Institute for the Wireless Internet of Things
at Northeastern University

EECE5155: Wireless Sensor Networks and the Internet of Things

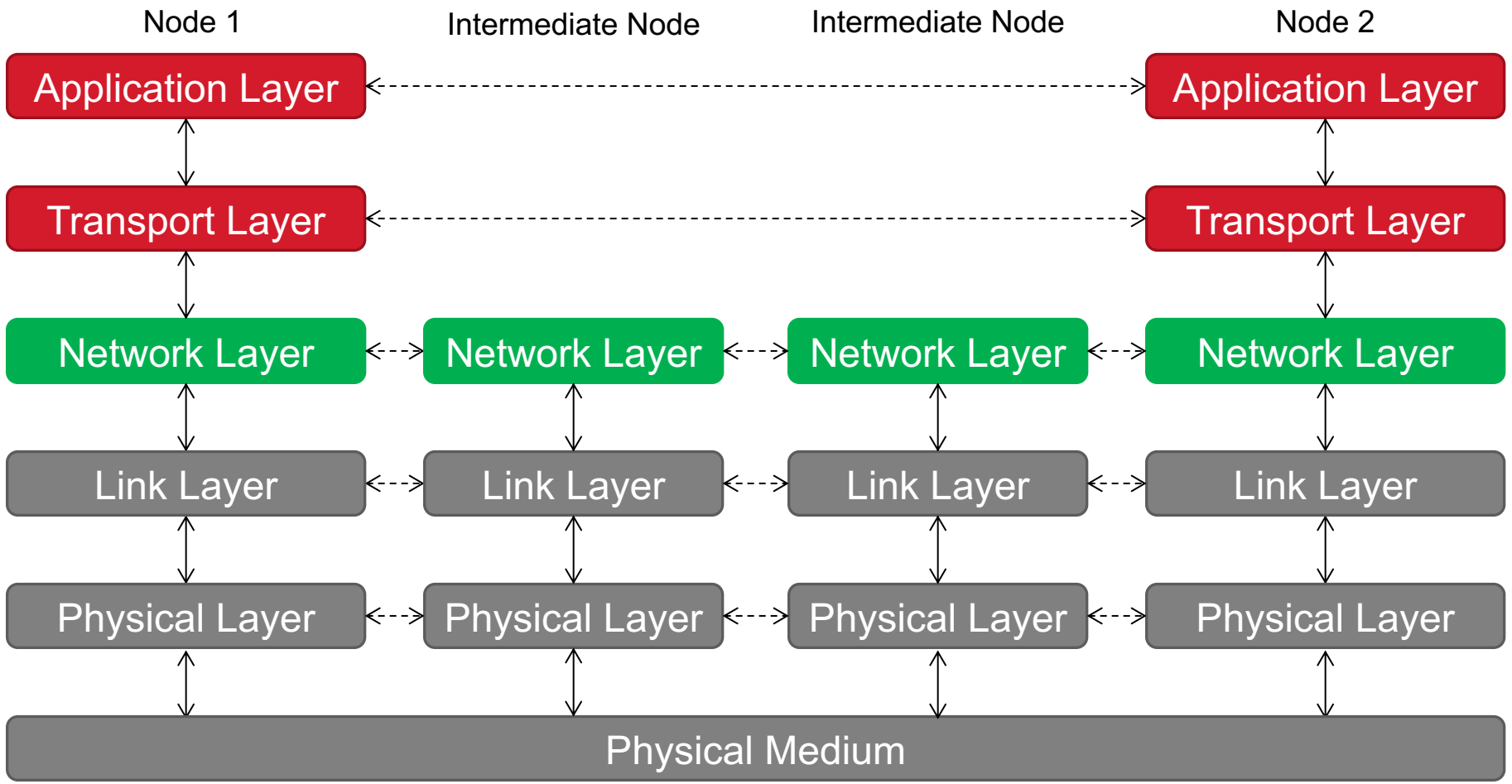
Josep Miquel Jornet
Associate Professor, Department of Electrical and Computer Engineering
Director, Ultrabroadband Nanonetworking Laboratory
Member, Institute for the Wireless Internet of Things
Northeastern University
jmjornet@northeastern.edu
www.unlab.tech

Institute for the Wireless Internet of Things at Northeastern

The Network Layer

The Protocol Stack

Institute for the Wireless Internet of Things at Northeastern



The Network Layer

- **Objective:** to enable the end-to-end transmission of packets between any source node and any destination node, which are not necessarily on the same link or network

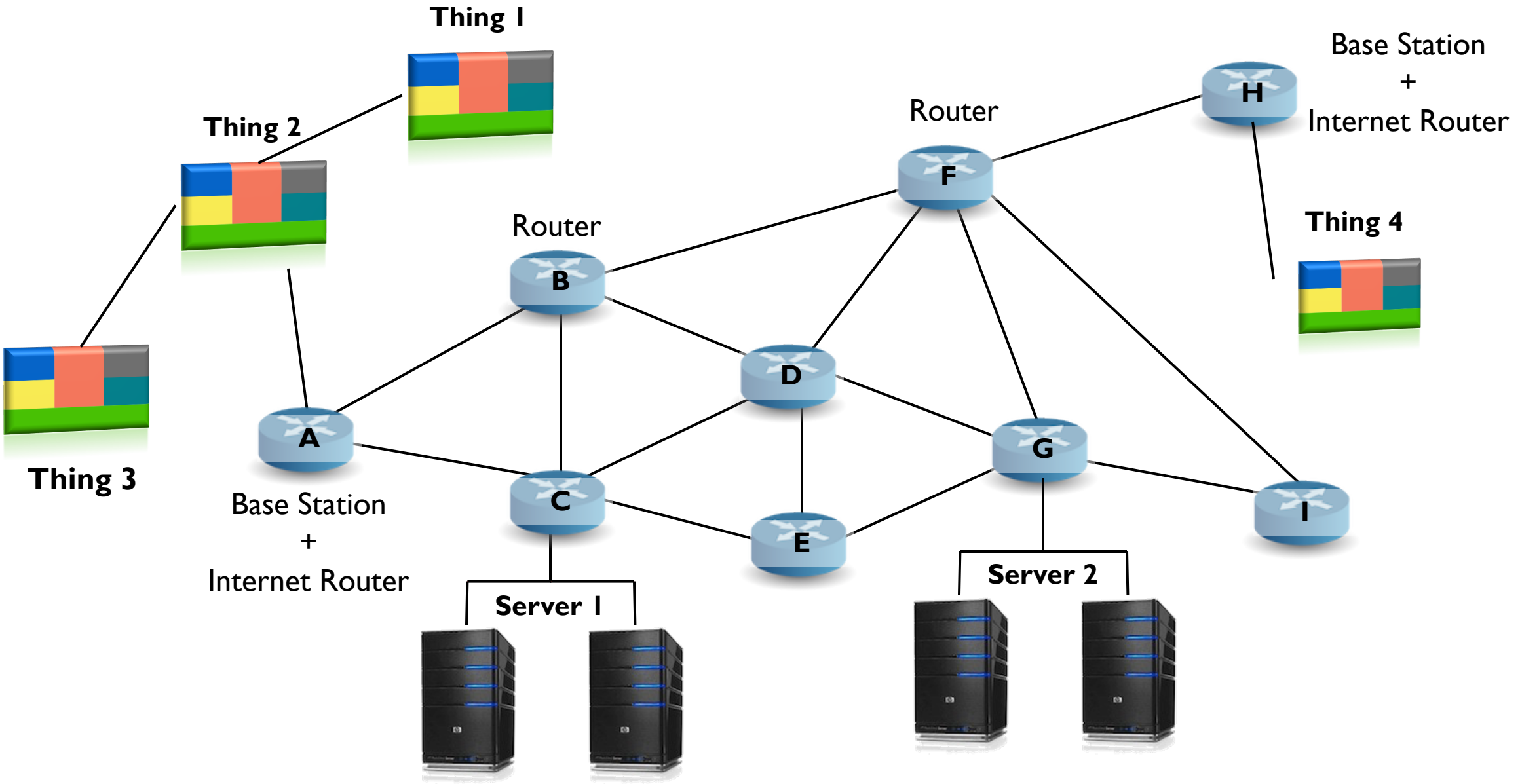
- **Functionalities:**
  - **Routing:** To find the **best paths** from the source to the destination, by taking into account the **network topology**
  - **Load balancing:** To prevent some intermediate nodes and links to become congested, by distributing the load in the network
  - **Internetworking:** To enable the interconnection of different networks independently of their underlying technologies

