Internet of Things (IoT)

RPL: Routing over Low-Power and Lossy Networks

RFC 6550: https://tools.ietf.org/html/rfc6550

Dr. Manas Khatua
Associate Professor
Dept. of CSE, IIT Guwahati
E-mail: manaskhatua@iitg.ac.in

"The Man who works for others, without any selfish motive, really does good to himself." – **Shri Ramakrishna**

What is Low-Power and Lossy Network?

RFC 7228

- **Constrained Node**: A node where some of the characteristics that are otherwise pretty much taken for granted for Internet nodes are not attainable, often due to cost constraints and/or physical constraints on characteristics such as size, weight, and available power and energy.
  - tight limits on **power**, **memory**, and **processing resources**

- **Constrained Network**: A network where some of the characteristics pretty much taken for granted with link layers in common use in the Internet are not attainable.
  - low achievable **bitrate/throughput**; high **packet loss** and variability of packet loss; limits on reachability over **time**

- **Constrained-Node Network**: A network whose characteristics are influenced by being composed of a significant portion of constrained nodes.

- **LLN (Low-Power and Lossy Network):** Typically composed of many embedded devices with limited power, memory, and processing resources interconnected by a variety of links, such as IEEE 802.15.4 or low-power Wi-Fi.

Routing challenges in LLNs

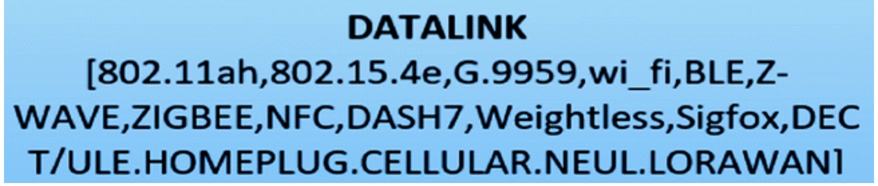
- Energy consumption is a major issue (for battery powered sensors/controllers)
- Limited processing power
- Very dynamic topologies
  - Link failure (as Low-powered RF)
  - Node failures (as fast energy depletion)
  - Node mobility (in some environments)
- Data processing usually required on the node itself,
- Sometimes deployed in harsh environments (e.g. Industrial, hilly region),
- Potentially deployed at very large scale,
- Must be self-managed network (auto-discovery, self-organizing,)

Can’t use OSPF, OLSR, RIP, AODV, DSDV, DSR, etc

Routing over Low-power and Lossy link: ROLL WG

- ROLL Working Group Formed in Jan 2008

- **Mission**: define Routing Solutions for LLN
  - ➢ Should be able to operate over a variety of different link layer technologies



DATALINK

[802.11ah,802.15.4e,G.9959,wi_fi,BLE,Z-WAVE,ZIGBEE,NFC,DASH7,Weightless,Sigfox,DECT/ULE.HOMEPLUG.CELLULAR.NEUL.LORAWAN]

- **Work Items**:
  - ▪ Routing Protocol work
  - ▪ Routing is designed to support different LLN application requirements
    - ▪ RFC 5548 - Routing requirements for Urban LLNs
    - ▪ RFC 5673 - Routing requirements for Industrial LLNs
    - ▪ RFC 5826 - Routing requirements for Home Automation LLNs
    - ▪ RFC 5867 - Routing requirements for Building Automation LLNs
  - ▪ Routing metrics for LLN
  - ▪ Produce a security Framework
  - ▪ Applicability statement of ROLL routing protocols

- Proposed protocol: **RPL (IPv6 Routing Protocol for LLNs)**

RPL is a ……

- Distance Vector (DV) protocol
- Source Routing Protocol

What is a Distance Vector (DV) protocol?

- The term distance vector refers -
  - protocol manipulates vectors of distances to all other nodes in network
- It is based on calculating the Direction and Distance to any node in a network.
  - "**Direction**" usually means the next hop address and the exit interface.
  - "**Distance**" is a measure of the cost to reach a certain node.
- Least cost route = route with minimum distance.
- Each node maintains a vector (table) of minimum distance to every node.
- Router shares its knowledge about the whole network to its neighbours periodically.
- It is an Intra-domain routing protocol (i.e. inside a AS)
- Have less computational complexity and message overhead

Cont…

What is a Source Routing (path addressing) protocol?

- Allows a sender of a packet to partially or completely specify the route the packet takes through the network.
- Enables a node to discover all the possible routes to a host.

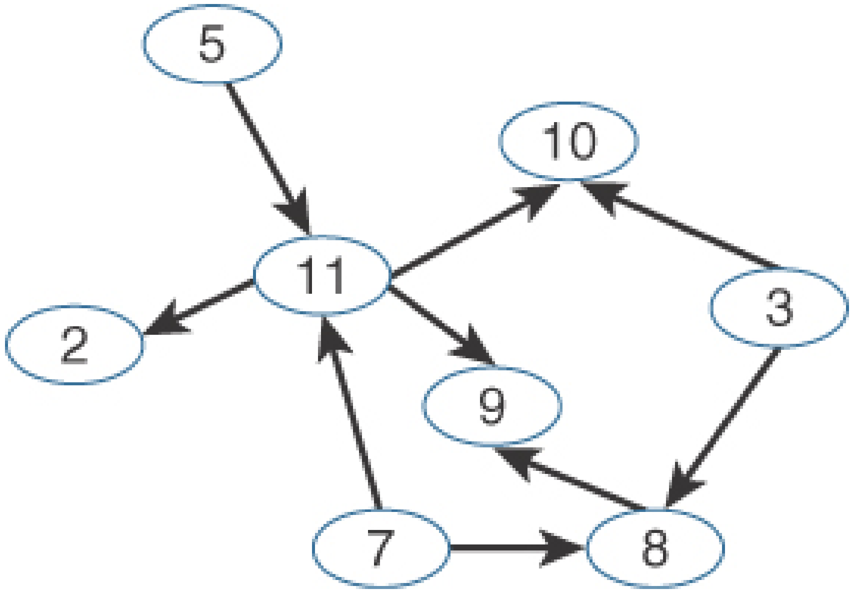
Two modes of RPL:

- **Storing mode:**
  - All nodes contain the full routing table of the RPL domain.
  - Every node knows how to directly reach every other node.

- **Non-storing mode:**
  - Only the border router(s) of the RPL domain contain(s) the full routing table.
  - Boarder router knows how to directly reach every other node.

RPL Topology (1/2)

RPL organizes a topology as a DAG



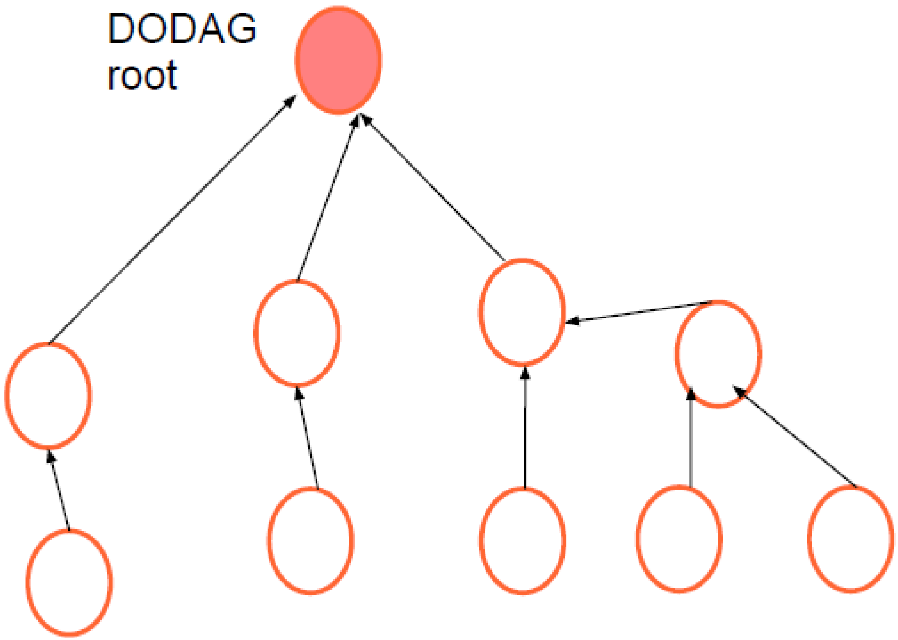
DAG(Directed Acyclic Graph)

- A DAG is a directed graph where no cycles exist.

RPL Topology (2/2)

- A DAG rooted at a single destination at a single DAG root (DODAG root) with no outgoing edges

- A basic RPL process involves building a DODAG.
- In RPL, this destination occurs at a border router known as the DODAG root.

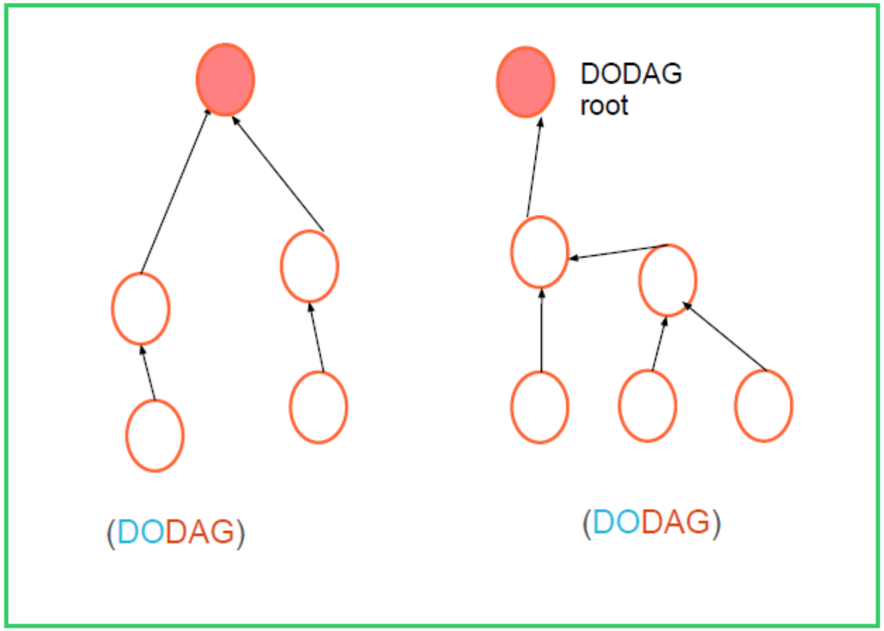


- Simplest RPL topology: single DODAG with one root
- Complex scenario: multiple uncoordinated DODAGs with independent roots
- More sophisticated and flexible configuration: single DODAG with a virtual root that coordinates several LLN root nodes

DODAG (Destination Oriented DAG)

RPL Instance

- An RPL Instance is a set of one or more DODAGs that **share a common RPLInstanceID**



RPL Instance

- **RPLInstanceID** is a **unique** identifier within a network.
- DODAGs with the same RPLInstanceID share the same Objective Function (OF)
  - used to compute the position of node in the DODAG .
- An **objective function** (OF) defines
  - how metrics are used to select **routes** and establish a node's **rank**.
  - RFC 6552 and RFC 6719
- Objective Function computes the "**rank**"
  - rank is the "distance" between the node and DODAG root
  - *Rank* should monotonically decrease along the DODAG and towards the destination

RPL Control Messages

The RPL Control Message consists of an ICMPv6 header followed by a message body.

1) DODAG Information Solicitation (**DIS**):
   - Link-Local multicast request for DIO (i.e. neighbour discovery).
   - Do you know of any DODAGs, asked by a node?

2) DODAG Information Object (**DIO**):
   - Downward RPL instance multicasts
   - Allows other nodes to discover an RPL instance and join it

3) Destination Advertisement Object (**DAO**):
   - From child to parents or root
   - Can I join you as a child on DODAG #x?

