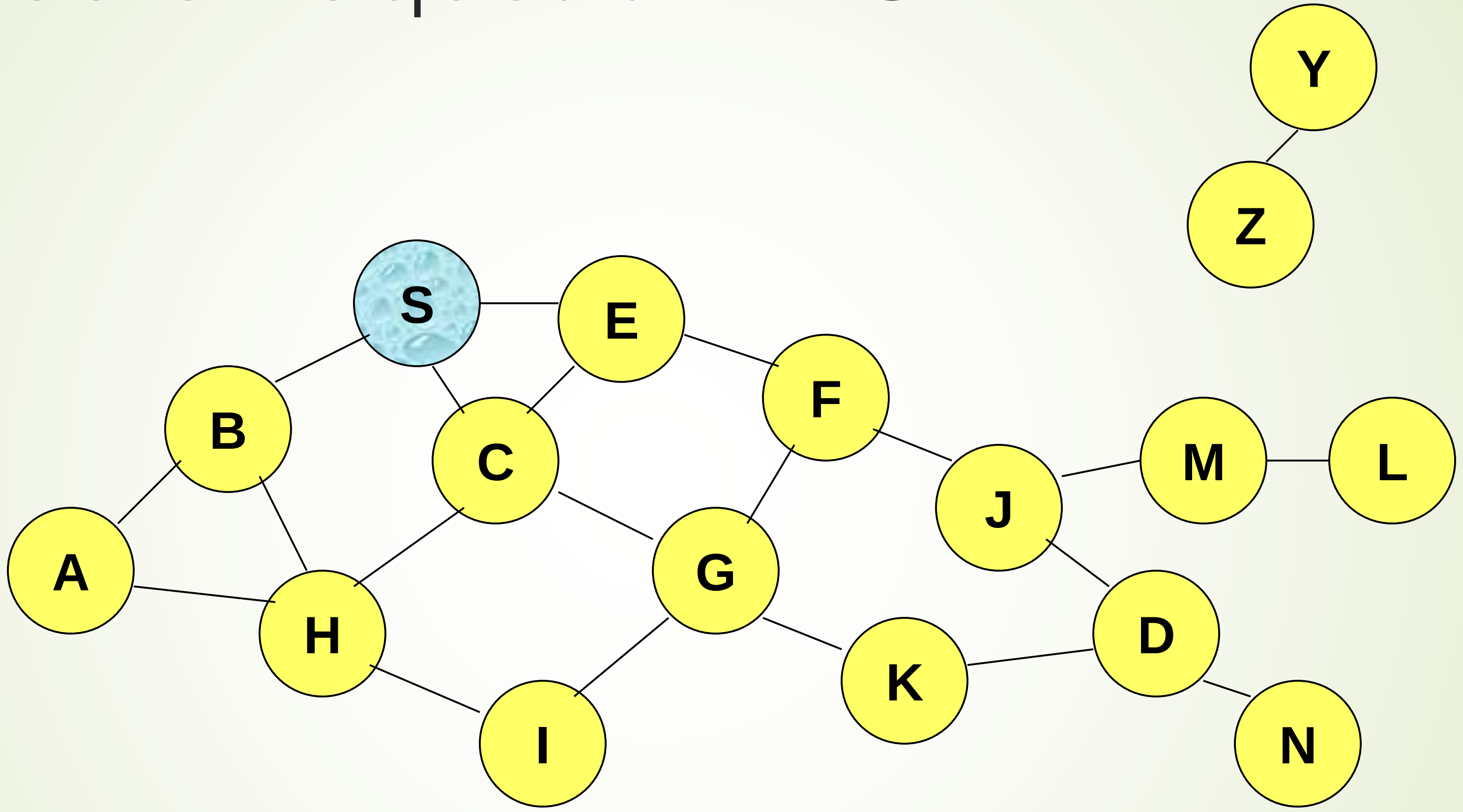
- AODV attempts to improve on DSR by maintaining routing tables at the nodes, so that data packets do not have to contain routes

- AODV retains the desirable feature of DSR that routes are maintained only between nodes which need to communicate

AODV

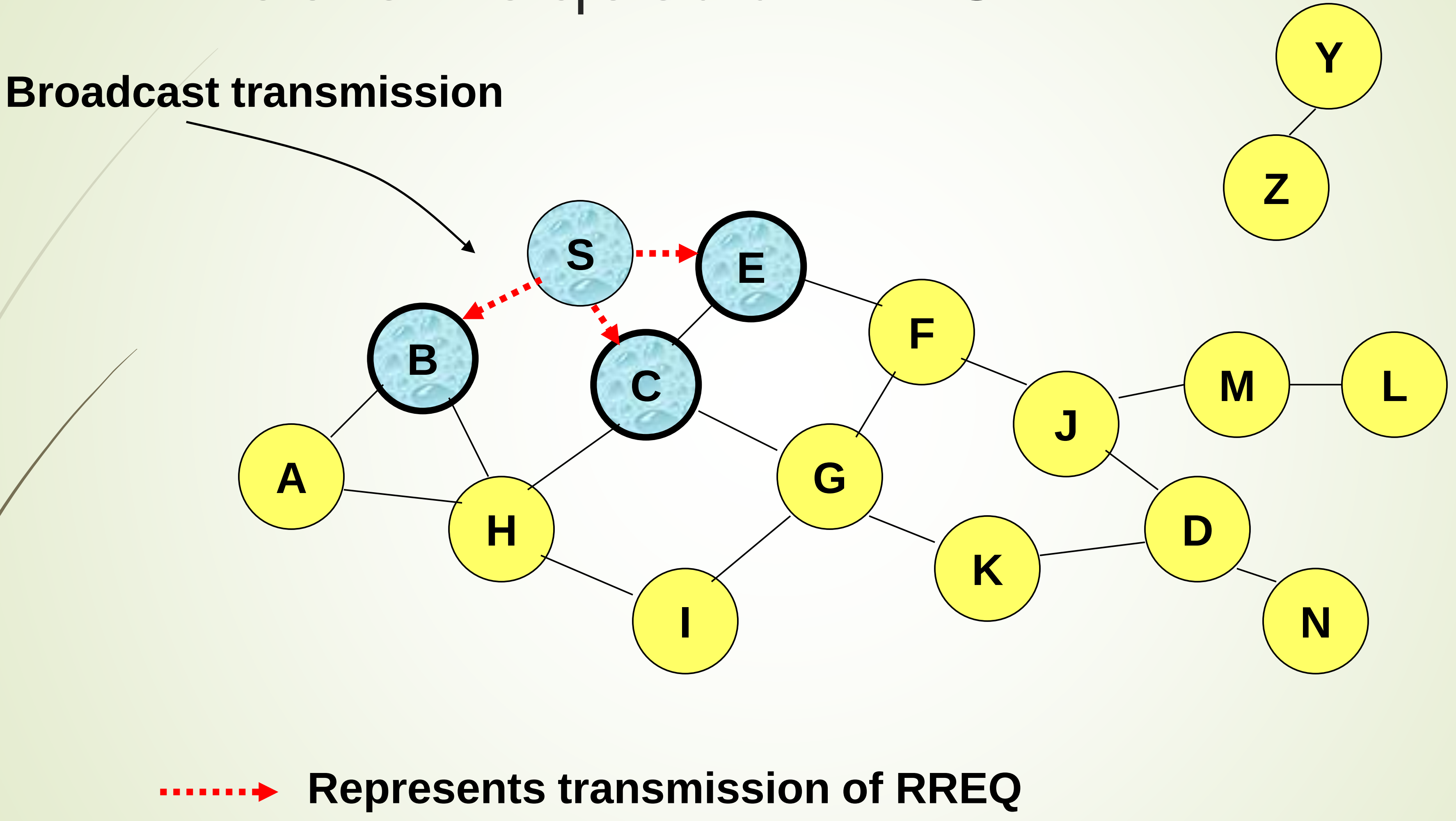
- Route Requests (RREQ) are forwarded in a manner similar to DSR
- When a node re-broadcasts a Route Request, it sets up a reverse path pointing towards the source
  - AODV assumes symmetric (bi-directional) links
- When the intended destination receives a Route Request, it replies by sending a Route Reply (RREP)
- Route Reply travels along the reverse path set-up when Route Request is forwarded