Routing Algorithms

Institute for the Wireless Internet of Things at Northeastern

- The **routing algorithm** is the process responsible for deciding which path a packet should follow towards its destination
- **Desired properties:**
  - Correctness
  - Simplicity
  - Robustness (against failures)
  - Stability (convergence of the paths)
  - Fairness
  - Energy Efficiency
- **Goal:** depends on the **routing metric**:
  - Minimum delay?
  - Maximum throughput?
  - Minimum energy consumption?

Example


Thing 1
Thing 2
Thing 3
Thing 4
Router
Router
Base Station
+
Internet Router
Base Station
+
Internet Router
A
B
C
D
E
F
G
H
I
Server 1
Server 2

Important

Institute for the Wireless Internet of Things at Northeastern

- In **traditional networks** (e.g., computer networks), user/end nodes and routers are very different devices
  - **User/end nodes:** generate or receive information
  - **Routers:** help move the information between user/end nodes

- In the **Internet of Things**, *Things* might act as both user/end nodes as well as routers
  - In many scenarios, Things are directly connected to a base-station/access point/controller
    - There is no routing needed here, they have a direct link to the network
  - In some scenarios, Things might help other Things get their information to other Things or to the base-station/access-point/controller
    - Here is when routing protocols enter the game

Important

Institute for the Wireless Internet of Things at Northeastern

- In the following slides:
  - First, we are going to review general concepts about routing in communication networks
  - Second, we are going to focus on the differences with the IoT and the new solutions

Routing Algorithms

- Types of routing:
  - **Non-adaptive (static) routing:**
    - Routes are computed in advance, off-line, and downloaded to the routers when the network is booted
    - Useful for situations in which the routing choice is clear
  - **Adaptive (dynamic) routing:**
    - Routes are periodically updated to reflect changes in the topology, network traffic, etc.
    - Different adaptive routing protocols differ in:
      - Where they get their information (e.g., locally, from adjacent routers, or from all routers)
      - When they change the routes (e.g., when the topology changes, or every ΔT seconds as the load changes)
      - What metric is used for optimization (e.g., distance, number of hops, or estimated transit time)

Optimality Principle

- If node *J* is on the optimal path from node *I* to node *K*, then the optimal path from *J* to *K* also falls along the same route

- Optimal routes from all sources to a given destination form a tree rooted at the destination, known as the **sink tree**:
  - Other trees with the same path lengths may exist
  - The tree **does not contain any loops**, so each packet will be delivered within a finite and bounded number of hops…
  - … at least in theory…

Shortest Path

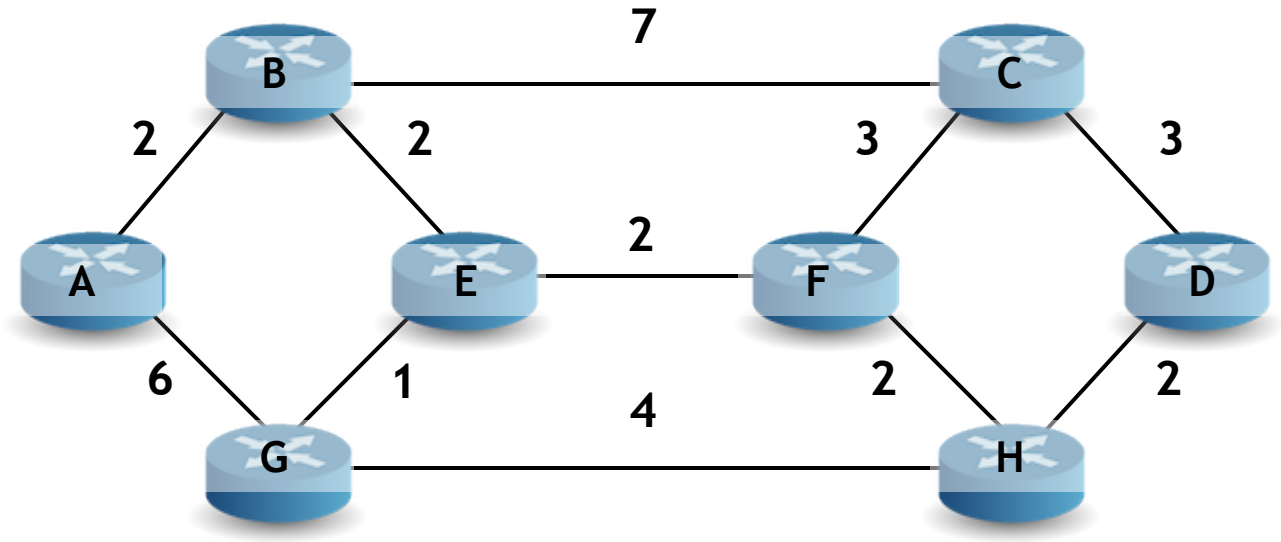
Institute for the Wireless Internet of Things at Northeastern