4) DAO-ACK: Yes, you can! Or Sorry, you can't!

5) Consistency Check (**CC**): Challenge-response messages for security

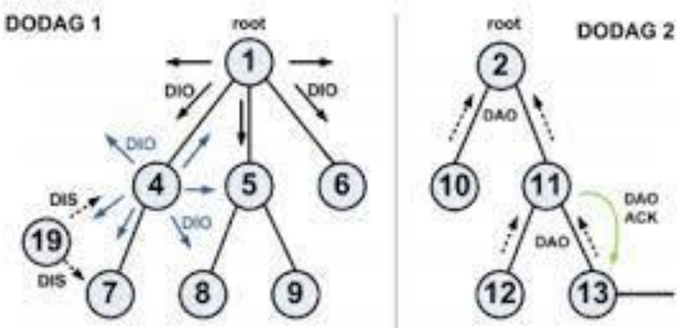
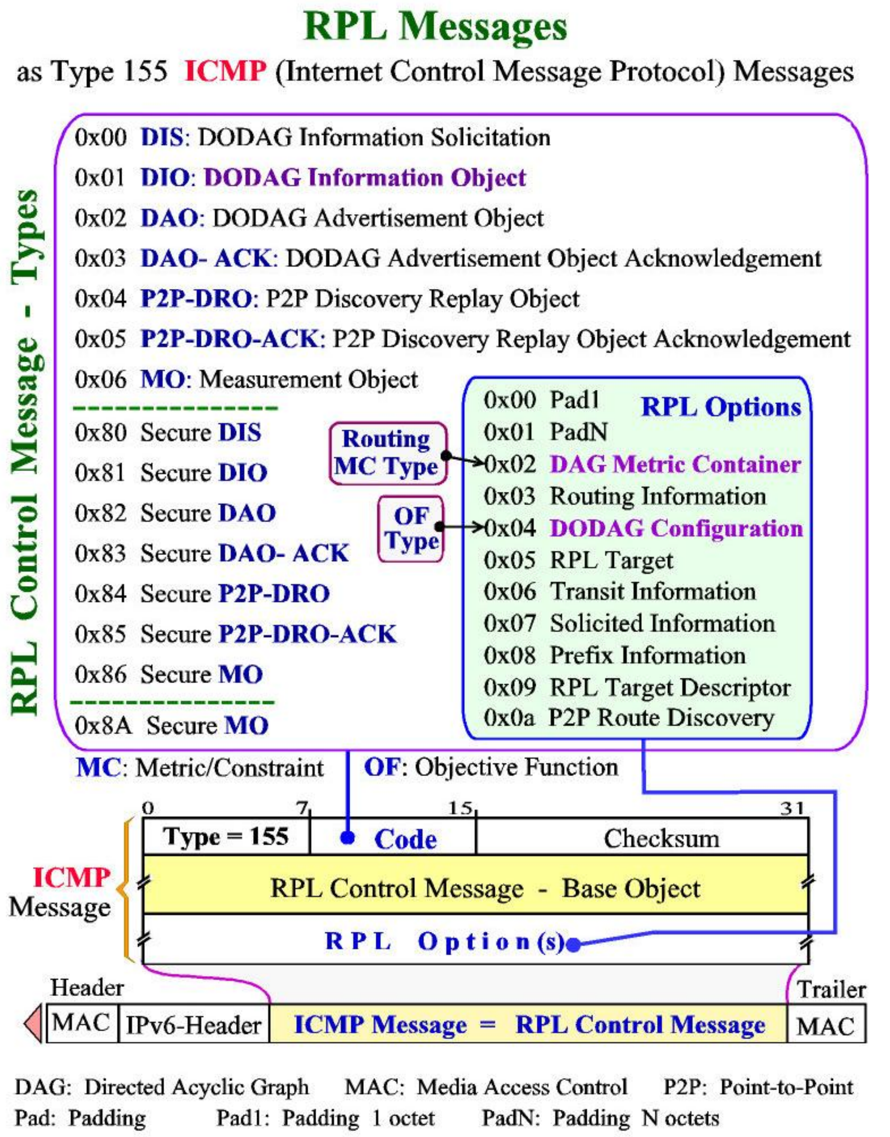


Fig. 1. RPL Control Messages

Source: https://ws.engr.illinois.edu/sitemanager/getfile.asp?id=1560

RPL Control Messages



RPL Messages

as Type 155 ICMP (Internet Control Message Protocol) Messages

RPL Control Message - Types

0x00	**DIS**: DODAG Information Solicitation
0x01	**DIO**: **DODAG Information Object**
0x02	**DAO**: DODAG Advertisement Object
0x03	**DAO- ACK**: DODAG Advertisement Object Acknowledgement
0x04	**P2P-DRO**: P2P Discovery Replay Object
0x05	**P2P-DRO-ACK**: P2P Discovery Replay Object Acknowledgement
0x06	**MO**: Measurement Object
0x80	Secure **DIS**
0x81	Secure **DIO**
0x82	Secure **DAO**
0x83	Secure **DAO- ACK**
0x84	Secure **P2P-DRO**
0x85	Secure **P2P-DRO-ACK**
0x86	Secure **MO**
0x8A	Secure **MO**

RPL Options

0x00	Pad1
0x01	PadN
0x02	**DAG Metric Container**
0x03	Routing Information
0x04	**DODAG Configuration**
0x05	RPL Target
0x06	Transit Information
0x07	Solicited Information
0x08	Prefix Information
0x09	RPL Target Descriptor
0x0a	P2P Route Discovery

MC: Metric/Constraint

OF: Objective Function

DAG: Directed Acyclic Graph MAC: Media Access Control P2P: Point-to-Point

Pad: Padding Pad1: Padding 1 octet PadN: Padding N octets

Source:
https://www.researchgate.net/publication/326960497_RPL_messages_and_their_structure

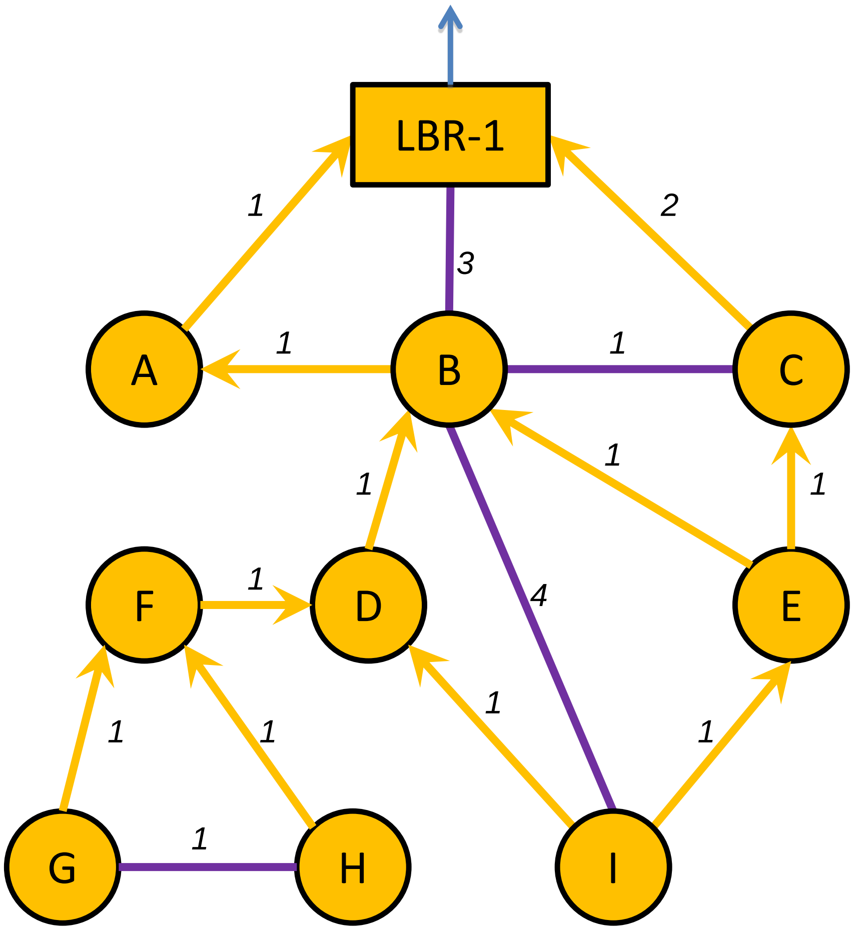
RPL Traffic Types

1) MP2P : *Multipoint-to-Point*
   - It is the dominant traffic in many LLN applications.
   - usually routed towards destination nodes such as LLN gateway
   - these destinations are the DODAG roots, and they act mainly as data collection points

2) P2MP: *Point-to-Multipoint*
   - data streams can be used mainly for actuation purposes
   - messages sent from DODAG roots to destination nodes

3) P2P: *Point-to-Point*
   - to allow communications between two devices belonging to the same LLN

(1) MP2P Traffic

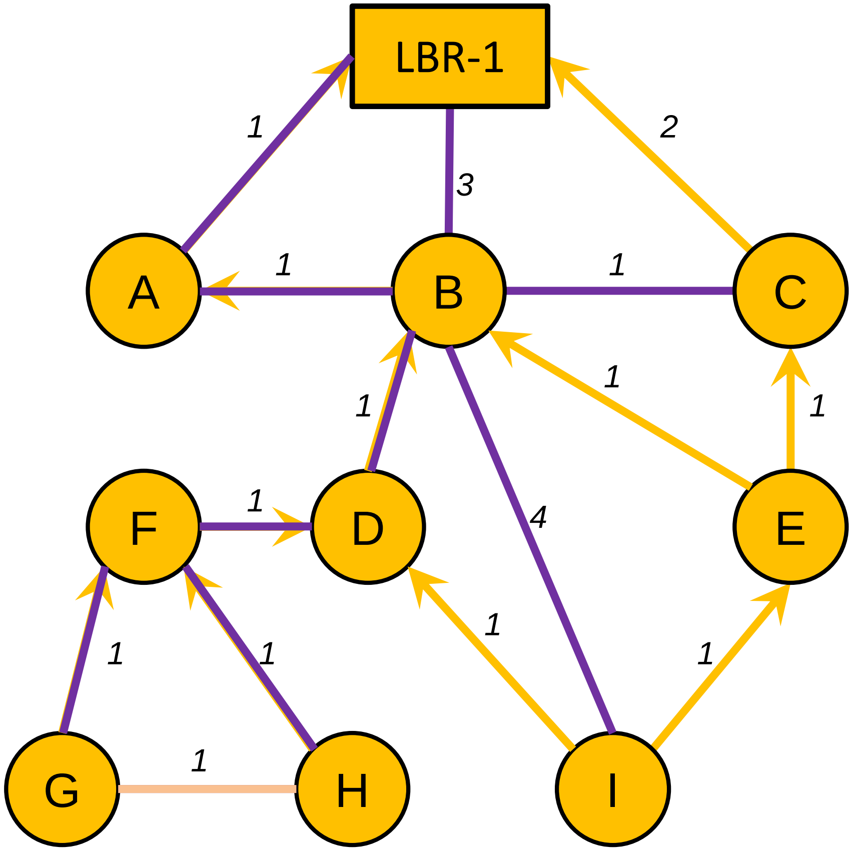


- MP2P traffic **flows inwards** along DAG, toward DAG Root

- DAG Root may also extend connectivity to other prefixes beyond the DAG root, as specified in the DIO