Route Requests in AODV

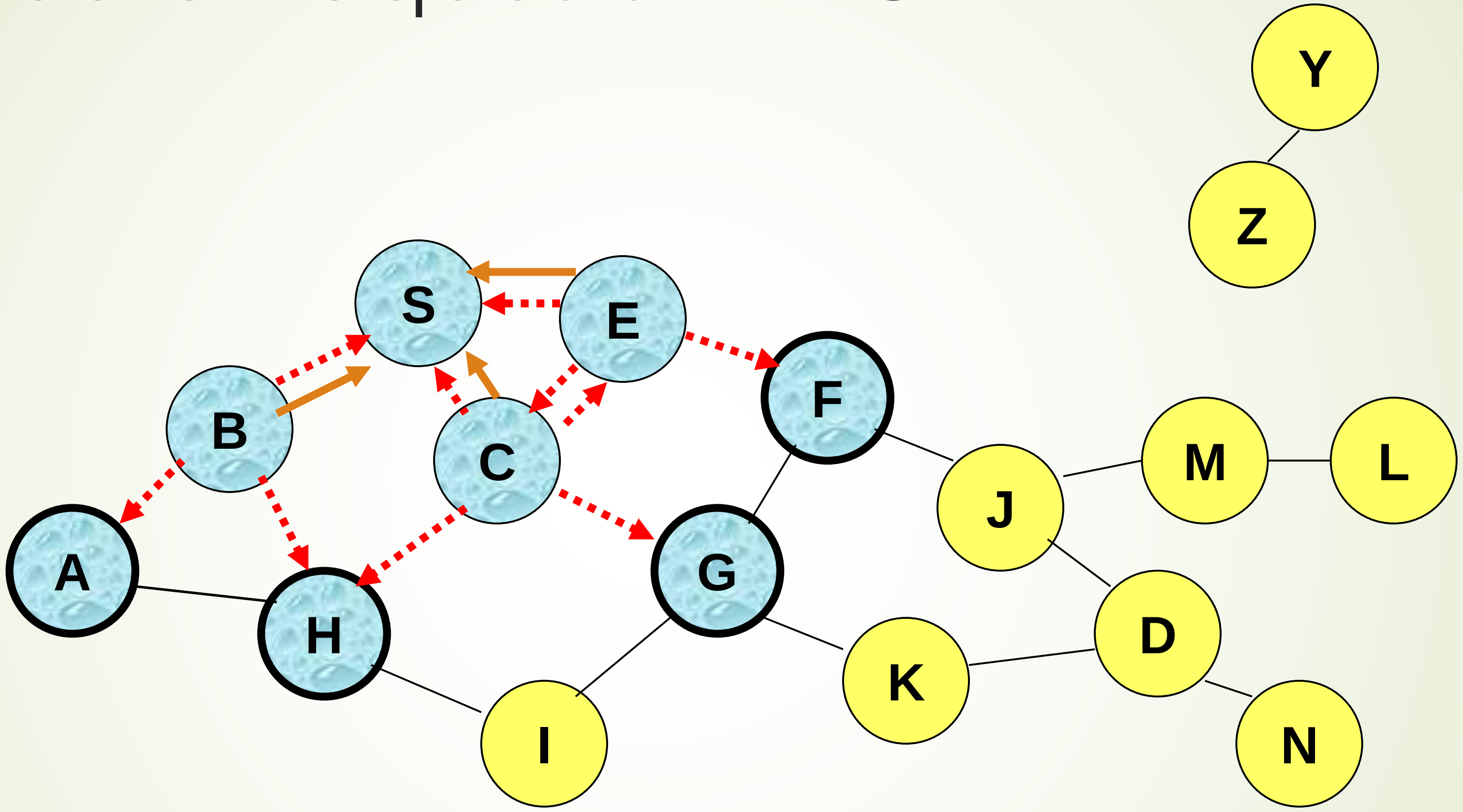


Represents a node that has received RREQ for D from S

Route Requests in AODV

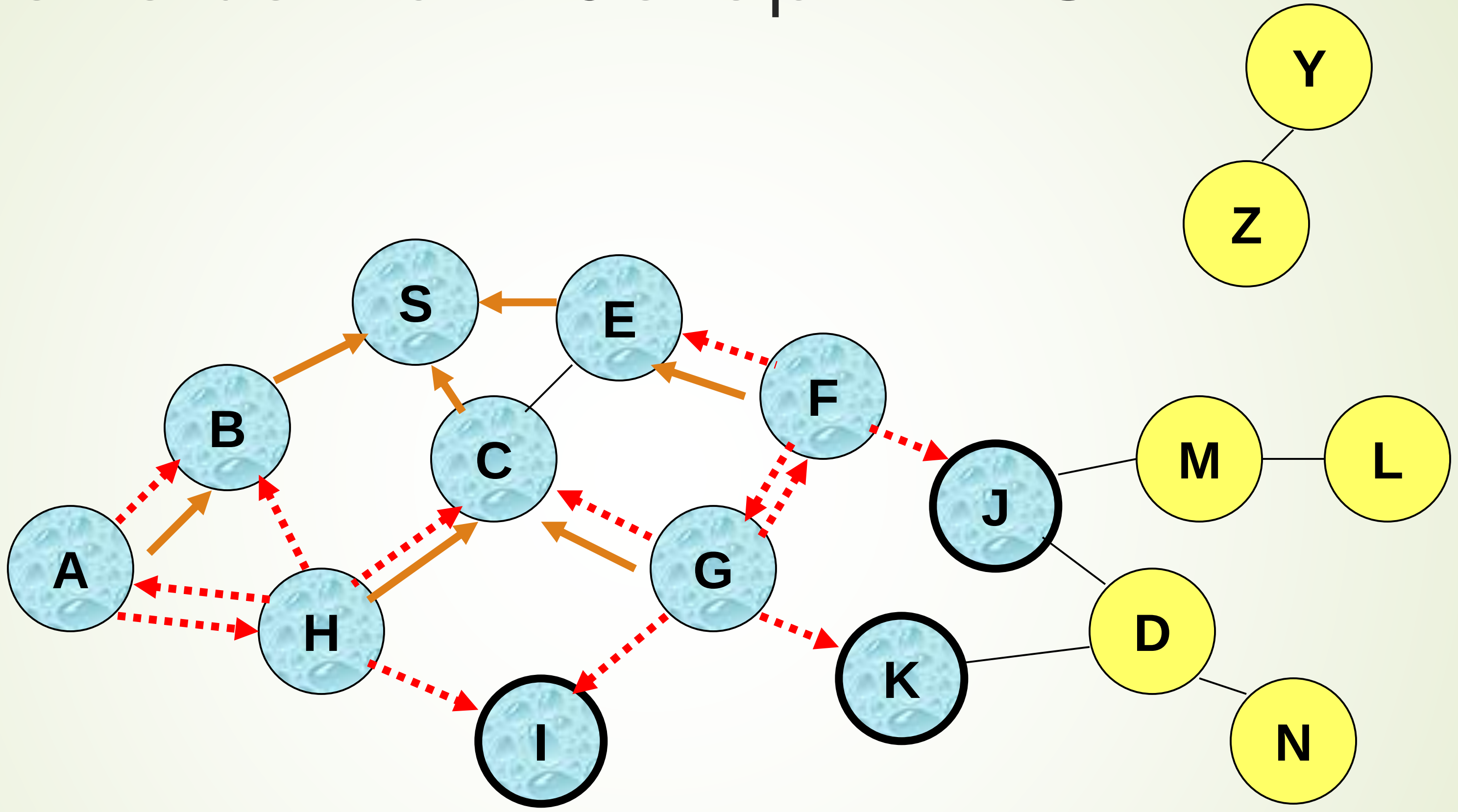


Route Requests in AODV



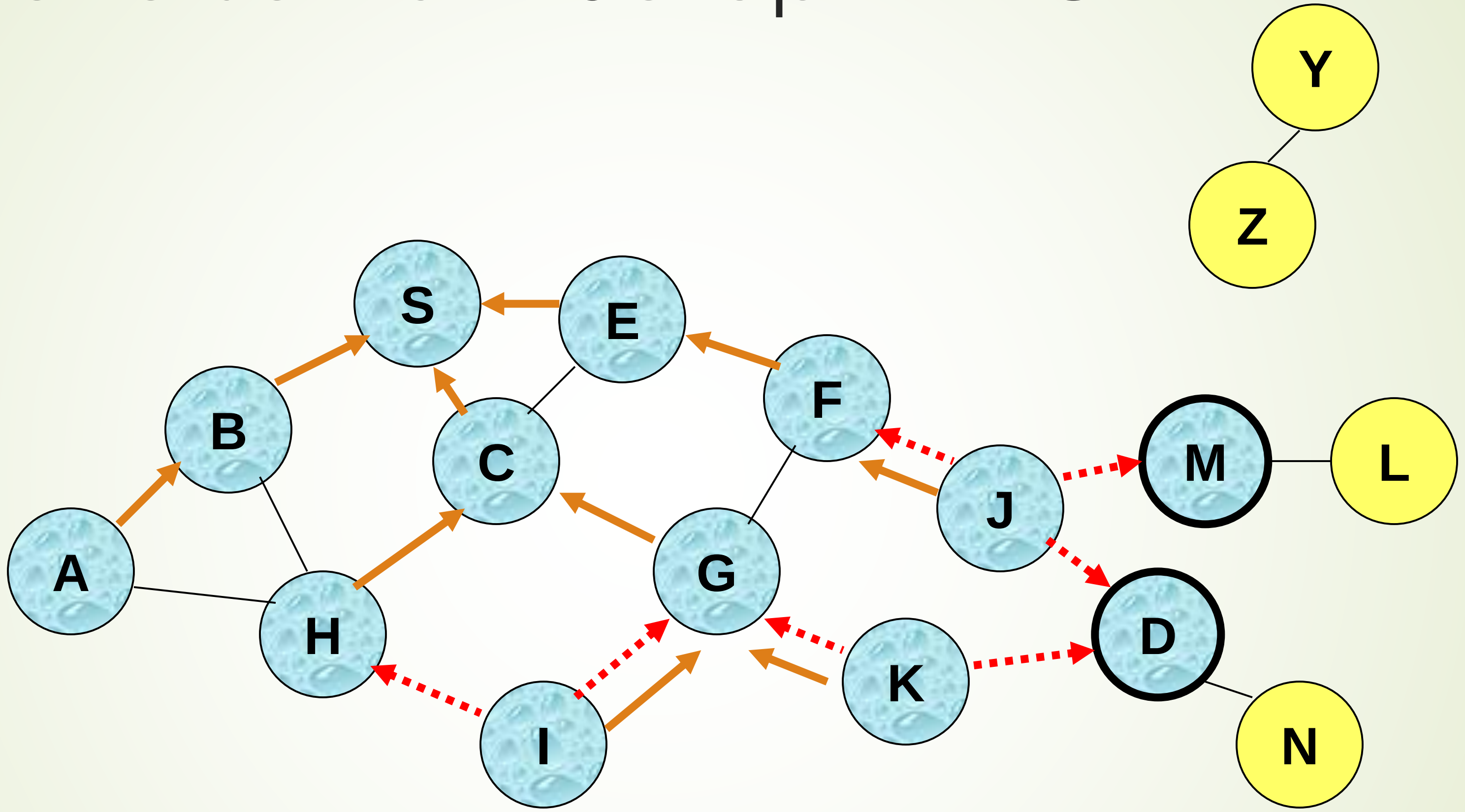