- **Objective:** To compute the optimal paths between any two nodes in the network given the complete network topology
  - These set the upper bound for any distributed routing protocol which might not have the full picture of the network
- **Starting point:**
  - Build a graph of the network:
    - Each node of the graph represents a node
    - Each edge of the graph represents a link
    - Each edge is assigned a "distance" in light of a specific criteria or routing metric:
      - Bandwidth/Capacity of the link
      - Propagation delay of the link
      - Energy left at the nodes in the ends of that link
      - Many others and/or combinations of them

Note: In our examples,

- We will aim at minimizing the distance of the total path, i.e., the *shortest path*
- We will consider that links are symmetric, i.e., their distance is the same whether we use the link in one direction or in the other:
  - This is not always true, specially in wireless networks, where the channel conditions might be really different…

Shortest Path



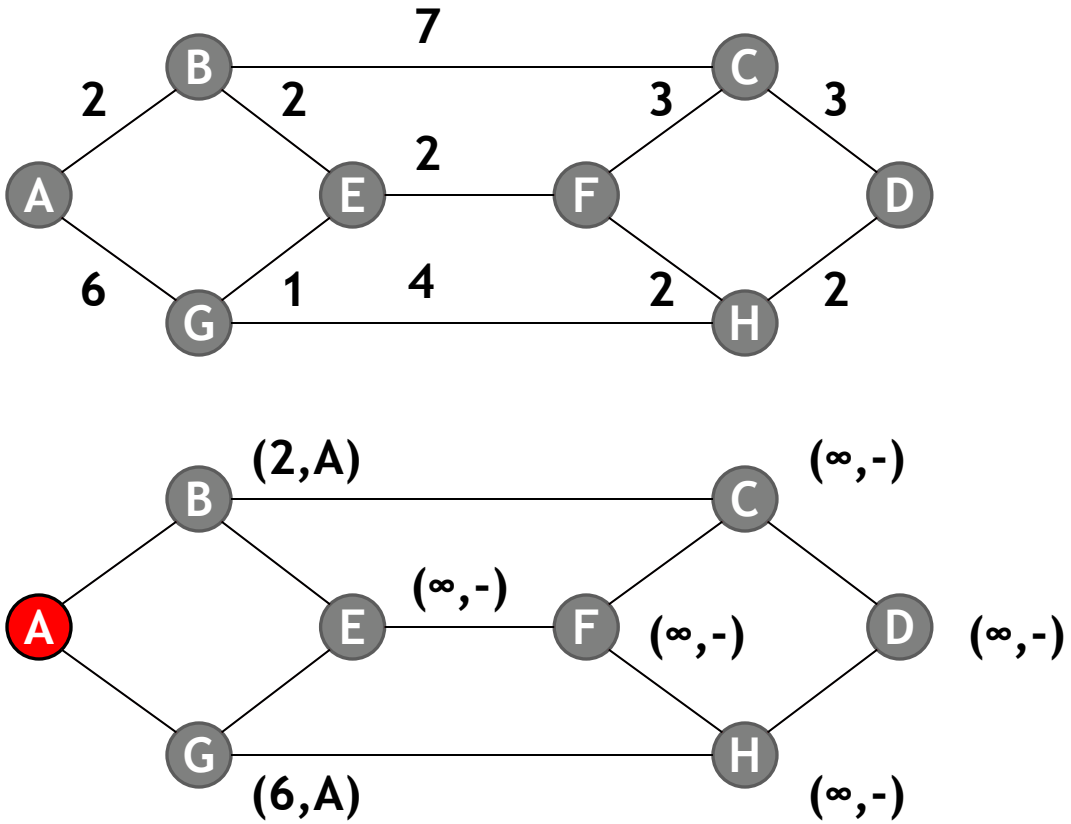
- **How can we systematically find the shortest path between any two nodes in the network?**

Dijkstra's Algorithm

- Published in 1959 by Edsger Dijkstra
  - Can be used in any application that requires the computation of the shortest path (not just routing in telecommunication networks)

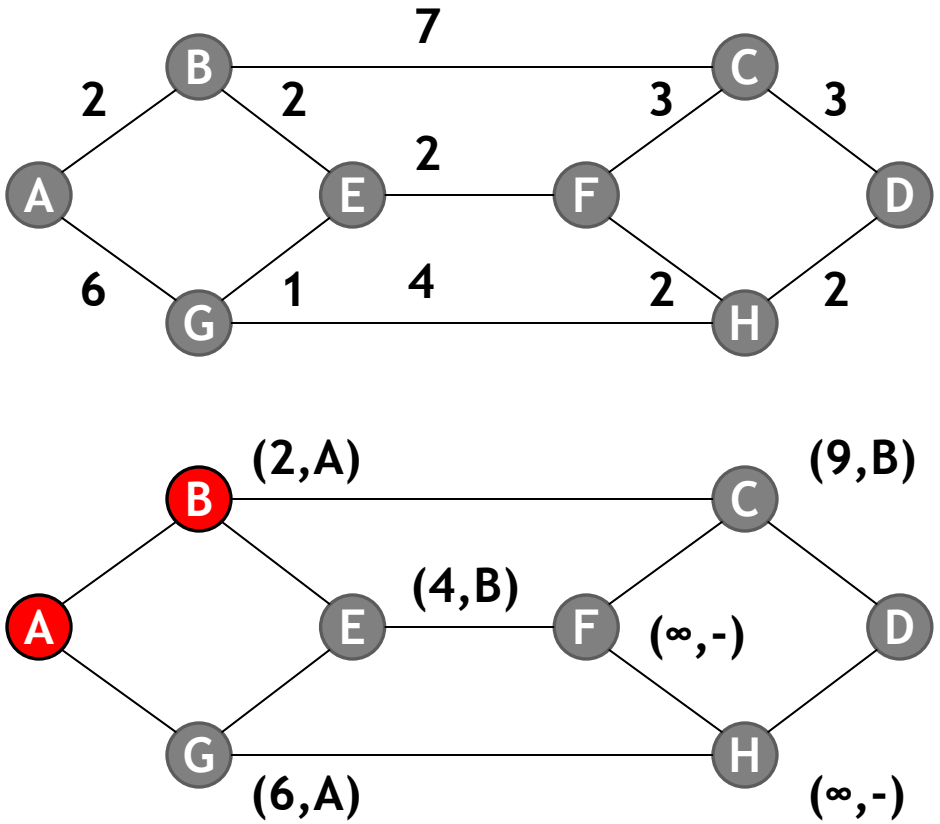
- **Procedure:**
  - Each node is labeled with its distance from the source node along the best-known path
    - Distances must be non-negative
    - Initially, no paths are known, so all nodes are labeled with infinity
    - A label might be tentative or permanent → Initially, all labels are tentative
    - A label becomes permanent when it is discovered that it represents the shortest path from the source to the node