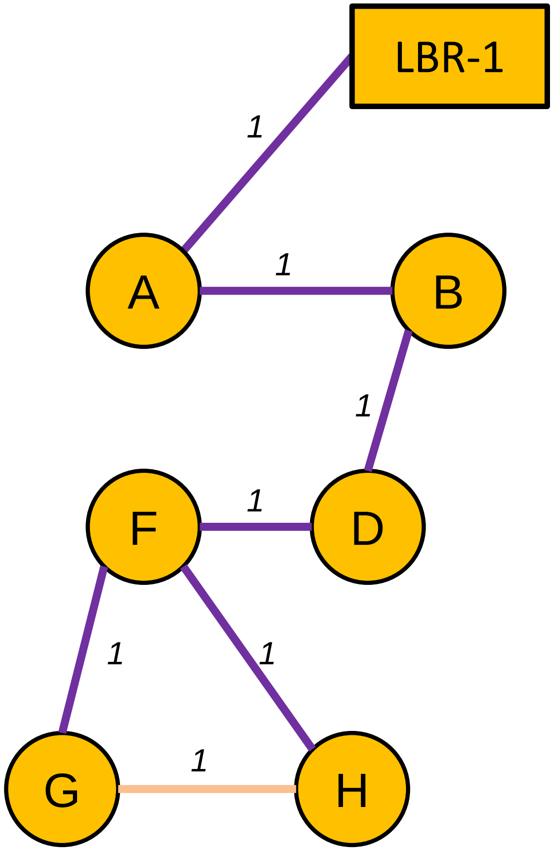
- Nodes may join multiple DAGs as necessary to satisfy application constraints

Destination Advertisements (1/7)



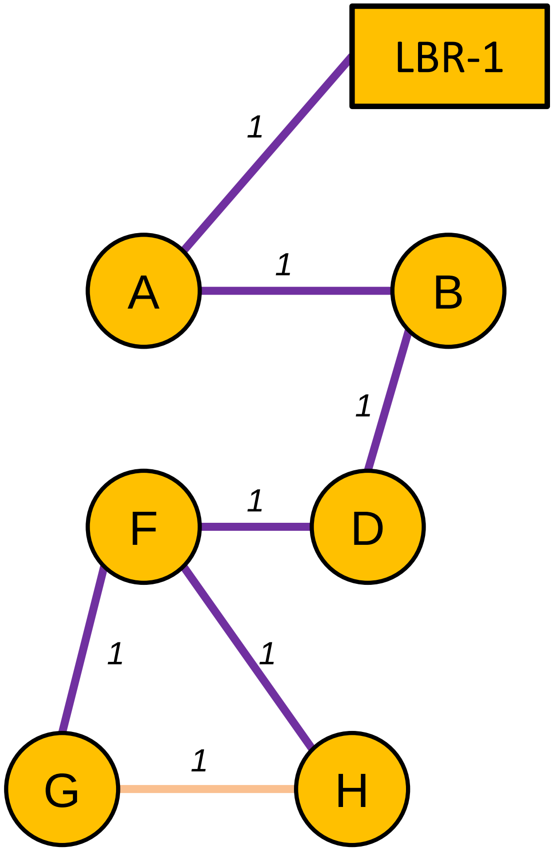
- Destination Advertisements (DA) build up **routing state**
  - to support P2MP traffic **flows outward**, from the sink to other nodes
- DA uses the same DAG
- For simplicity, we will focus on a subset of DA in the example

Destination Advertisements (2/7)



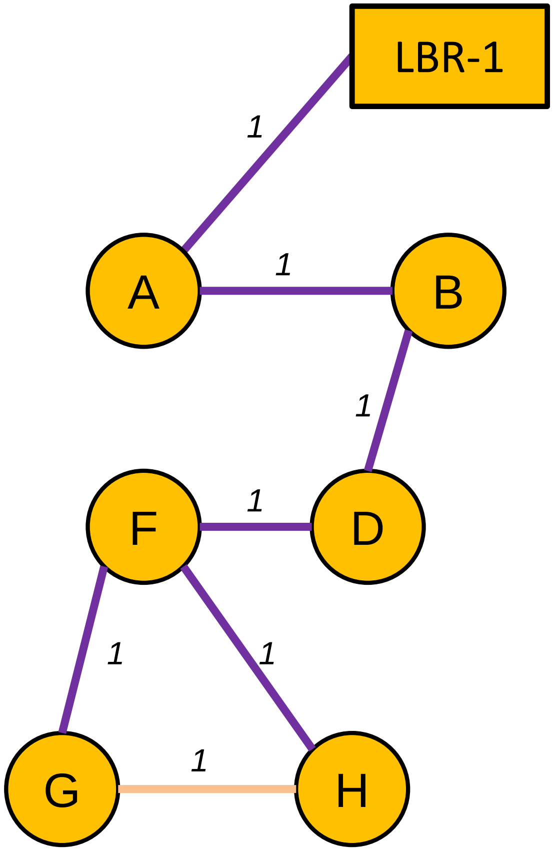
- Let us consider,
  - Some nodes may be able to store routing state for outward flows (LBR-1, A, F)
  - Some nodes may not (B, D)
  - Some nodes may have a limited ability;
- DAs may indicate a priority for storage
- DAs may be triggered by DAG root or node who detects a change
- DA timers configured such that DAs start at greater depth, and may aggregate as they move up

Destination Advertisements (3/7)



- LBR-1 triggers DA mechanism in DIO
- G emits neighbor advertisement (NA) to F with DAO
  - indicating reachability to destination prefix G::
- F stores G:: via G
- H emits NA to F for destination prefix H::
- F stores H:: via H

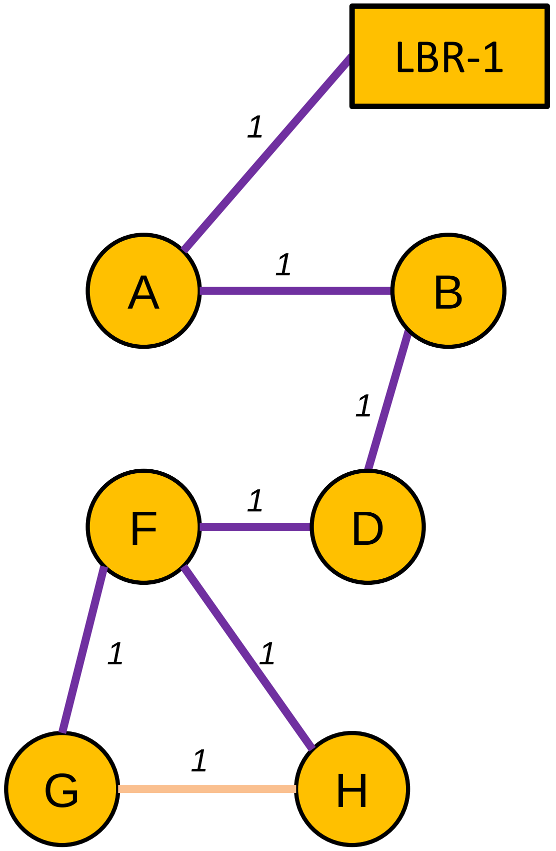
Destination Advertisements (4/7)



- Suppose in this example F has a prefix F*:: capable of aggregating {F::, G::, H::}
  - The method to provision such a prefix is beyond the scope of RPL
- F emits NA to D with DAO indicating reachability to destination prefix F*::
- D cannot store…

(continued)

Destination Advertisements (5/7)



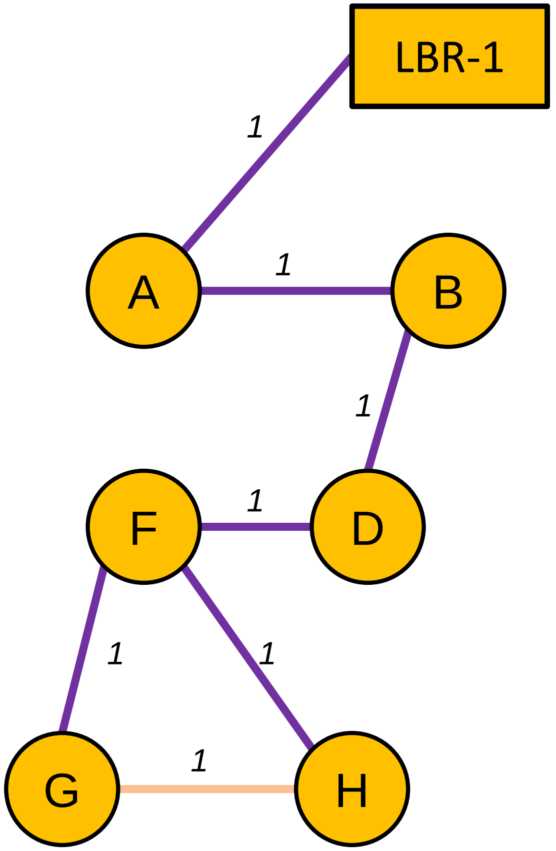
- D adds F to the Reverse Route Stack in the DAO, and passes DAO on to B for F*:: [F]

- D also emits a DAO indicating prefix D:: to B

- B cannot store routing state…

(continued)

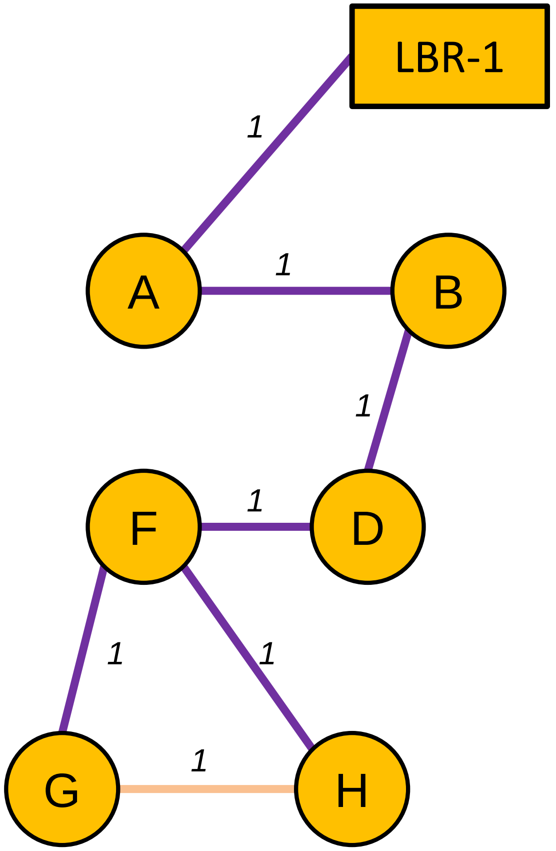
Destination Advertisements (6/7)



- B adds D to the Reverse Route Stack in the DAO for D::, and passes DAO D:: [D] on to A
- A stores D:: via B, with the piecewise source route [D]
- B also emits a DAO indicating prefix B:: to A
- A stores B:: via B
- A also stores F*:: via B, with the source root [D,F]

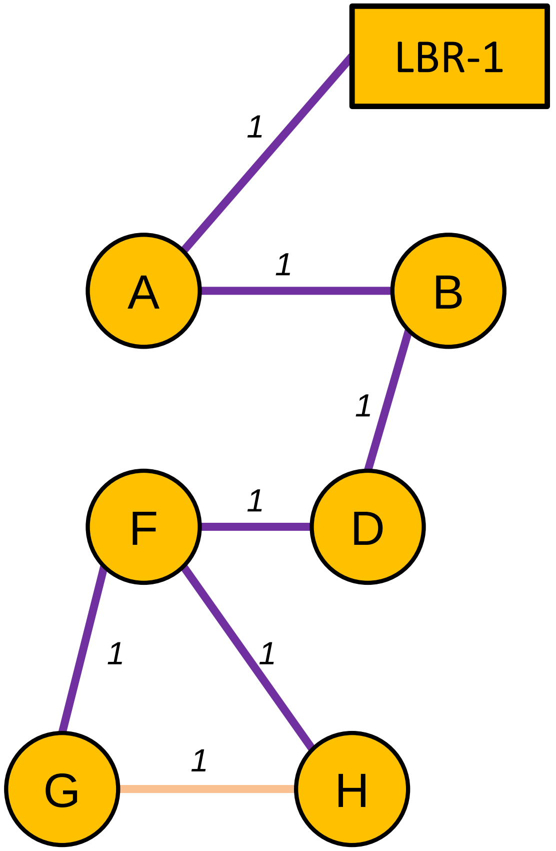
(continued)