Represents links on Reverse Path

Reverse Path Setup in AODV

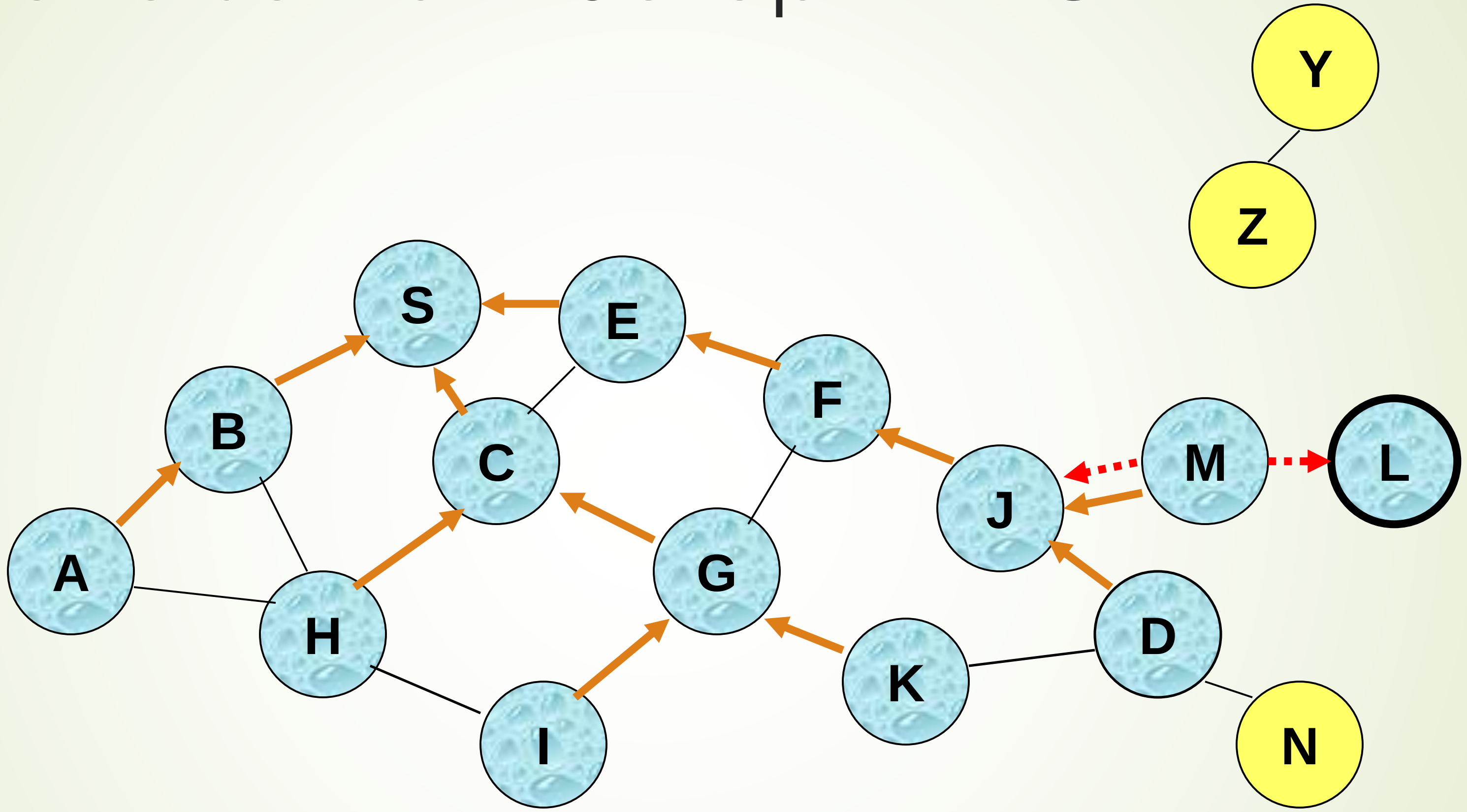


- Node C receives RREQ from G and H, but does not forward it again, because node C has **already forwarded RREQ** once