Dijkstra's Algorithm



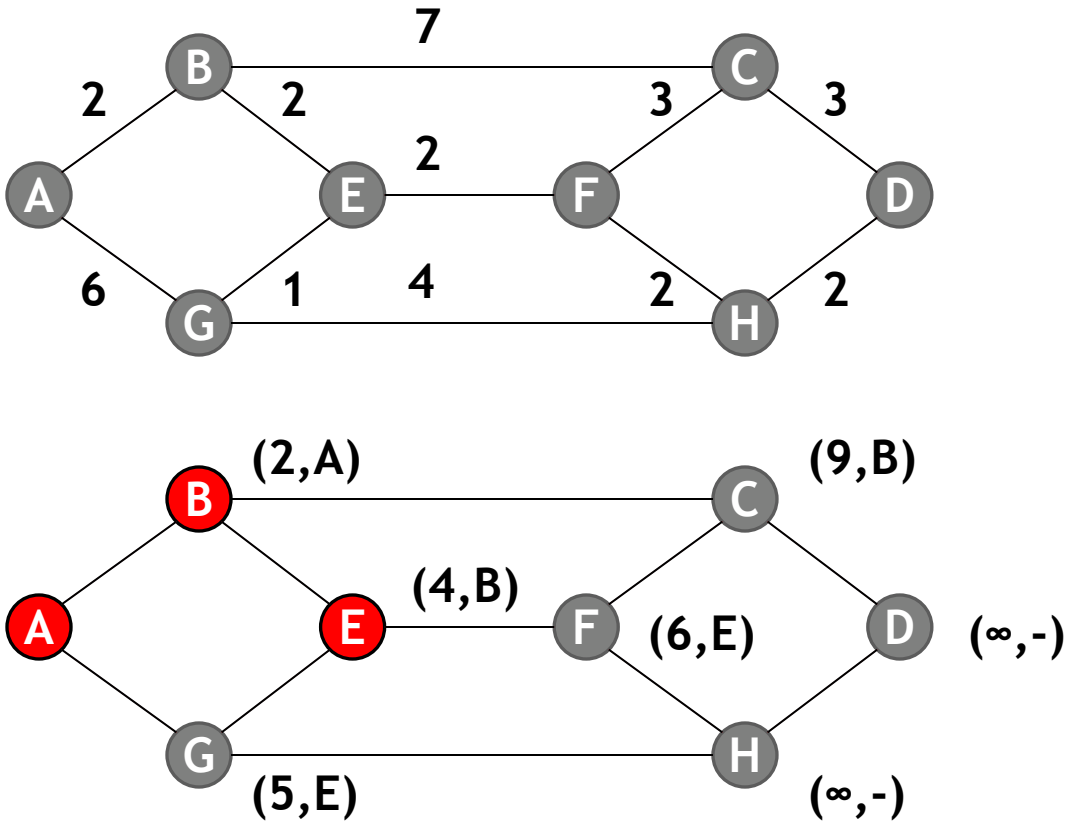
- We want to compute the shortest path from A to D:
  - We mark node A as permanent
  - We mark its one-hop neighbors with the distance and previous hop name (so we can reconstruct the route)

Dijkstra's Algorithm



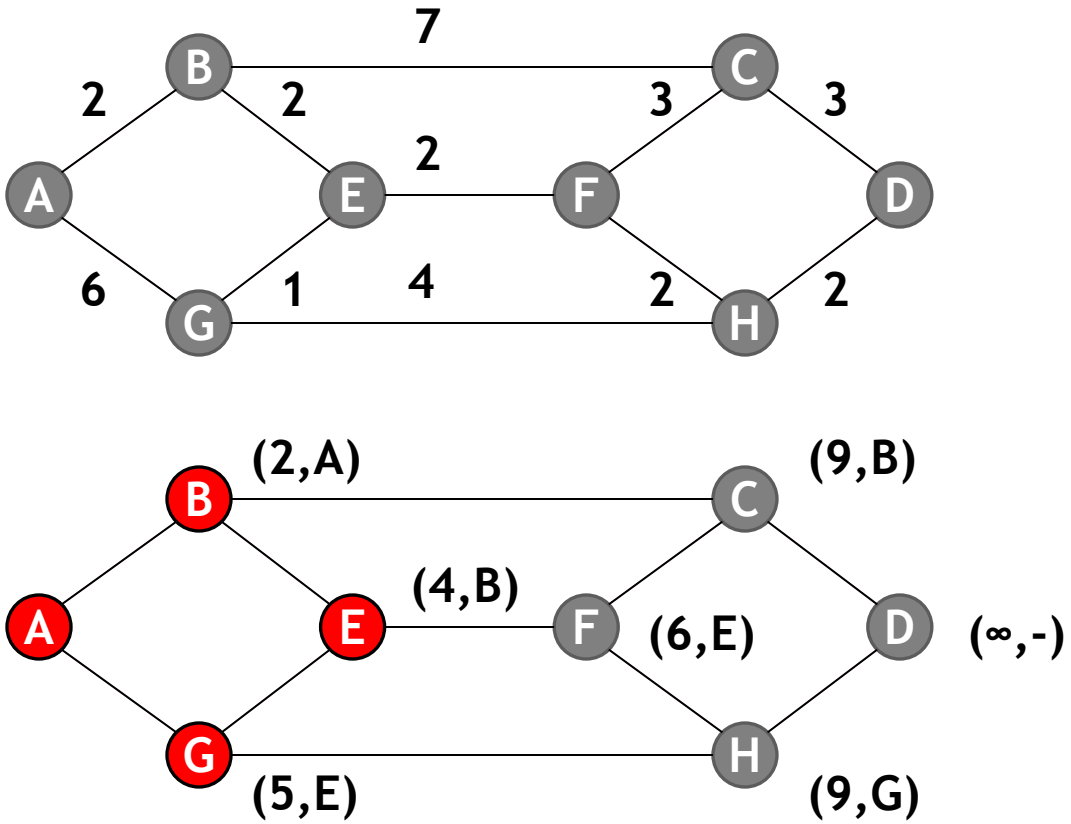
- We mark the node with the shortest distance to A as permanent: node B in our example
  - If there are two nodes with the same distance, we would have to compute and remember all the options
- We repeat the same procedure as before, i.e., we mark its one-hop neighbors with the distance and previous hop name