Destination Advertisements (7/7)



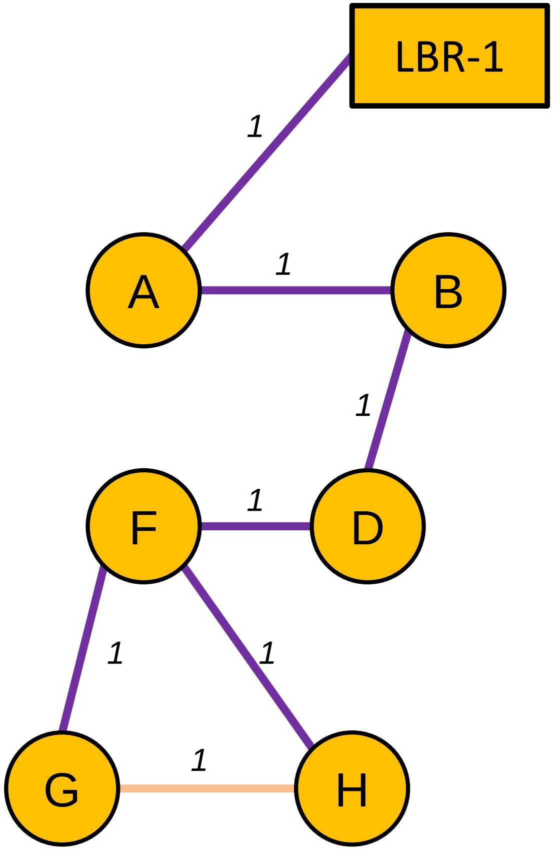
- A emits DAOs to LBR-1 for destination prefixes A::, B::, D::, and F*
- LBR-1 stores A:: via A, B:: via A, D:: via A, and F*:: via A
- It is done. So, in brief,
  - LBR-1 stores A:: via A, B:: via A, D:: via A, and F*:: via A
  - A stored B:: via B, D:: via B [D] , F* via B [D,F]
  - B, D stored nothing
  - F stored G:: via G, H:: via H

(2) P2MP Traffic (1/2)



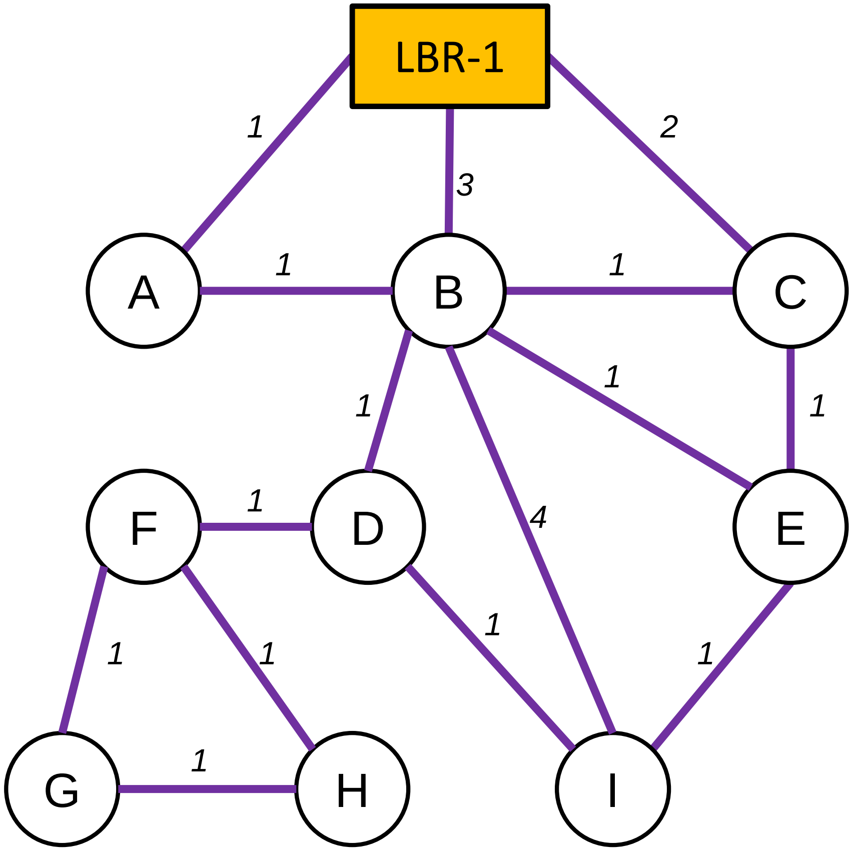
- The routing state setup by Destination Advertisement (DA) is used to direct P2MP traffic outward
- LBR-1 directs traffic for G (F*::) to A
- A adds source routing directive, [D, F], and forwards to B
- B uses source routing directive to forward to D...

P2MP Traffic (2/2)



- D uses source routing directive to forward to F
- F uses routing state to forward to G
- **Note** the use of source routing to traverse the stateless region of the LLN

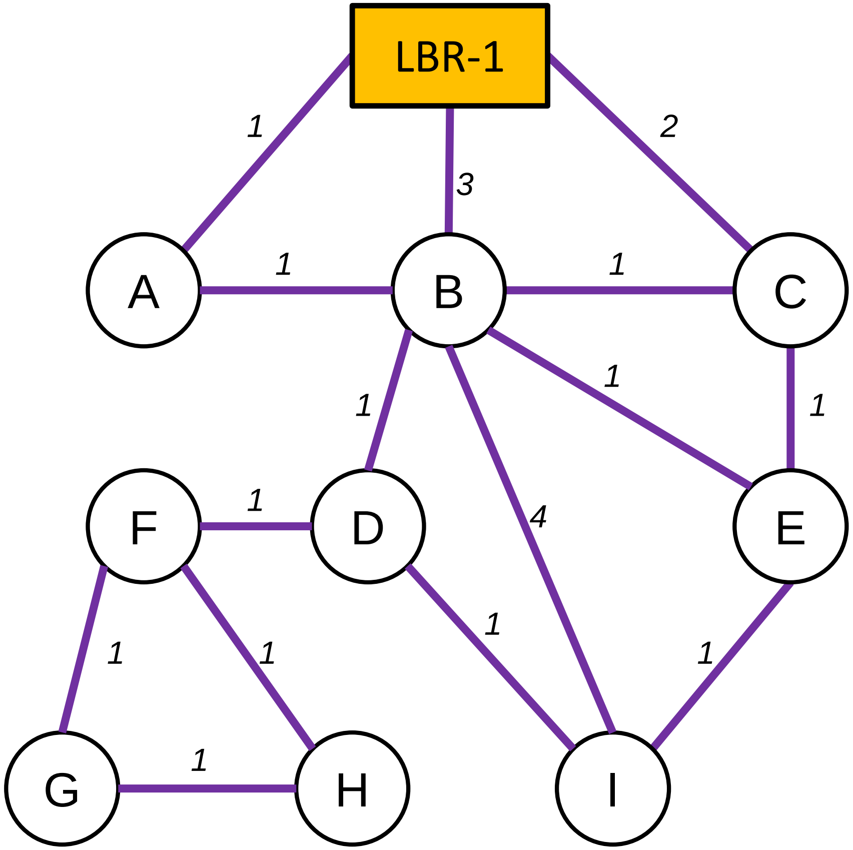
DAG Construction (1/9)



- LLN links are depicted
- RPL Objective functions:
  - ETX https://tools.ietf.org/html/draft-gnawali-roll-etxof-00
  - OF0 https://tools.ietf.org/id/draft-ietf-roll-of0-14.html
- Links are annotated w/ ETX
- It is expected that ETX variations will be averaged/filtered as per ROLL Metrics to be stable enough for route computation
  - Nodes observe the metric and gain confidence before use

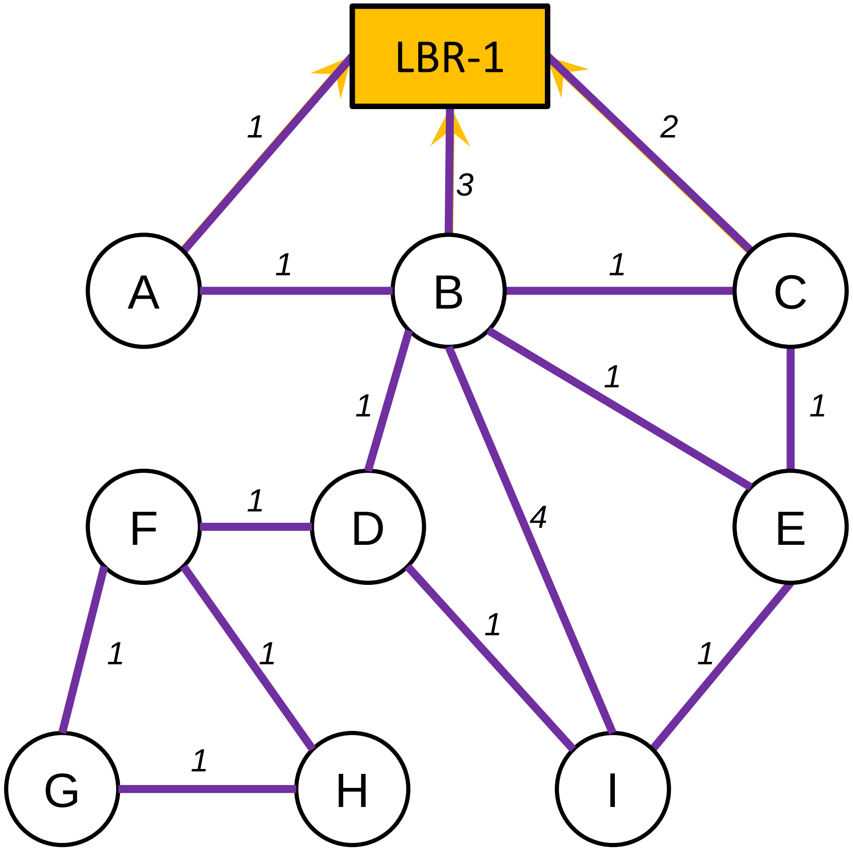
The ETX metric of a wireless link is the expected number of transmissions required to successfully transmit and acknowledge a packet on the link.