Reverse Path Setup in AODV

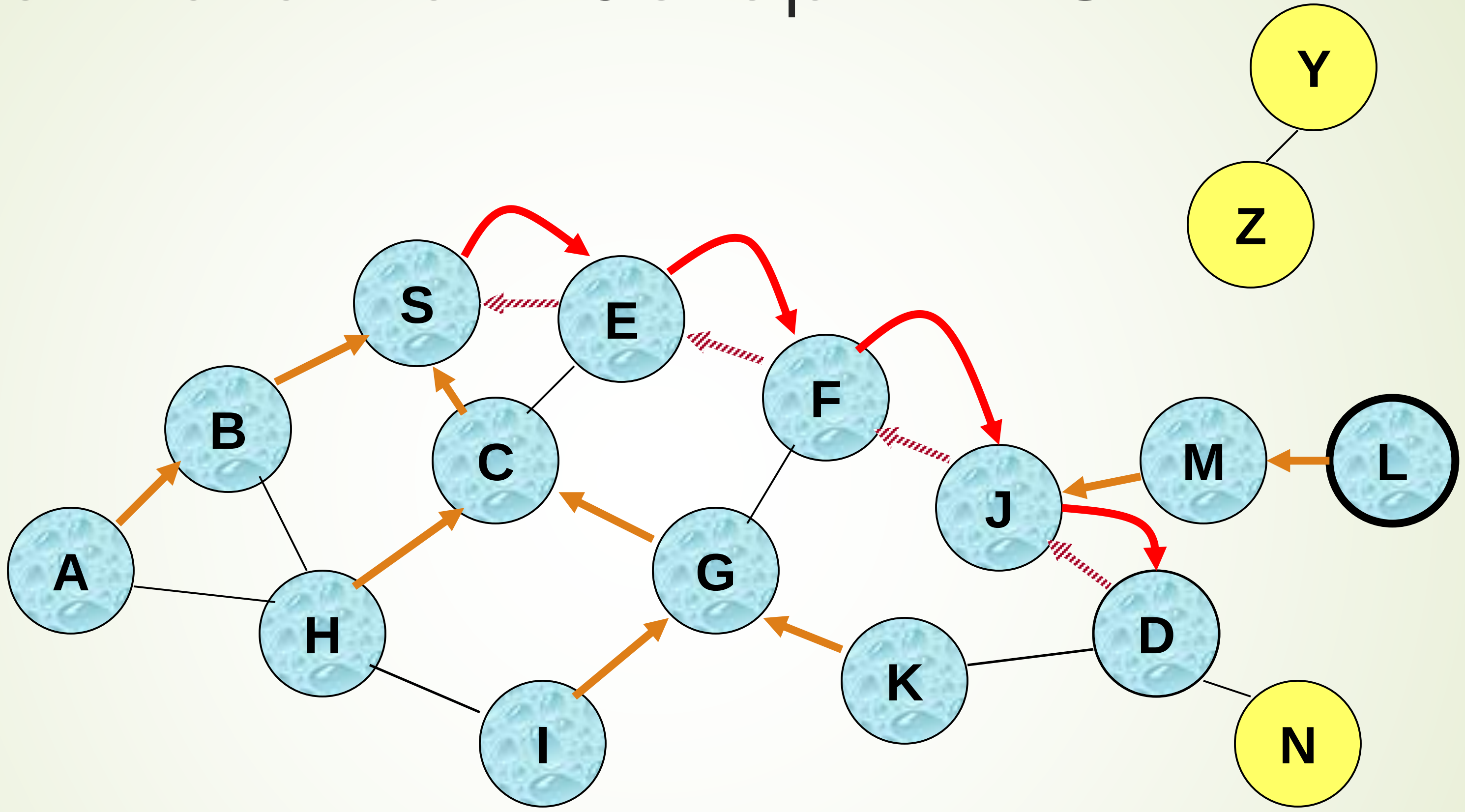


Reverse Path Setup in AODV



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

Forward Path Setup in AODV



Forward links are setup when RREP travels along the reverse path

Represents a link on the forward path

Proactive protocols

- Traditional distributed shortest-path protocols
- Maintain routes between every host pair at all times
- Based on periodic updates; High routing overhead
- Example: DSDV (destination sequenced distance vector)
- Optimized Link State Routing (OLSR)

Destination-Sequenced Distance-Vector Routing (DSDV)

- **Destination-Sequenced Distance-Vector Routing (DSDV)** is a table-driven routing scheme for ad hoc mobile networks based on the Bellman–Ford algorithm.
- It was developed by C. Perkins and P. Bhagwat in 1994. The main contribution of the algorithm was to solve the routing loop problem.
- Each entry in the routing table contains a sequence number, the sequence numbers are generally even if a link is present; else, an odd number is used.
- The number is generated by the destination, and the emitter needs to send out the next update with this number.
- Routing information is distributed between nodes by sending ***full dumps*** infrequently and smaller **incremental updates** more frequently.

Destination-Sequenced Distance-Vector Routing (DSDV)

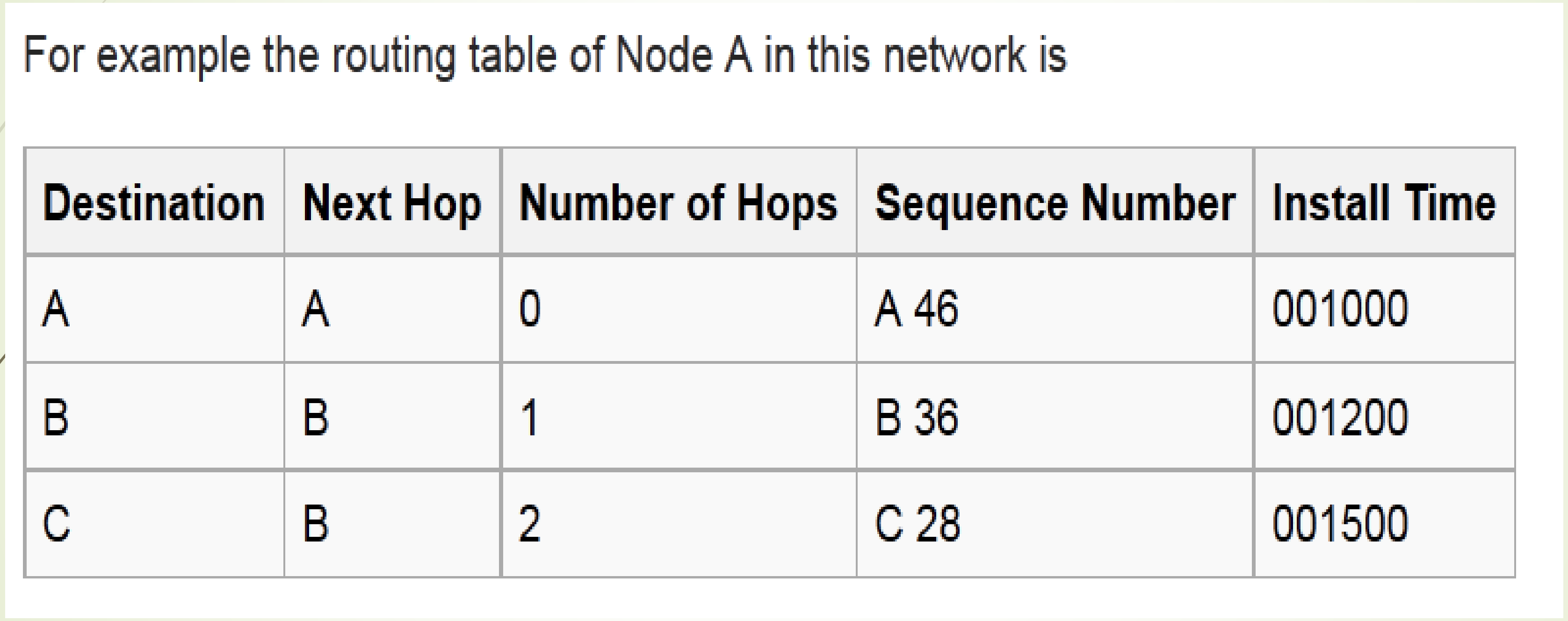
Full dump

Carries all available routing info and can require multiple network protocol data units.