Dijkstra's Algorithm



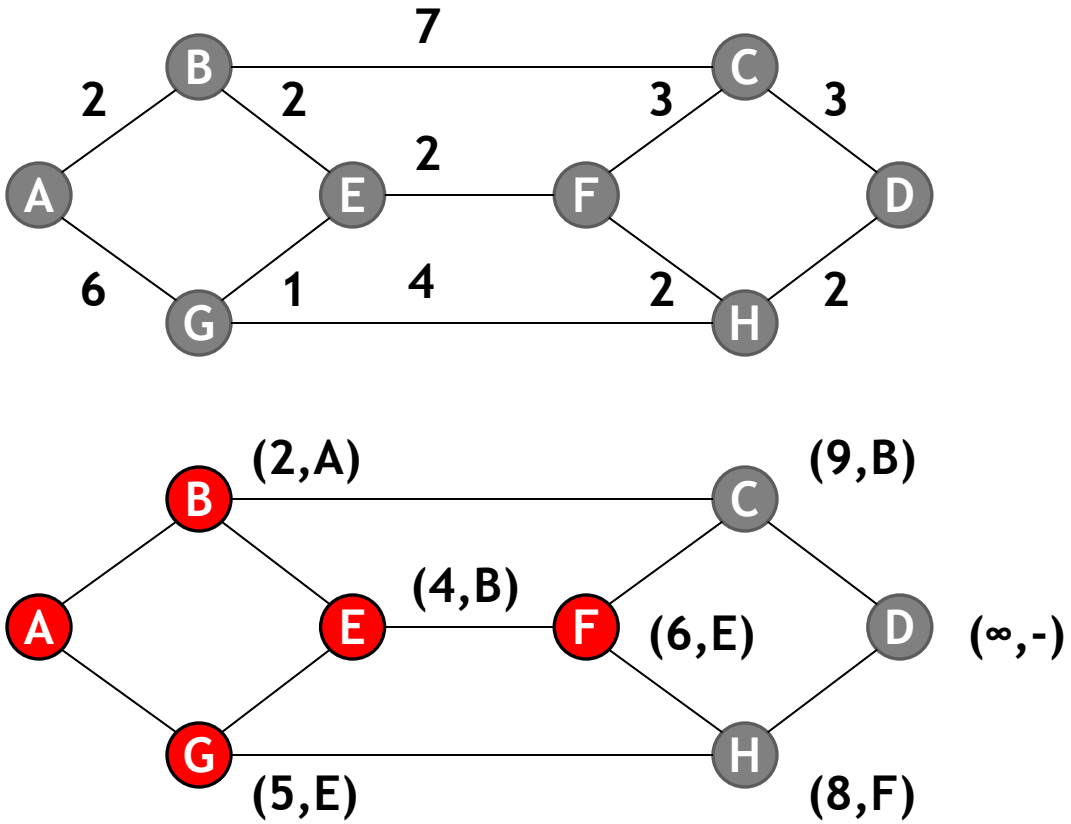
- We repeat the same procedure till reaching the destination

Dijkstra's Algorithm



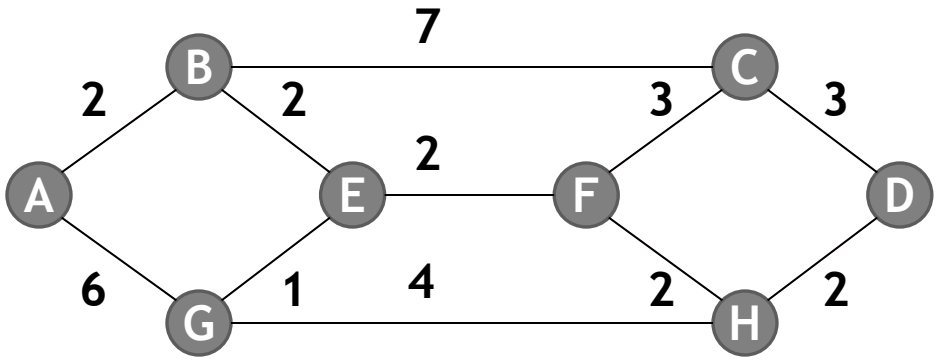
- We repeat the same procedure till reaching the destination