DAG Construction (2/9)



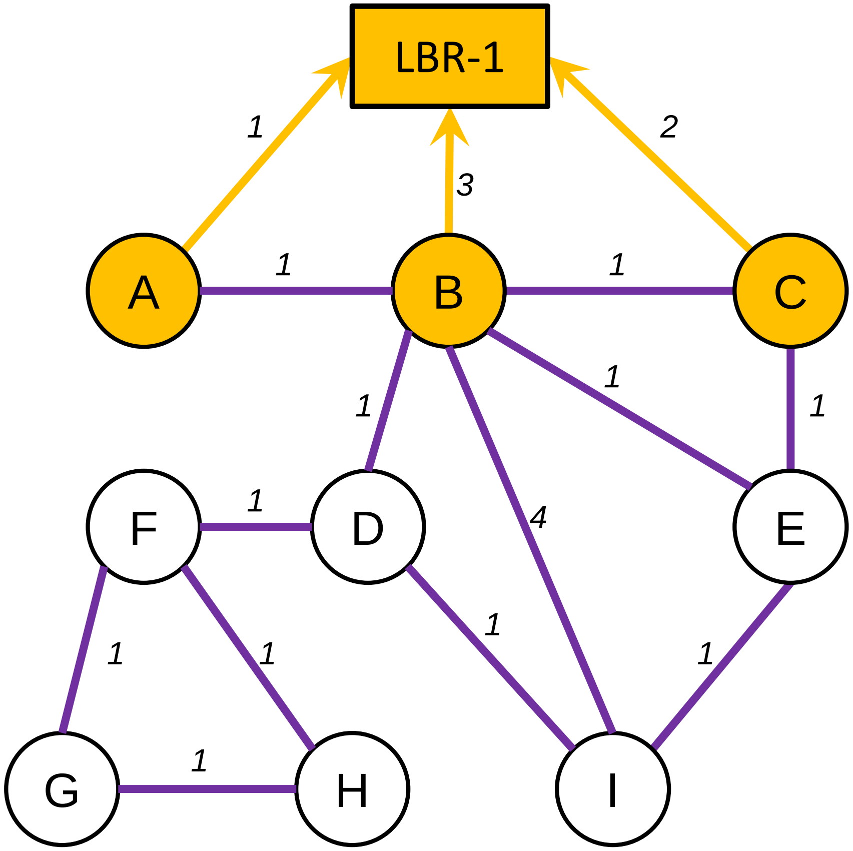
- Objective Code Point (OCP) for example
  - Metric: ETX
  - Objective: Minimize ETX
  - Depth computation: Depth ~ ETX
    - Note that a practical computation may be more coarse

DAG Construction (3/9)



- LBR-1 multicasts RA-DIO (i.e. router advertisement using DIO)
- Nodes A, B, C receive and process RA-DIO
- Nodes A, B, C consider link metrics to LBR-1 and the optimization objective
- The optimization objective can be satisfied by joining the DAG rooted at LBR-1
- Nodes A, B, C add LBR-1 as a DAG parent and join the DAG

DAG Construction (4/9)



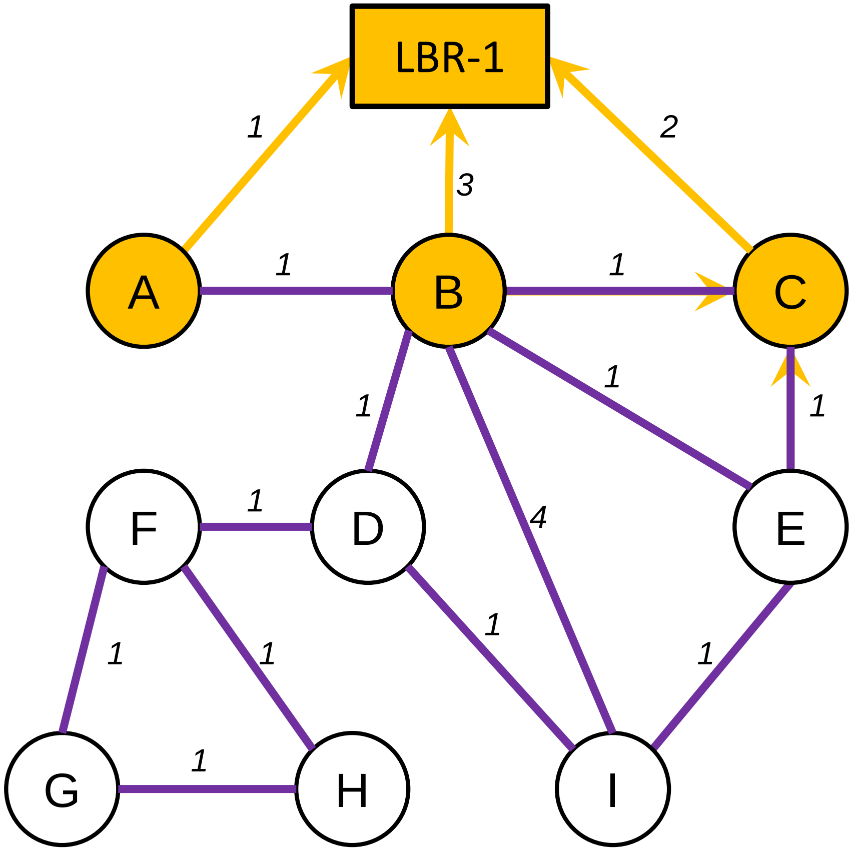
- Node A is at Depth 1 in the DAG, as calculated by the routine indicated by the example OCP (Depth ~ ETX)

- Node B is at Depth 3, Node C is at Depth 2

- Nodes A, B, C have installed default routes (::/0) with LBR-1 as successor

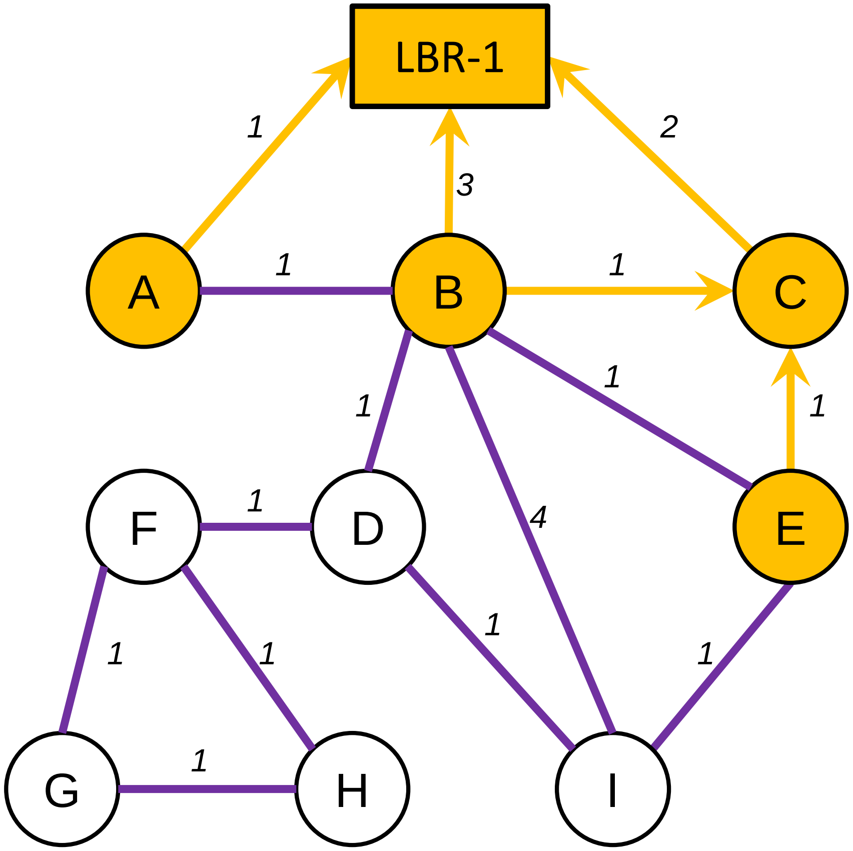
- **Note**: An arrow shows who is your parent. But, the links are bidirectional.

DAG Construction (5/9)



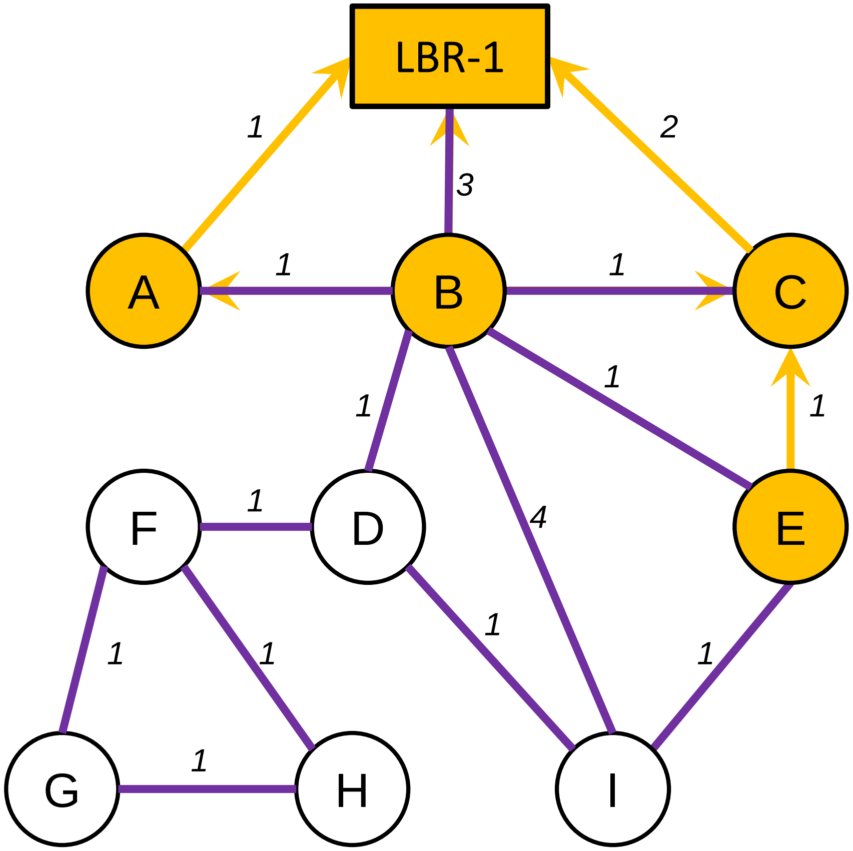
- The RA timer on Node C expires
- Node C multicasts RA-DIO
- LBR-1 ignores RA-DIO from deeper node
- Node B can add Node C as *alternate* DAG Parent, remaining at Depth 3
- Node E joins the DAG at Depth 3 by adding Node C as DAG Parent

DAG Construction (6/9)



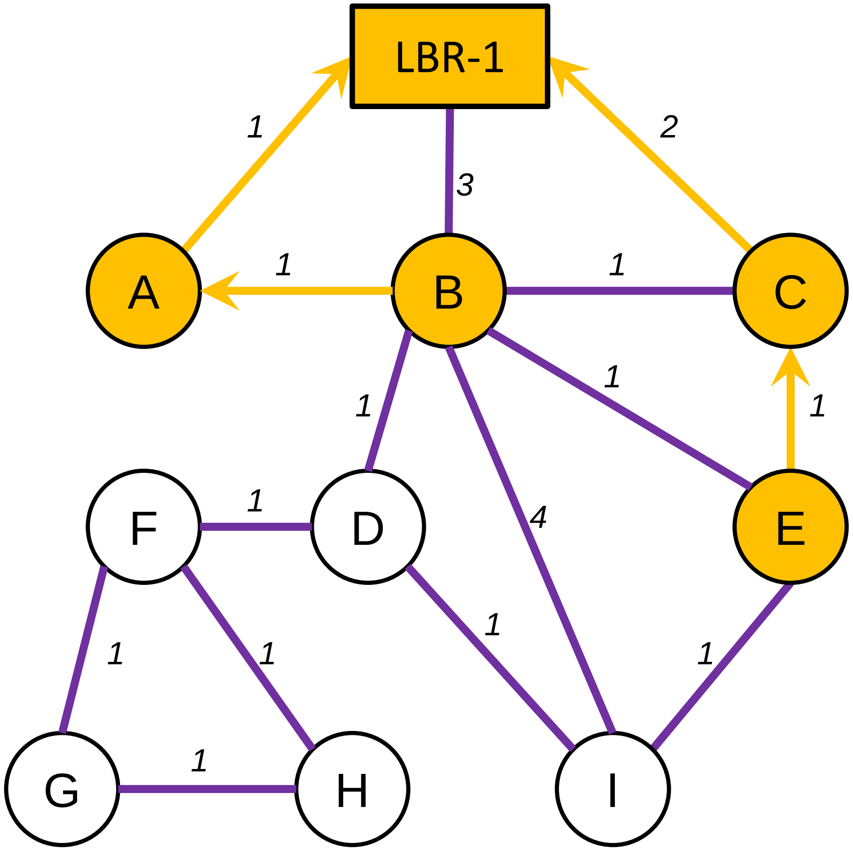
- Node A is at Depth 1, and can reach ::/0 via LBR-1 with ETX 1
- Node B is at Depth 3, with DAG Parents LBR-1, and can reach ::/0 via LBR-1 or C with ETX 3
- Node C is at Depth 2, ::/0 via LBR-1 with ETX 2
- Node E is at Depth 3, ::/0 via C with ETX 3