Incremental packets

These smaller packets are for relaying only the information that was updated since the last full dump

Naturally the table contains description of all possible paths reachable by node A, along with the next hop, number of hops and sequence number.



For example the routing table of Node A in this network is

Destination	Next Hop	Number of Hops	Sequence Number	Install Time
A	A	0	A 46	001000
B	B	1	B 36	001200
C	B	2	C 28	001500

Selection of Route

- If a router receives new information, then it uses the latest sequence number. If the sequence number is the same as the one already in the table, the route with the better metric is used.

Advantages

- The availability of paths to all destinations in network always shows that less delay is required in the path set up process.
- The method of incremental update with sequence number labels, marks the existing wired network protocols adaptable to Ad-hoc wireless networks. Therefore, all available wired network protocol can be useful to ad hoc wireless networks with less modification.

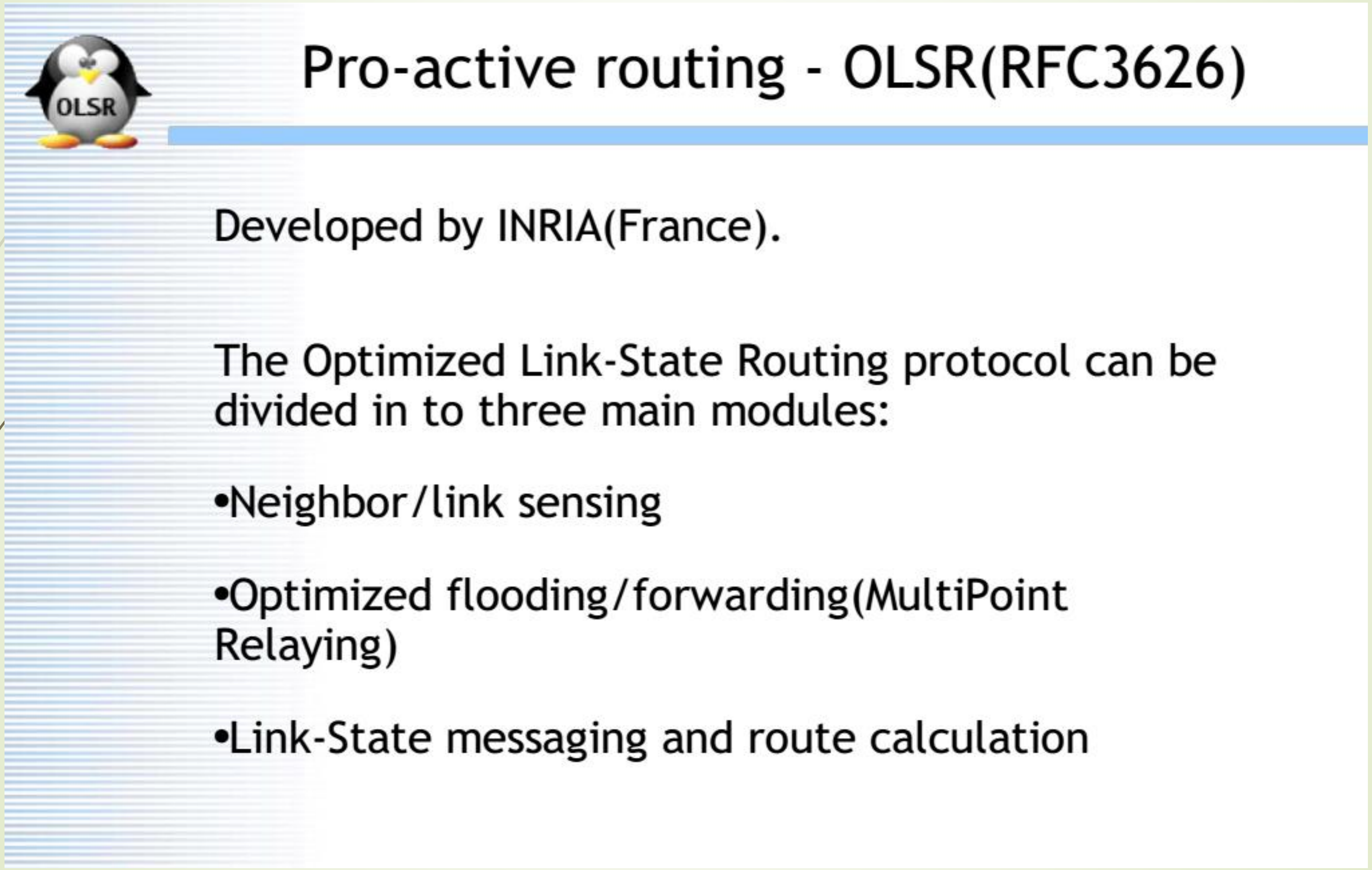
Disadvantages

- DSDV requires a regular update of its routing tables, which uses up battery power and a small amount of bandwidth even when the network is idle.
- Whenever the topology of the network changes, a new sequence number is necessary before the network re-converges; thus, DSDV is not suitable for highly dynamic or large scale networks.

Applications

- While DSDV itself does not appear to be much used today, other protocols have used similar techniques. The best-known sequenced distance vector protocol is AODV, which, by virtue of being a reactive protocol, can use simpler sequencing heuristics. Babel is an attempt at making DSDV more robust, more efficient and more widely applicable while staying within the framework of proactive protocols.

OLSR



Pro-active routing - OLSR(RFC3626)

Developed by INRIA(France).

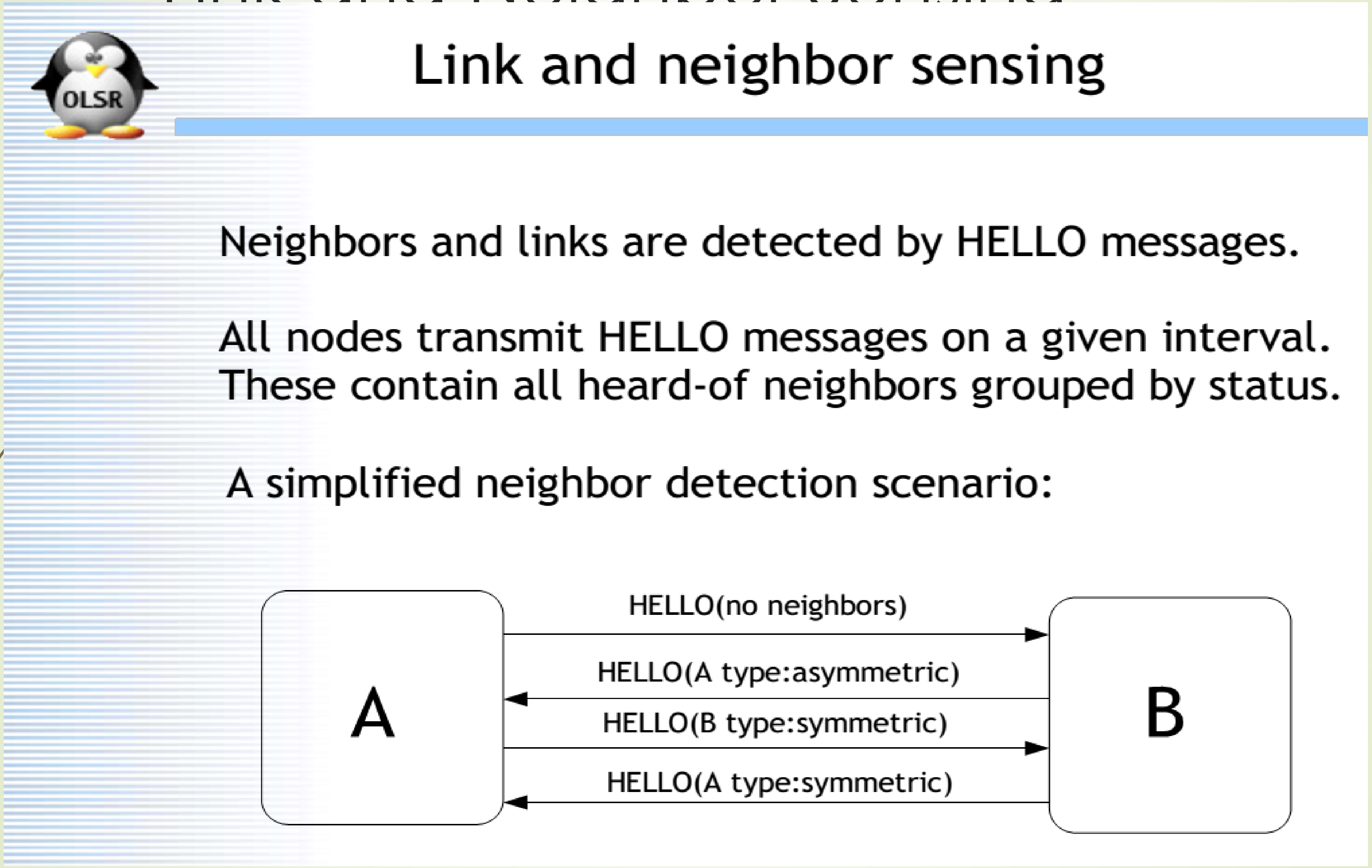
The Optimized Link-State Routing protocol can be divided in to three main modules:

•Neighbor/link sensing

•Optimized flooding/forwarding(MultiPoint Relaying)

•Link-State messaging and route calculation

Link and Neighbor Sensing

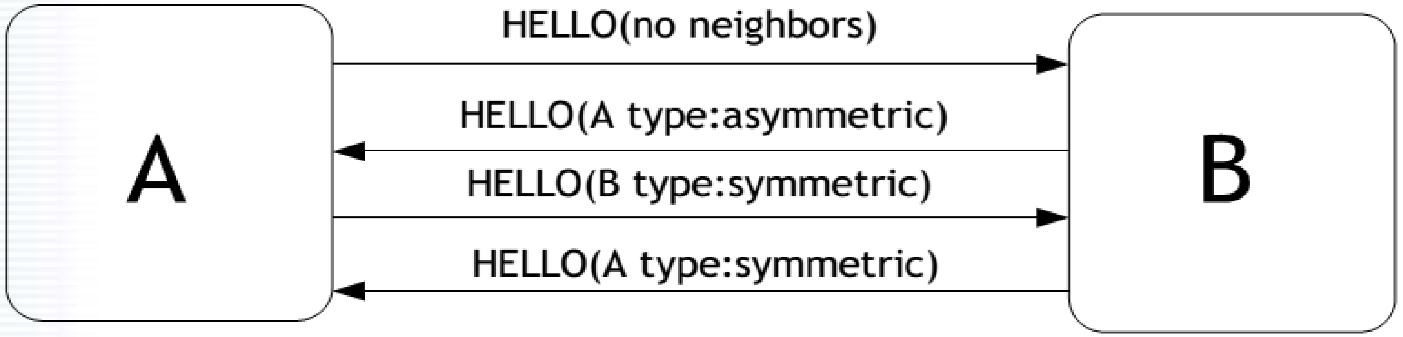


Link and neighbor sensing

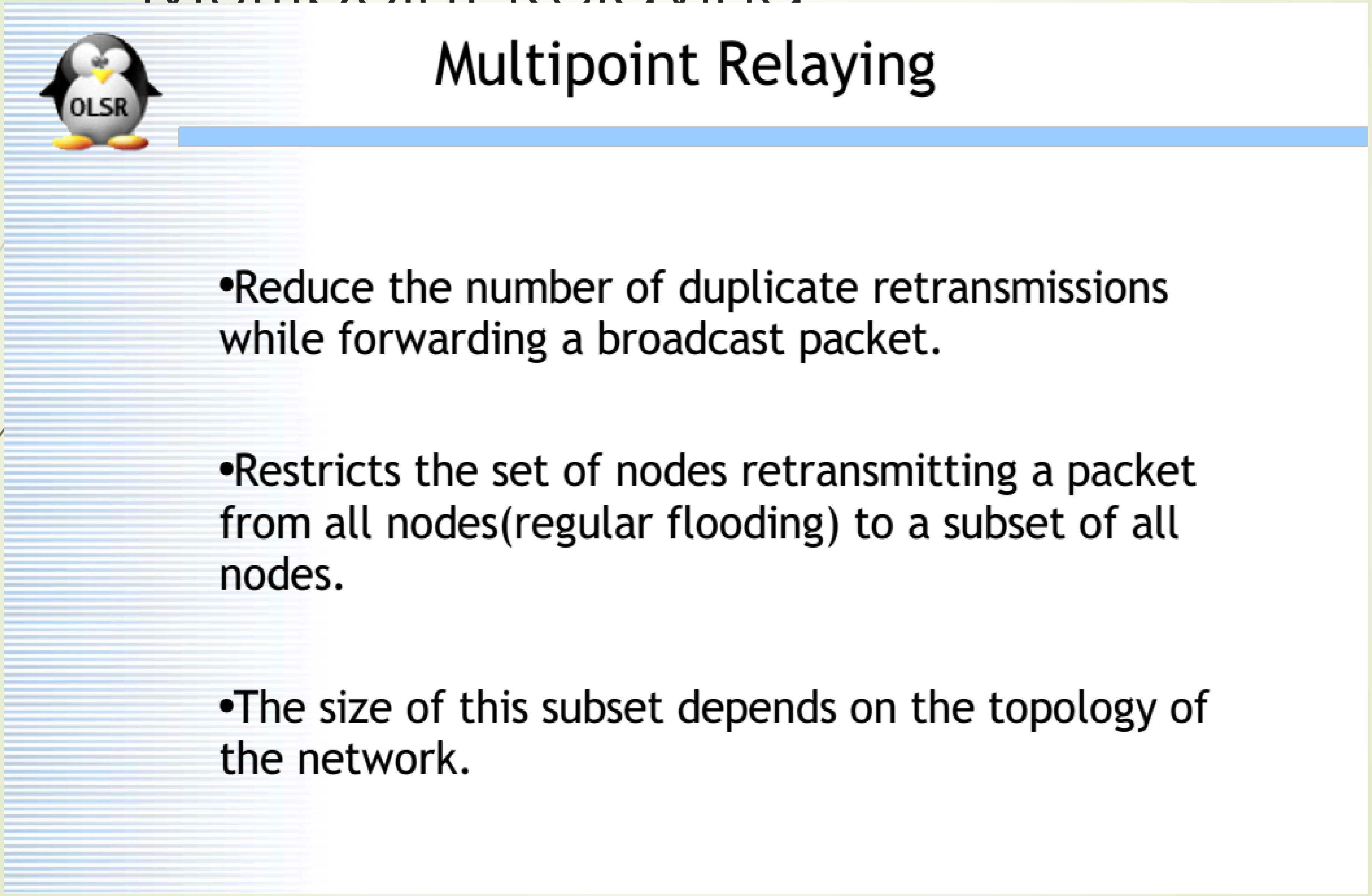
Neighbors and links are detected by HELLO messages.

All nodes transmit HELLO messages on a given interval. These contain all heard-of neighbors grouped by status.