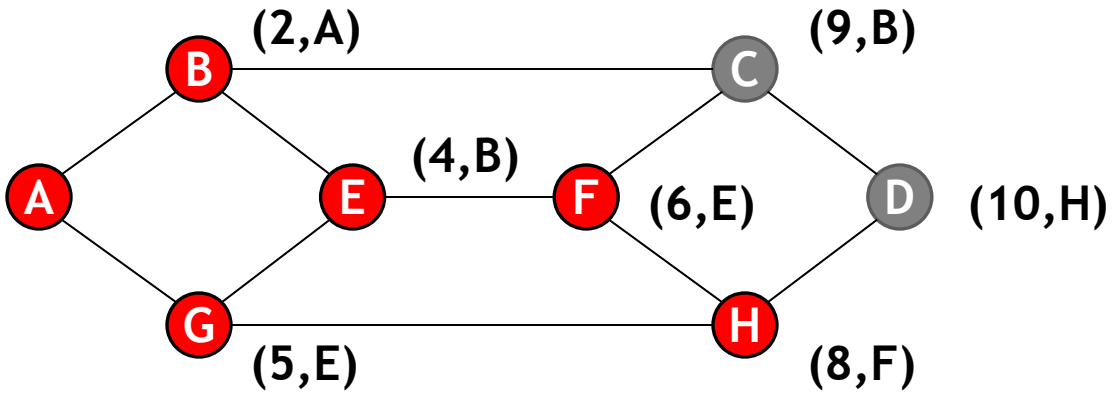
Dijkstra's Algorithm



- We repeat the same procedure till reaching the destination

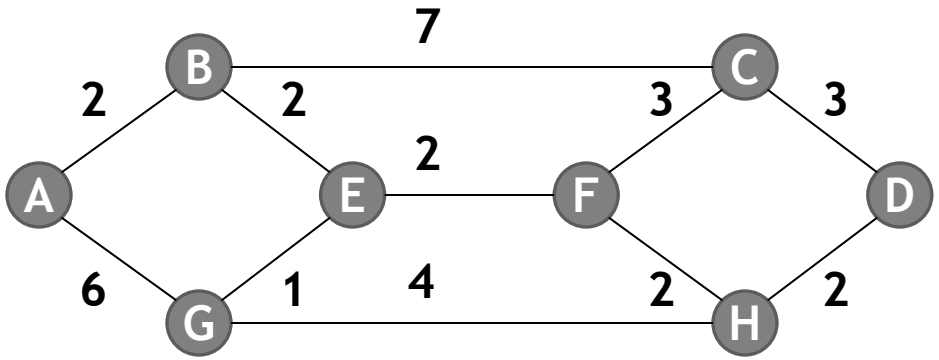
Dijkstra's Algorithm

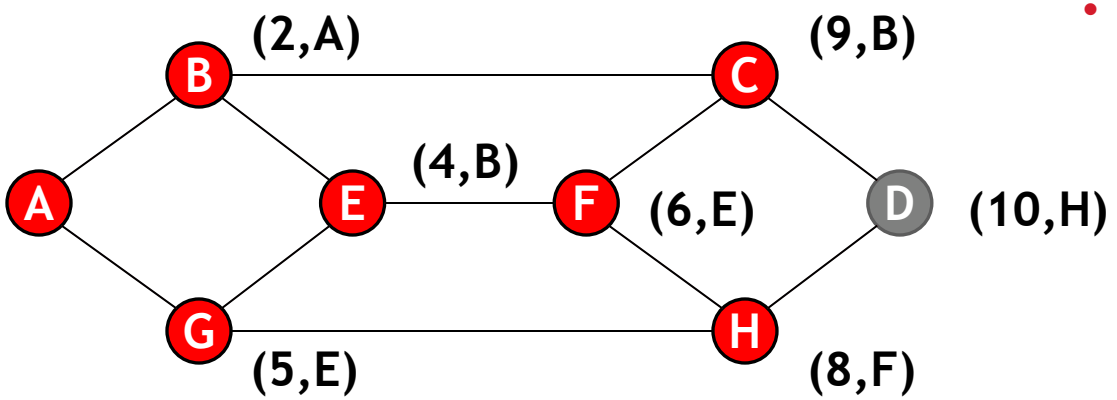




- We repeat the same procedure till reaching the destination

Dijkstra's Algorithm





- We reverse the path:
- D,H,F,E,B,A → ABEFHD

The Routing Table

- Once the routing algorithm is executed, the routing table is created:
- When a new packet reaches the router, it just checks the routing table and decides to which node the packet should be forwarded
- For example, the routing table at Node A is going to be:

Final Destination	Next Hop	Cost
B	B	2
C	B	9
D	B	10
E	B	4
F	B	6
G	B	5
H	B	8

Dijkstra's Algorithm

- This needs to be done for every possible node:
  - But remember the optimality principle…
- This can be easily programmed with a simple script
- However, this requires **complete knowledge of the network topology**…
  - Sometimes this is easy to obtain:
    - E.g., fixed network with static traffic in its links
  - Other times this is not realistic at all:
    - E.g., fixed network with dynamic traffic in its links
    - E.g., dynamic network with mobile and/or sleeping nodes

What can we do when we do not have this information?

Flooding

Institute for the Wireless Internet of Things at Northeastern

- To send every received packet through all the possible links except the one that it was received through
  - Generates vast numbers of duplicate packets → Infinite number unless something is done:
    - **Option 1:**
      - A hop counter can be added in the packet header
      - At each node, the hop counter is decreased by one
      - If the counter reaches zero, the packet is discarded
        - The initial value of the counter should be, in the worse case, the full diameter of the network
      - **Problem:** we still have an exponential number of duplicate packets
    - **Option B:**
      - A sequence number can be added in the packet header
      - A node memorizes the sequence number of the packets it has transmitted for each specific node
      - If a packet with the same sequence number is received, the packet is discarded
      - **Problem:** the list can grow without bound → Instead, for each node, only an indicator of the largest sequence number can be stored (but then, what happens with out-of-order packets?)

Flooding

- Still some positive aspects:
  - Controlled flooding (e.g., with hop counter and/or sequence number control) is very useful for **information broadcasting**
    - E.g., control information broadcasting in wireless networks
  - Controlled flooding is also **very robust**:
    - Even if some of the paths/links become broken, if there is a path between the source and the destination, flooding will eventually traverse it
      → This might be very useful for military applications or emergency recovery networks
  - It can serve as a metric to compare with other protocols:
    - E.g., if we don't consider the congestion that duplicates packets create in the network, it can provide us with the shortest path/shortest delay, because it actually **explores all the paths in parallel**!

Distance Vector Routing (DVR)

- Each router maintains a table (i.e., a vector) that specifies the best-known distance to each destination and the link to use to get there
- These tables are updated by exchanging information with the neighbors
- Eventually, every router knows the best link to reach each destination
- It is commonly referred to as the Bellman-Ford routing algorithm (Bellman 1957, Ford and Fulkerson, 1962)
- Was the original ARPANET routing algorithm and was also used in the Internet under the name RIP

Link State Routing (LSR)

Institute for the Wireless Internet of Things at Northeastern

- Officially replaced the Distance Vector Routing in ARPANET in 1979:
  - Much faster convergence in the presence of changes
  - Variants of Link State Routing such as IS-IS and OSPF are widely used today inside large networks and the Internet

- **Basic idea:** each router should proceed as follows:
  1. Discover its neighbors and learn their network addresses
  2. Set the distance or cost metric to each of its neighbors
  3. Construct a packet telling all it has just learned
  4. Send this packet to and receive packets from all other routers
  5. Compute the shortest path to every other router

- The complete topology is distributed to every router
  - Then **Dijkstra's algorithm** can be run at each router to find the shortest path to every other route!

Some Remarks for DVR and LSR

- DVR, LSR and many other routing algorithms rely on processing at all the routers to compute routes

- Problems with the hardware or software at even a small number of routers can create network-wide problems. For example,
  - If a router claims to have a link it does not have or forgets a link it does have, the network graph will be incorrect
  - If a router fails to forward packets or corrupts them while forwarding them, the route will not work as expected
  - If a router runs out of memory or does the routing calculation wrong, problems will appear

- As the network grows into the range of tens or hundreds of thousands of nodes, the probability of some router failing occasionally becomes non-negligible:
  - The trick is to try to arrange to limit the damage when the inevitable happens

Hierarchical Routing

- As networks grow in size, the **routing tables grow proportionally**:
  - More memory is consumed at the router to store them
  - More CPU time is needed to scan them
  - More bandwidth is needed to send status reports about them

- At a certain point, the network may grow to the point where it is no longer feasible for every router to have an entry for every router!

- **Solution:** hierarchical routing (as it is in the telephone network)

Hierarchical Routing

- **Basic idea:** Routers are divided into **regions**:
  - Each router knows all the details about how to route packets to destinations within its own region…
    - …but knows nothing about the internal structure of other regions
  - When different networks are interconnected, each one is considered as a separate region…
    - … so the routers in one network do not need to know the topological structure of the other networks
- For huge networks, a two-level hierarchy may be insufficient:
  - It may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on, until we run out of names for aggregations

Broadcast Routing

Institute for the Wireless Internet of Things at Northeastern

- **Broadcasting:** sending a packet to all destinations simultaneously
  - E.g., weather alerts, traffic alerts, live radio shows, etc.