DAG Construction (7/9)



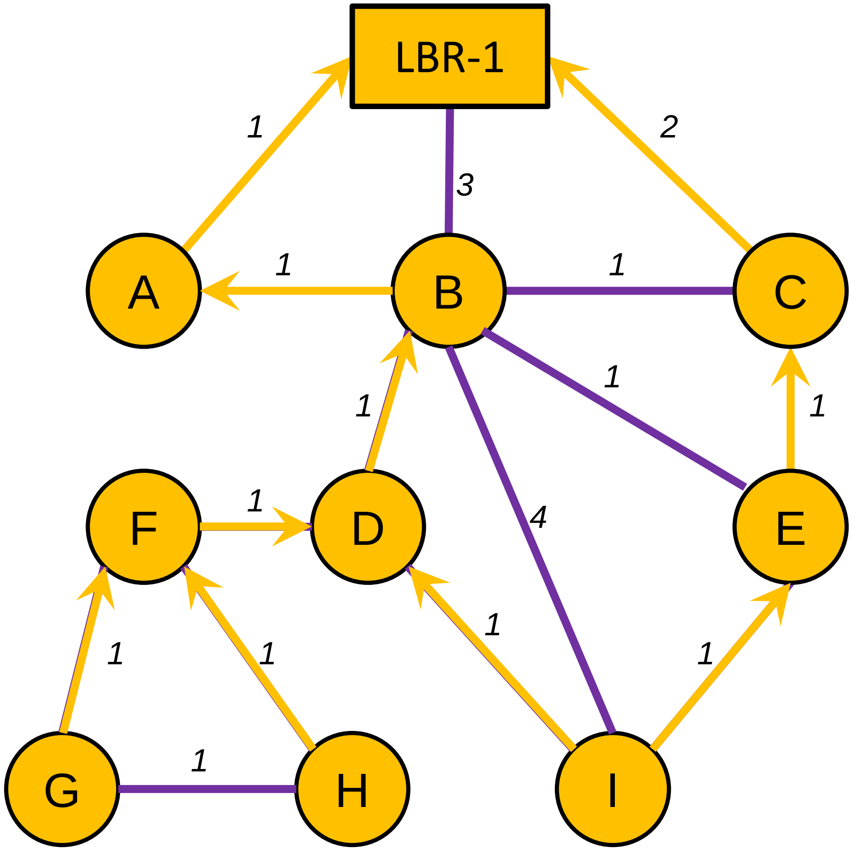
- The RA timer on Node A expires
- Node A multicasts RA-DIO
- LBR-1 ignores RA-DIO from deeper node
- Node B adds Node A
- Node B can improve to a more optimum position in the DAG
- Node B *removes* LBR-1 and Node C as DAG Parents

DAG Construction (8/9)



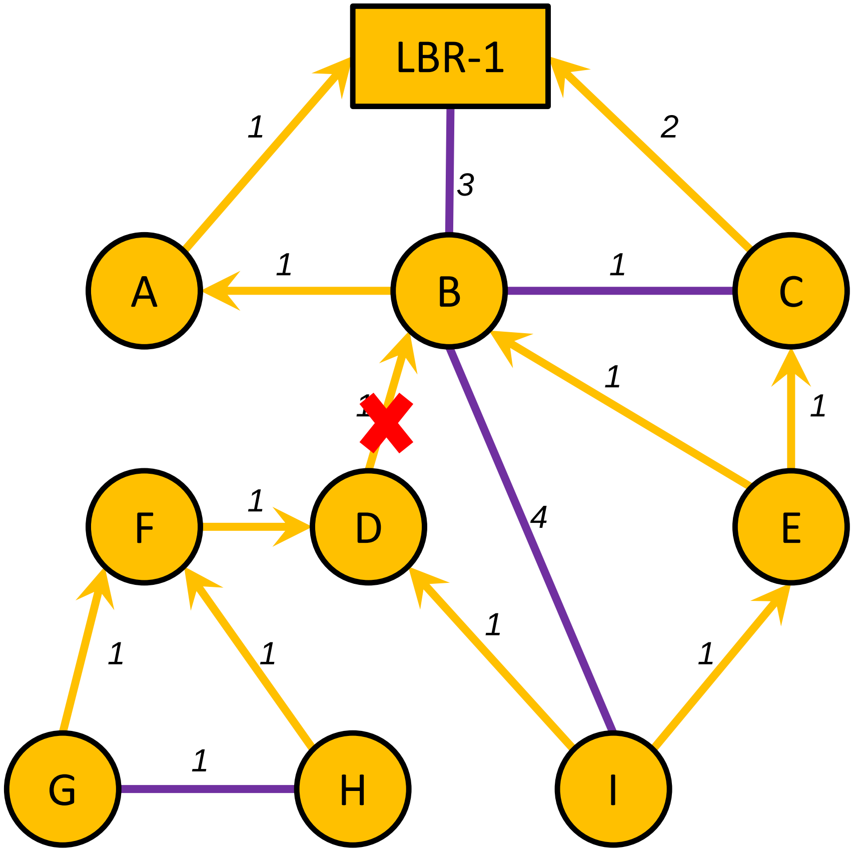
- Node A is at Depth 1, ::/0 via LBR-1 with ETX 1
- Node B is at Depth 2, ::/0 via A with ETX 2
- Node C is at Depth 2, ::/0 via LBR-1 with ETX 2
- Node E is at Depth 3, ::/0 via C with ETX 3

DAG Construction (9/9)



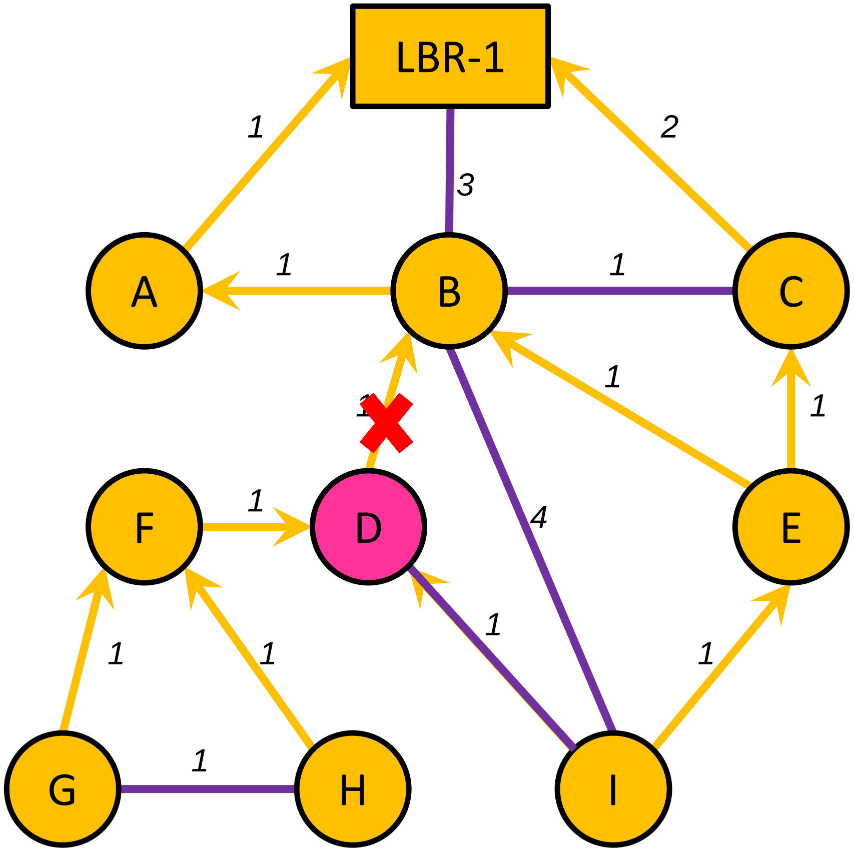
- DAG Construction continues…
- And is continuously maintained

DAG Maintenance (1/10)



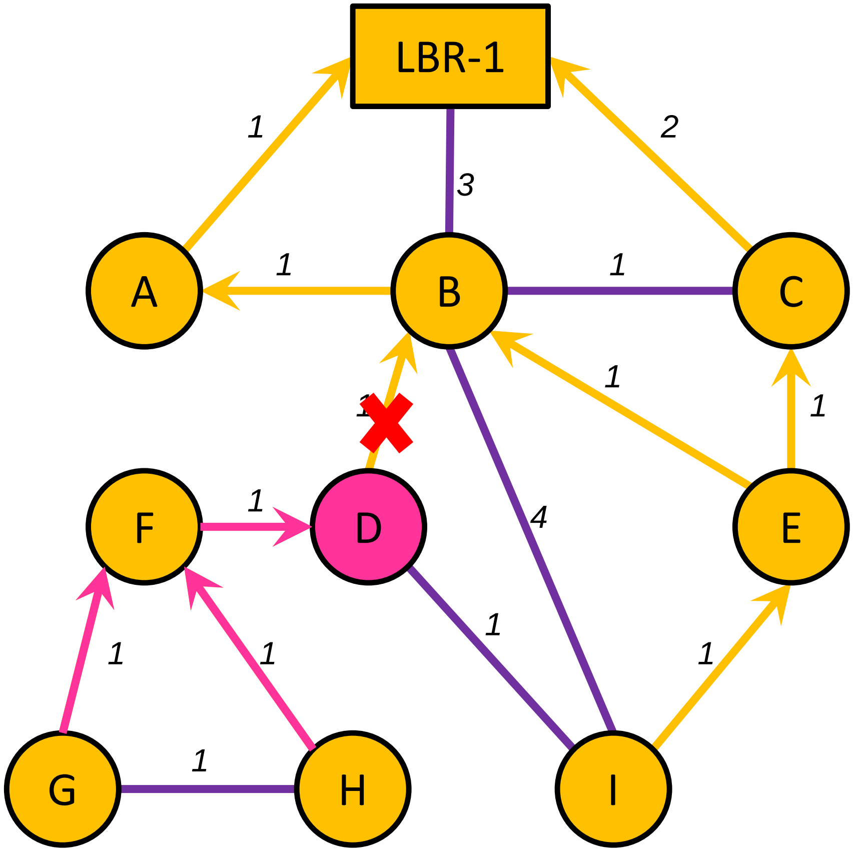
- Consider the case where the link B—D goes bad
- Node D will remove B from its DAG parent set
- Node D no longer has any DAG parent in the grounded DAG, so it will become the root of its own floating DAG

DAG Maintenance (2/10)