A simplified neighbor detection scenario:



Multipoint Relaying



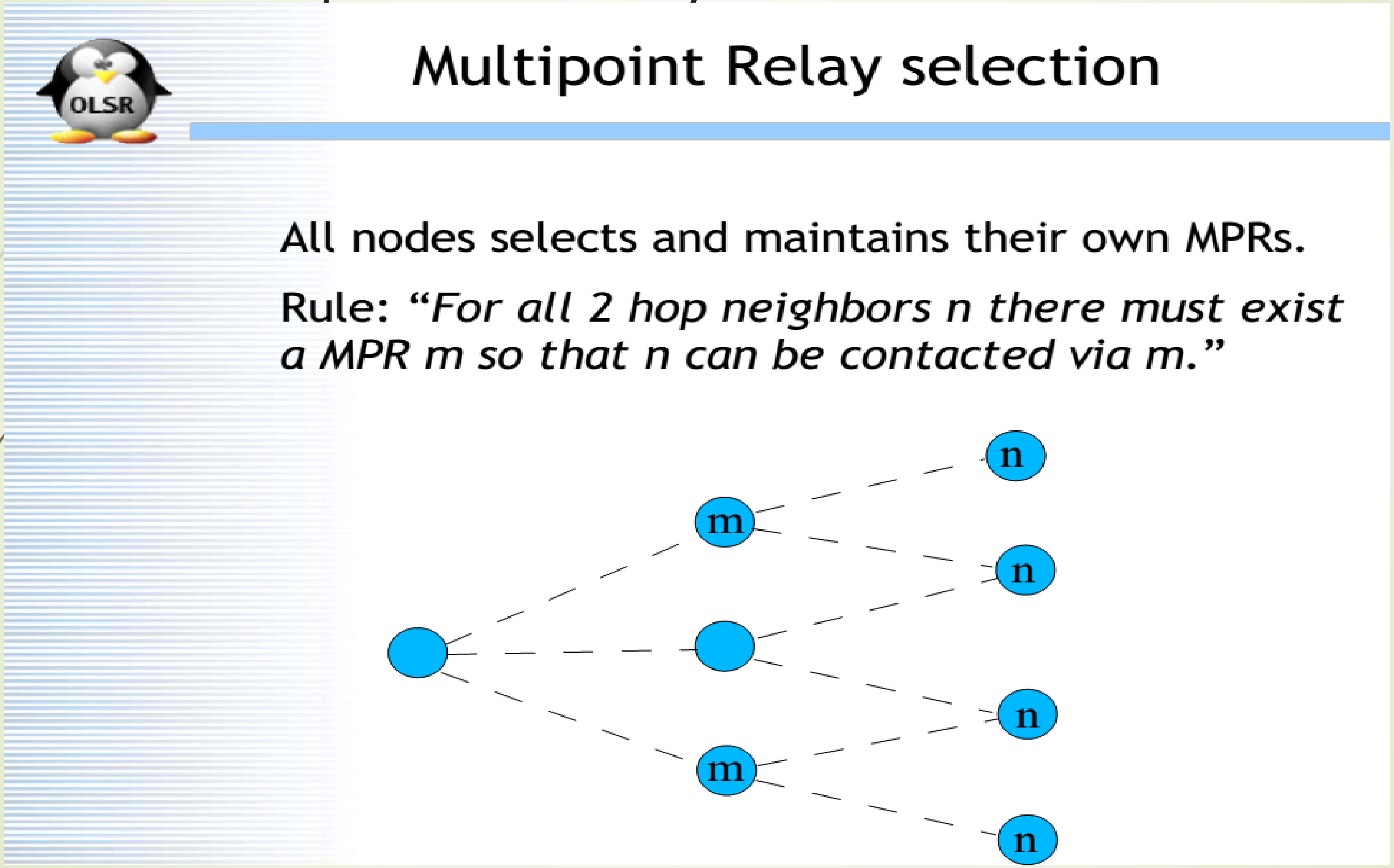
Multipoint Relaying

•Reduce the number of duplicate retransmissions while forwarding a broadcast packet.

•Restricts the set of nodes retransmitting a packet from all nodes(regular flooding) to a subset of all nodes.

•The size of this subset depends on the topology of the network.

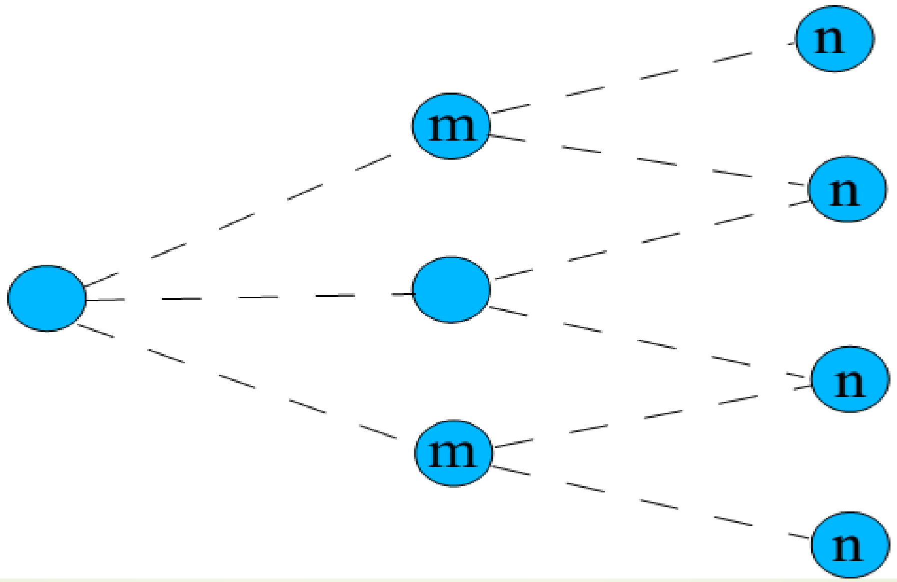
Multipoint Relay selection



Multipoint Relay selection

All nodes selects and maintains their own MPRs.

Rule: "*For all 2 hop neighbors n there must exist a MPR m so that n can be contacted via m.*"



Hybrid protocols

- Adaptive; Combination of proactive and reactive
- Example : ZRP (zone routing protocol)

The Zone Routing Protocol (ZRP)

The Zone Routing Protocol (ZRP)

- Invented by Zygmunt Haas of Cornell University.
- Combines both reactive and proactive schemes.
- Finds loop free routes to the destination.
- Flat structure (as opposed to hierarchical). Hierarchical schemes can lead to congestion localization.
- Link-State routing is possible – can enable QoS.