- Various methods have been proposed to broadcast packets:
  - **Option 1:** to simply send a distinct packet to each destination → Super inefficient
  - **Option 2: Multidestination Routing**, in which each packet contains either a list of destinations or a bit map indicating the desired destinations → Still inefficient
  - **Option 3: Flooding:** Good to broadcast quickly the information… but we can do better!
  - **Option 4:** There are many smart ways to achieve broadcasting (starting from smartly controlled flooding, reverse path forwarding, etc.)

Multicast Routing

- **Multicasting:** to send messages to well-defined groups that are numerically large in size but small compared to the network as a whole
  - All multicasting schemes require some way to create and destroy groups and to identify which routers are members of a group
    - How these tasks are accomplished is not of concern to the routing algorithm
    - We assume that each group is identified by a multicast address and that routers know the groups to which they belong

- E.g., multi-player gaming, live video of a sports event, etc.

Anycast

- A packet is delivered to the nearest member of a group

- This is useful when the importance is on getting the correct information, not about who sends the information
  - E.g., many applications of wireless sensor networks!

- Every node in a group is assigned the same address
  - If we now run DVR or LSR, we will just find the closest neighbor

The Advantage of Wireless Networks

- Broad/Multi/Any-casting in wireless is unlike broad/multi/any-casting in a wired medium
  - Wires: locally distributing a packet to *n* neighbors: *n* times the cost of a unicast packet
  - Wireless: sending to *n* neighbors can incur costs
    - As high as sending to a single neighbor – if receive costs are neglected completely
    - As high as sending once, receiving *n* times – if receives are tuned to the right moment
    - As high as sending *n* unicast packets – if the MAC protocol does not support local multicast
- If local multicast is cheaper than repeated unicasts, then wireless multicast advantage is present
  - Can be assumed realistically

Mobile Routing

Institute for the Wireless Internet of Things at Northeastern

- The presence of mobile nodes introduces several challenges to the functioning of telecommunication networks
- Many different scenarios:
  - **Mobile hosts and static routers**
    - **Option A:** A mobile host replaces its network address when it moves
      - E.g., you take your laptop from your room to the library
      - All the applications need to know your new address
        → You reconnect to all the services you are using
    - **Option B:** A mobile host keeps its network address when it moves
      - E.g., all-IP networks such as VoIP, corporate networks, etc.
      - Everything is transparent to the applications
        → But the network layer needs to make sure to find you
  - **Mobile hosts and mobile routers**
    - E.g., mobile ad-hoc networks (more later)

Mobile Hosts and Stationary Routers: Option A

Institute for the Wireless Internet of Things at Northeastern

- A host forgets its old address and obtains a new address
  - This is not real network layer problem, i.e., the routing protocol doesn't need to do much
  - Example: You take your laptop from your room to the library
    - **MAC address:** The same (goes with your network adapter)
    - **Network address:** Different → Your closest router assigns you a network address of its region
    - **Applications:** You need to login again → You tell the specific servers your new address
    - The routing protocol does not need to do anything, as long as there were already routes created for all the nodes

Mobile Hosts and Stationary Routers: Option B

- A host always keeps its network address

- How can the information be **routed to it**?
  - **Option 1:** To recompute all the routing tables by taking into account the new network topology
    - Too much burden!
  - **Option 2:** The host tells a host in the home location where it is now, so it can forward the packets to him
    - This is the concept behind **Mobile IP networks** (not that common anymore)

Mobile Hosts and Mobile Routers

- This is the case in **mobile ad-hoc networks (MANETs)**:
  - E.g., military networks (in the battlefield), emergency networks (after an earthquake, tsunami), **mobile sensor and actor networks**, etc.

- In a MANET,
  - Nodes can act both as hosts and as routers
  - Nodes can appear and disappear in the same place or a different place of the network

- Many protocols have been designed for MANETs:
  - They are all usually variations of the same fundamental concepts

Ad Hoc Routing Protocols – Classification

Institute for the Wireless Internet of Things at Northeastern

1. **When does the routing protocol operate?**
   - **Proactive Protocols**
     - Routing protocol always tries to keep its routing data up-to-date
     - Active before tables are actually needed
     - Also known as table-driven
   - **Reactive Protocols**
     - Route is only determined when actually needed
     - Protocol operates on demand
   - **Hybrid protocols**
     - Combine these behaviors

Ad Hoc Routing Protocols – Classification

Institute for the Wireless Internet of Things at Northeastern

2. **Network Topology**
   - Flat, Hierarchical
3. **Which data is used to identify nodes?**
   - An arbitrary identifier?
   - The ***position*** of a node?
     - Can be used to assist in ***geographical*** routing protocols because choice of next hop neighbor can be computed based on destination address
     - Scalable and suitable for sensor networks
   - Identifiers that are not arbitrary, but carry some structure?
     - As in traditional routing
     - Structure akin to position, on a logical level?

Ad-hoc On-demand Distance Vector Routing

- Each node has a distance vector table, which specifies the distance and the neighbor to which packets should be sent **for each known destination**

- **Reactive Routing:** Routes to a destination are discovered on demand, only if needed:
  - When a node needs to send a packet, it first checks its routing table:
    - If there is a valid route to the destination (i.e., it is a direct neighbor or it has recently transmitted to it), then the packet is just sent through that route
    - If not, the **route discovery process** is triggered