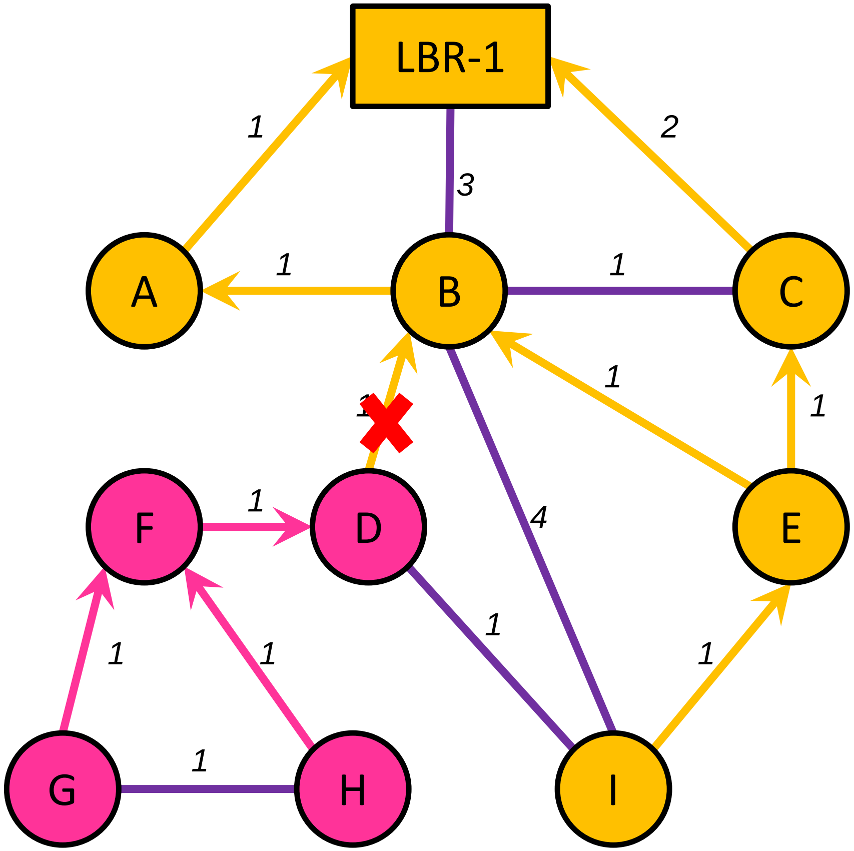
- Node D multicasts an router advertisement (RA)-DIO
  - to inform its sub-DAG of the change
- Node 'I' has an alternate DAG Parent, E
  - so it does not have to leave the DAG rooted at LBR-1.
- Node I removes Node D as a DAG Parent

DAG Maintenance (3/10)



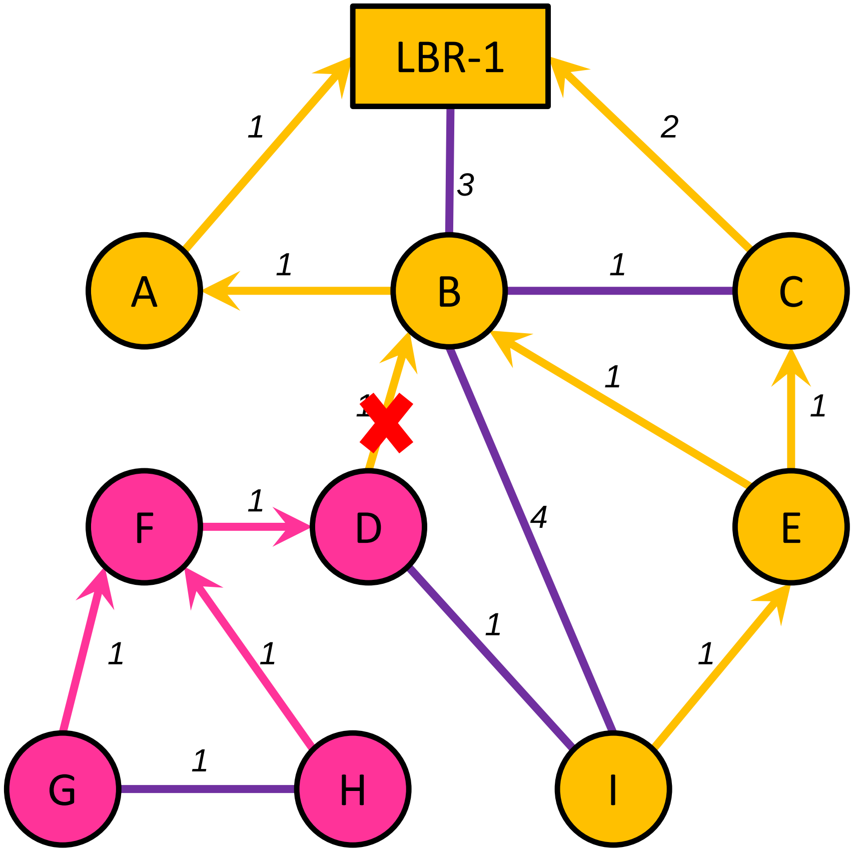
- Node F does not have an option to stay in the DAG rooted at LBR-1 (no alternate DAG Parents),
  - So, Node F follows Node D into the **floating DAG**
- Node F multicasts an RA-DIO
- Nodes G and H follow Node F into the floating DAG

DAG Maintenance (4/10)



- The sub-DAG of node D has now been frozen
- Nodes contained in the sub-DAG have been identified, and by following node D into the floating DAG, all old routes to LBR-1 have been removed
- The floating DAG seeks to rejoin a grounded DAG…

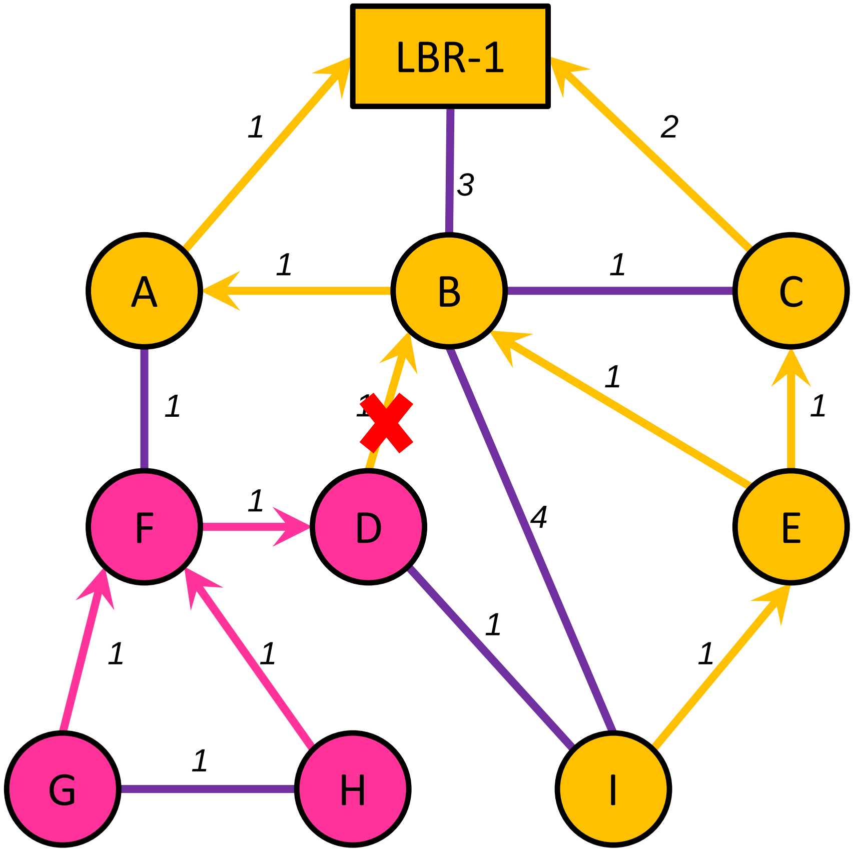
DAG Maintenance (5/10)



Re-join the Sub-DAG

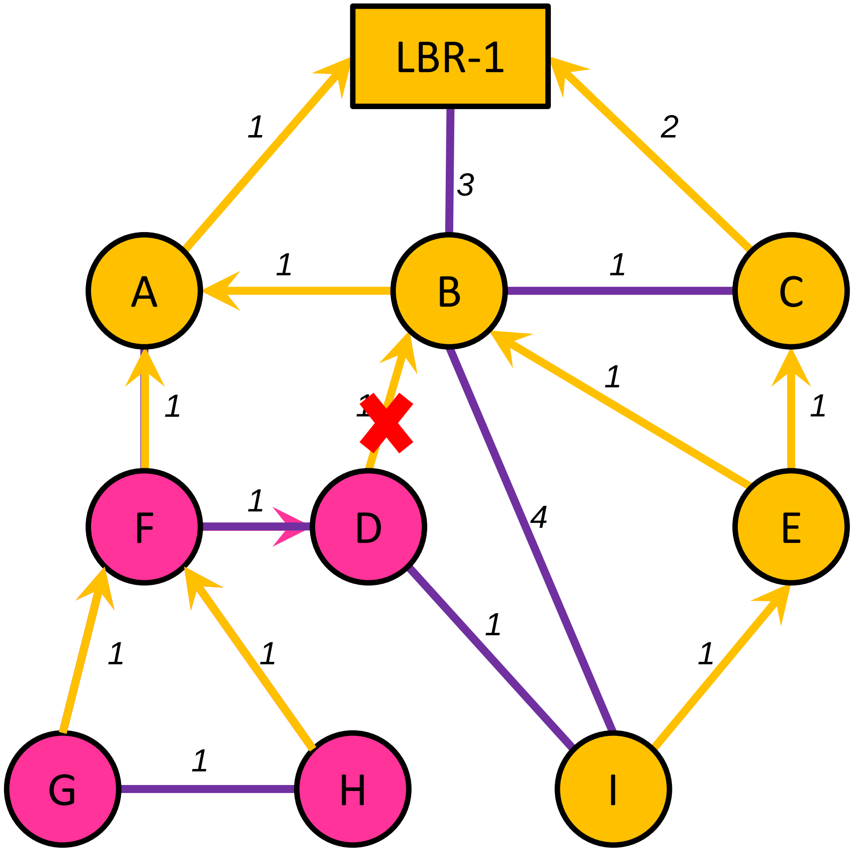
- Node I multicasts an RA-DIO
- Node D sees a chance to rejoin grounded DAG at depth 5 through Node I
- Node D starts a DAG Hop timer of duration α 4 (i.e. depth) associated with Node I

DAG Maintenance (6/10)



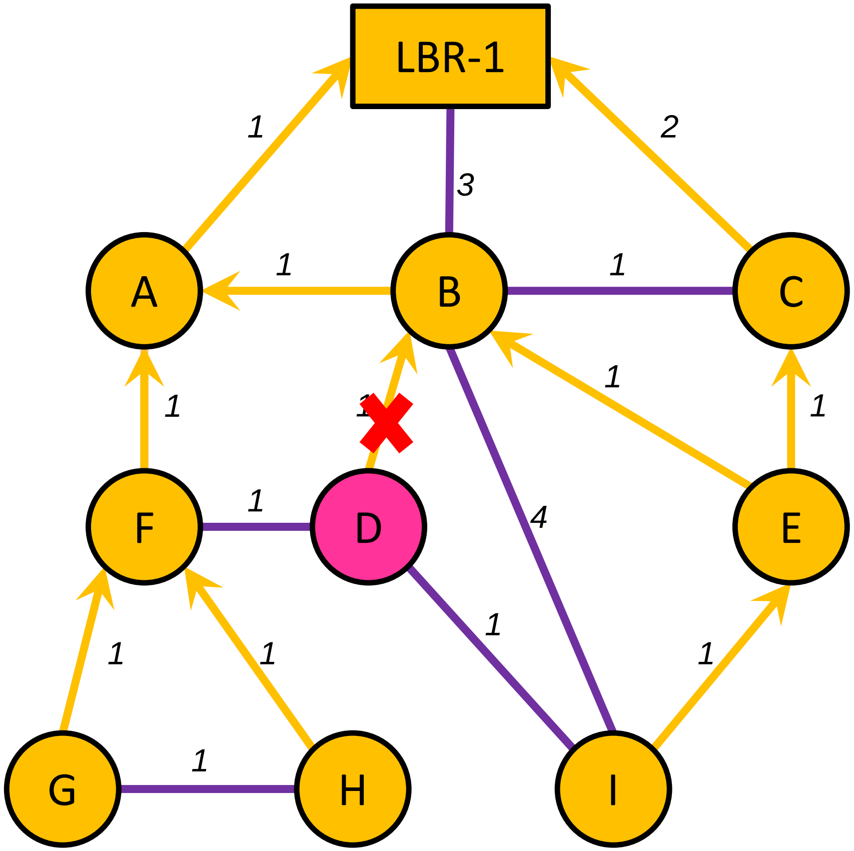
- Suppose a link A—F becomes viable
- Node A multicasts an RA-DIO
- Node F sees a chance to rejoin grounded DAG at depth 2 through Node A
- Node F starts a DAG Hop timer of duration α 1 (i.e. depth) associated with Node A