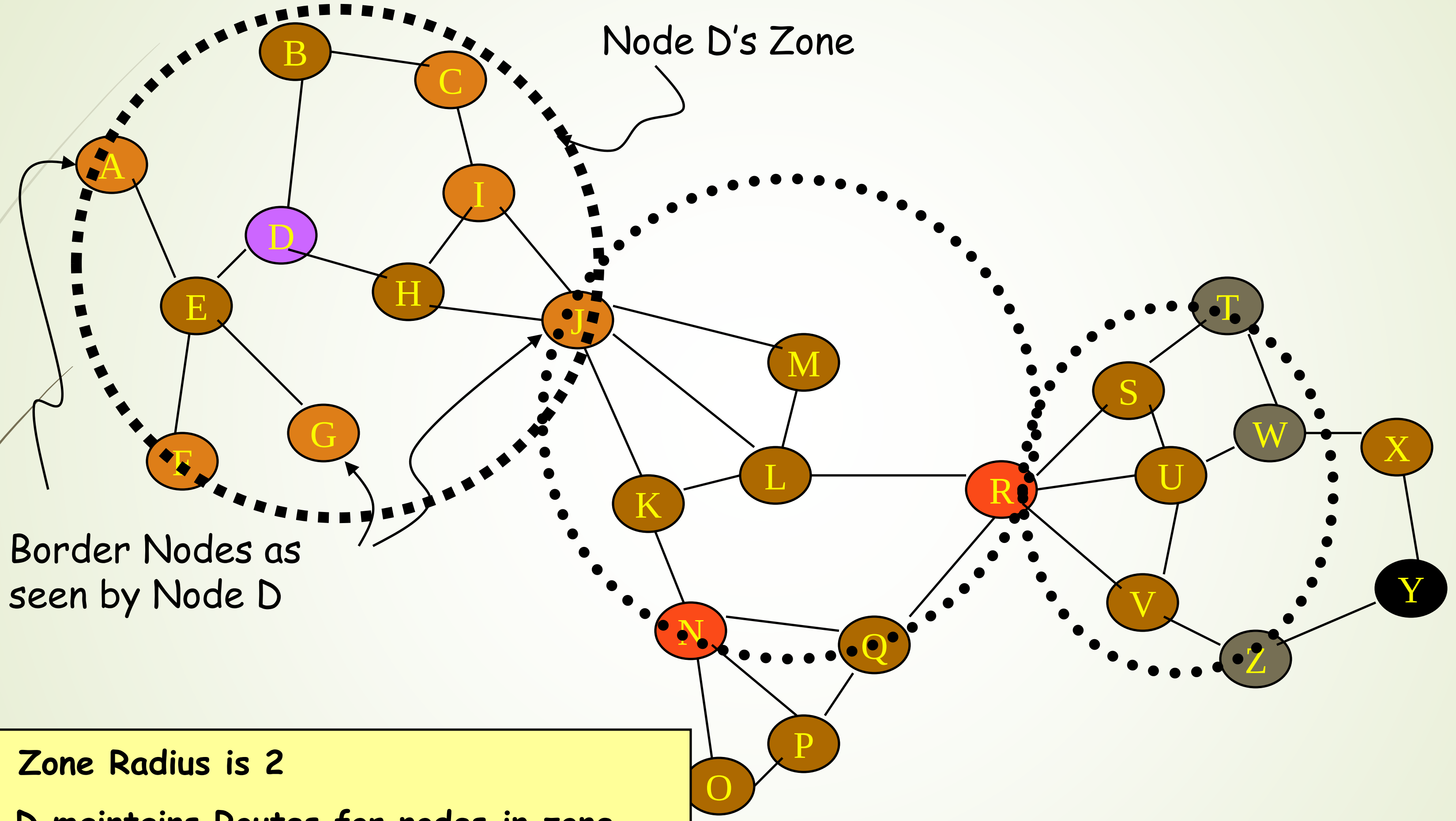
The concept of routing zones

- Each node, individually creates its own neighborhood which it calls a routing zone.
- The zone is defined as a collection of nodes whose minimum distance (in hops) from the node in question is no greater than a value that is called the “zone radius”.
- Peripheral nodes are those nodes whose minimum distance from the node in question is equal to the zone radius.

Intra-Zone Routing Protocol (IARP)

- Could be any link state or distance vector routing protocol.
- Maintained only within a zone.
- Notice the link-state maintenance gives the ability to have QoS.
- One question that arises is how large should the zone be ?
- If it is too large, then updates too much.
- If it is too small, often resort to re-active methods.

How does IARP work ?



- Zone Radius is 2
- D maintains Routes for nodes in zone --IARP e.g. D knows route to G.
- If node not found, resort to Inter zone search.

Neighbor Discovery Protocol (NDP)

- Note that IARP requires the presence of a Neighbor Discovery Protocol.
- Hello messages are required in order to ensure that neighbors are still present.
- This helps detect link failures.

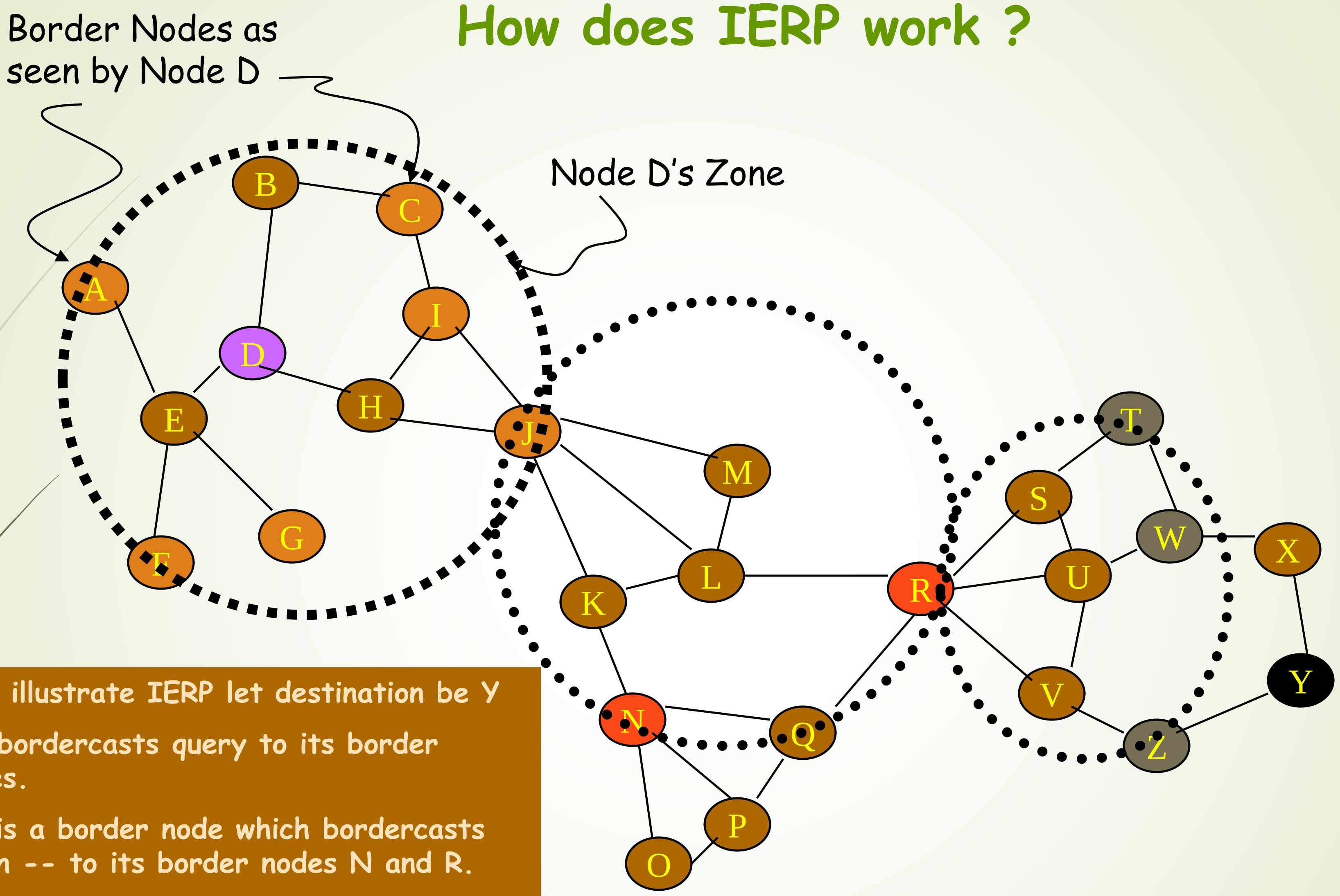
The Interzone routing protocol (IERP)

- If the IARP cannot find the destination, i.e., the destination is beyond a node's zone, the IERP is invoked.
- It is a reactive protocol that enables the discovery of the destination.
- Instead of doing a standard flood search, it exploits the structure of the routing zone to do more intelligent query dissemination.
- This is achieved by what is called “bordercasting”.

Bordercasting

- The node would direct the query message out only to its peripheral nodes.
- These nodes would execute the same algorithm that the primary node executed which is:
  - Check to see if the destination can be found within its zone. (How ?).
  - If yes, send a route-reply back to the source, indicating the route to the destination.
  - If not, forward the route-request to its peripheral nodes which execute the same procedure.

How does IERP work ?



- To illustrate IERP let destination be Y
- D bordercasts query to its border nodes.
- J is a border node which bordercasts again -- to its border nodes N and R.
- R bordercasts to its border nodes W,T and Z
- Y is found.

THANK
YOU