DAG Maintenance (7/10)



- Node F's DAG Hop Timer expires
- Node F joins to the grounded DAG at depth 2 by adding A as a DAG parent, and removing D
- Node F multicasts an RA-DIO
- Nodes G and H follow Node F to the grounded DAG

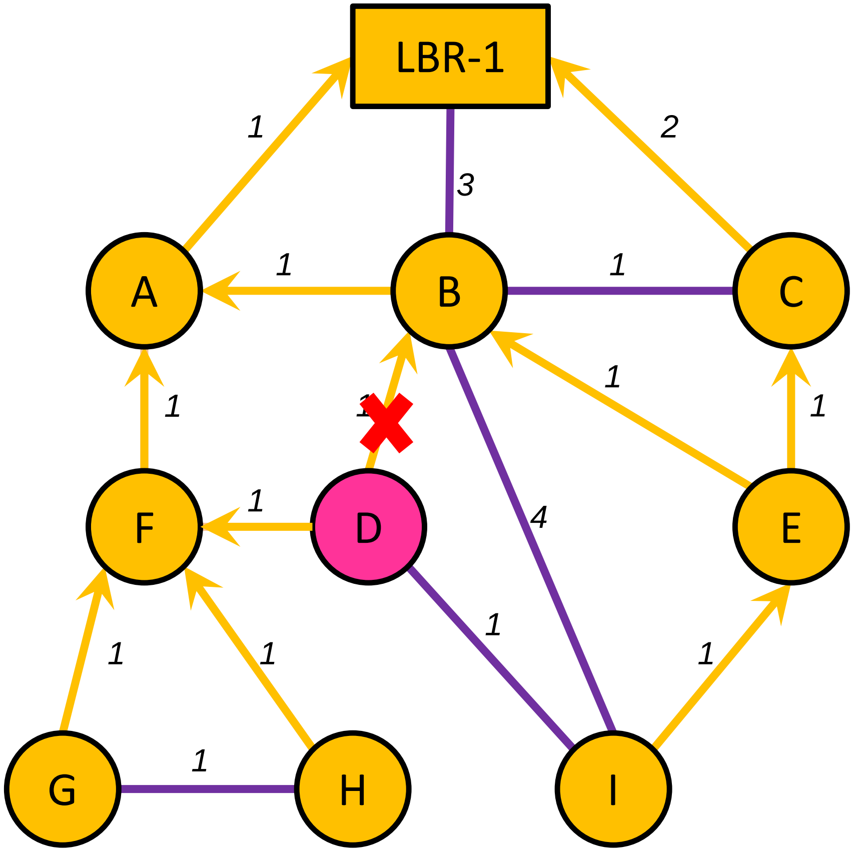
DAG Maintenance (8/10)



- Node D sees a chance to rejoin DAG LBR-1 at depth 3 through Node F

- Node D starts a DAG Hop timer of duration α 2 associated with Node F,
- in addition the DAG Hop timer already running with duration α 4 associated with Node I

DAG Maintenance (9/10)

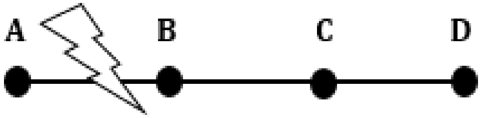


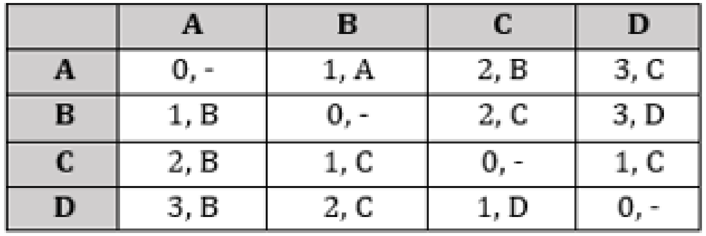
- Node D’s DAG Hop timer of duration α 2 tends to expire first
- Node D joins the grounded DAG at depth 3 by adding Node F as a DAG Parent
- The breaking-off and re-joining of the broken sub-DAG is thus coordinated with loop avoidance

DAG Maintenance (10/10)

- Loop Avoidance
  - ➢ mechanisms to avoid count-to-infinity problem

Link Between A & B is Broken





	A	B	C	D
A	0, -	1, A	2, B	3, C
B	1, B	0, -	2, C	3, D
C	2, B	1, C	0, -	1, C
D	3, B	2, C	1, D	0, -

Solutions:

➢ **Floating DAG**

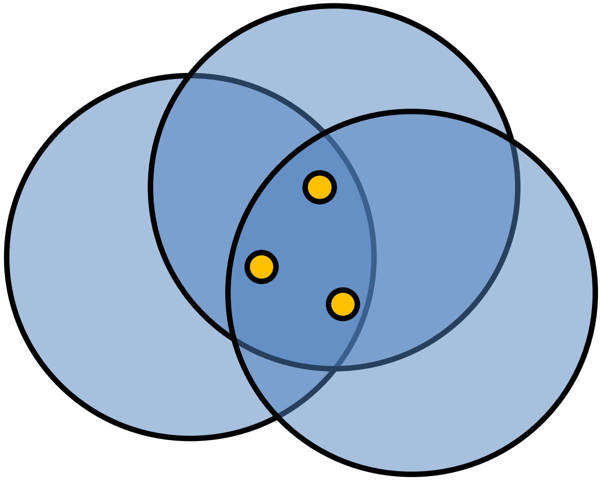
- Leave DAG, color sub-DAG, then look for new routes
- Operation local to nodes that must increase their depth
- Does not guarantee loop freedom

➢ **Sequence number change**

- Loop freedom, but expensive network-wide operation
- Used infrequently if possible

Trickle Algorithm [RFC 6206]

- Concerns
  - Broadcast is expensive
  - Wireless channel is a shared, spatial resource
- Idea
  - Dynamic adjustment of DIO transmission period
  - Suppress transmissions that may be redundant



- Parameters:
  - T_min: Minimum advertisement period
  - T_max: Maximum advertisement period
  - k: Suppression threshold
- Period adjustment:
  - On receiving *inconsistent* route information, reset to T_min
  - Otherwise, double up to T_max

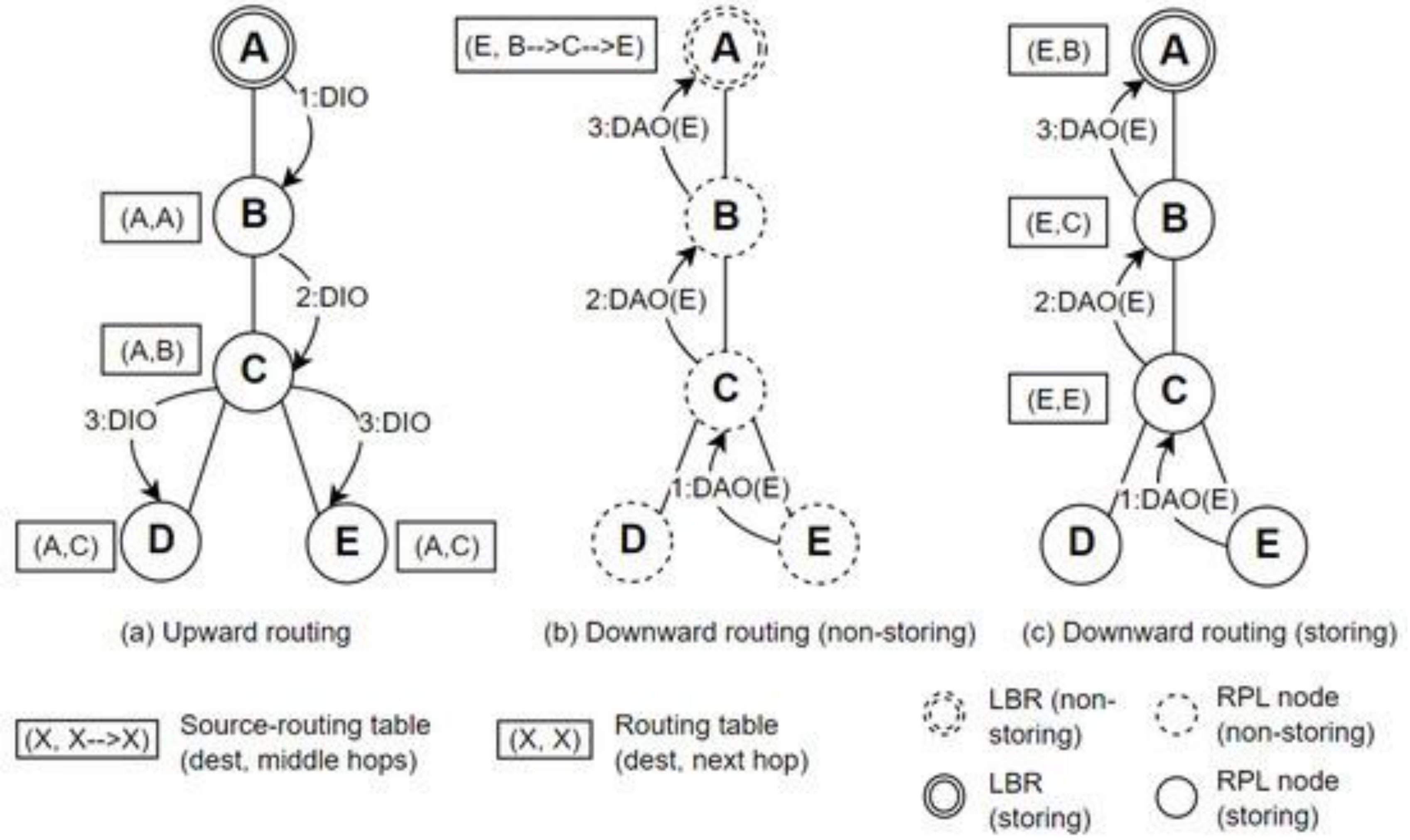
- Suppression:
  - Increment count (c) when receiving *similar* advertisement
  - At end of period, transmit if c < k, set c = 0
- Proposal:
  - Carry T_min, T_max, and k in RA-DIO

Thanks!

Figures and slide materials are taken from the following sources:

1. https://tools.ietf.org/agenda/75/slides/roll-1.ppt

Routing Establishment in RPL



(a) Upward routing (b) Downward routing (non-storing) (c) Downward routing (storing)

Zhang, K.; Bhandari, K.S.; Cho, G. “TB-RPL: A Try-the-Best Fused Mode of Operation to Enhance Point-to-Point Communication Performance in RPL”. *Electronics* **2023**, *12*, 1639.