AODV: Route Discovery

Institute for the Wireless Internet of Things at Northeastern

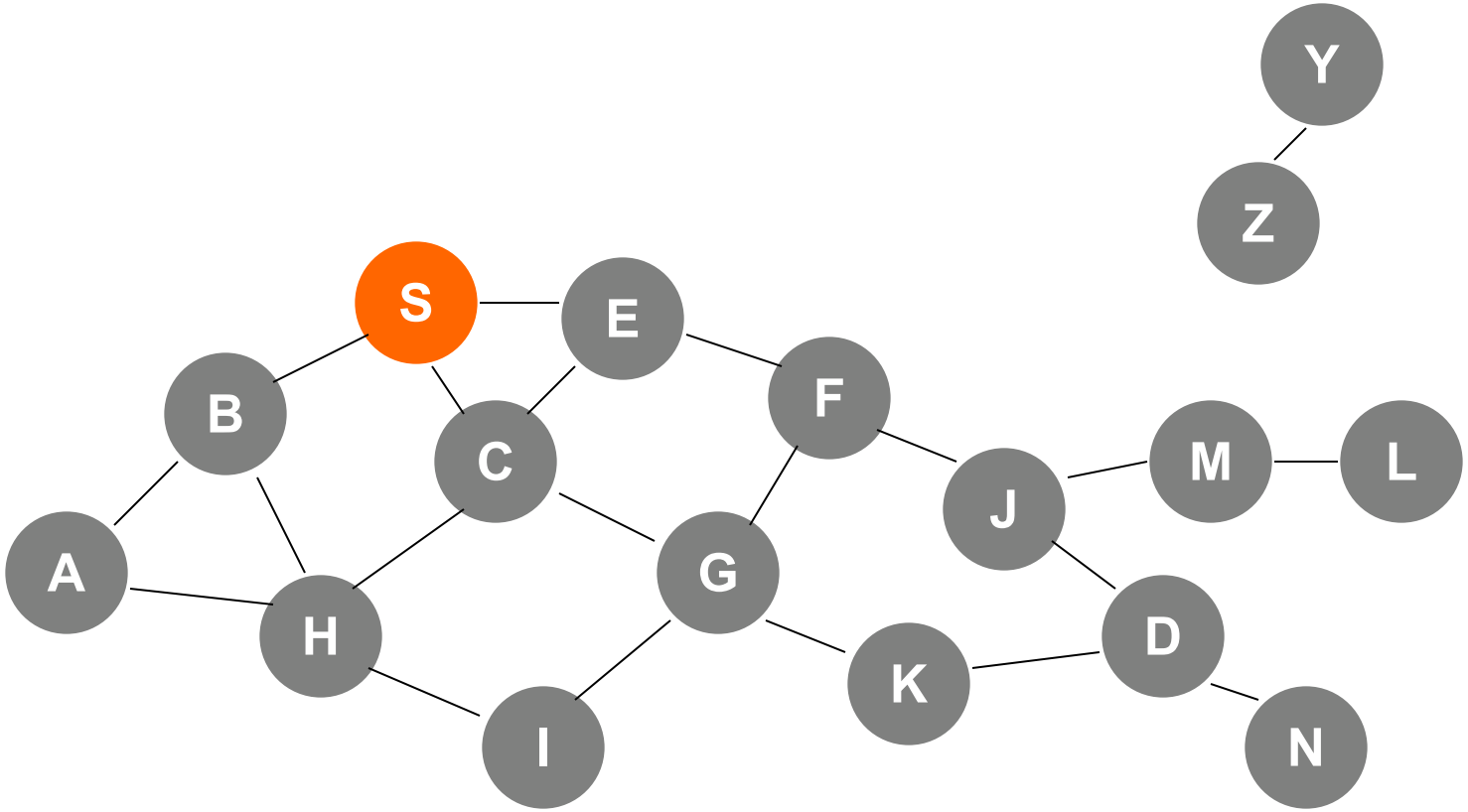
- The source node constructs a **Route Request** packet and broadcasts it using controlled flooding with sequence number:
  - Each intermediate node memorizes the node that has sent the packet to it, to be able to compute the reverse route

- When the Route Request packet reaches the destination, this constructs a **Route Reply** packet and sends it back using the reverse path
  - Each intermediate node increases the hop counter in the Route Reply packet, to learn how far they are from the destination
  - Each intermediate node learns the best route to the destination, i.e., the neighbor that has just sent them the Route Reply

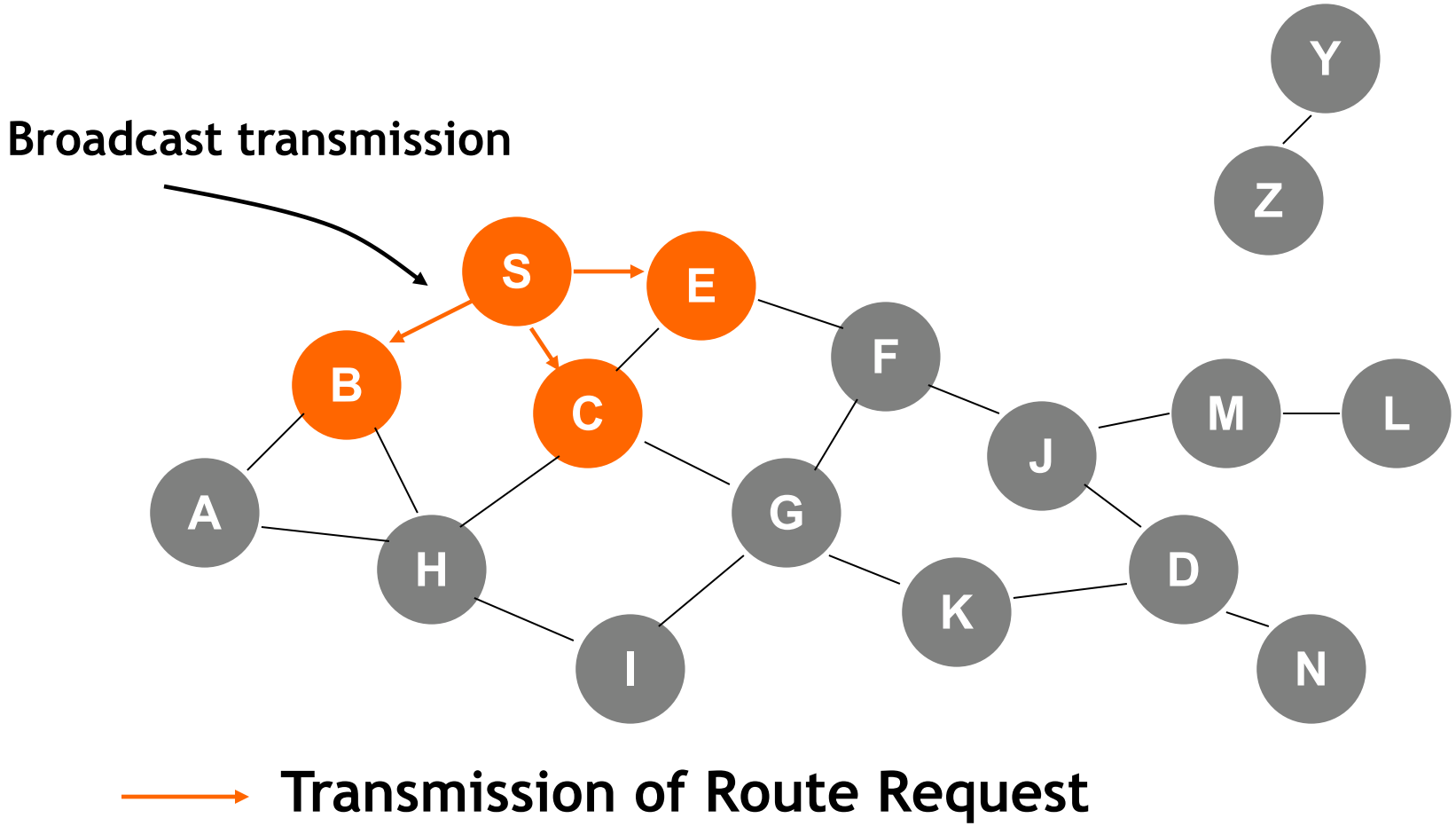
- In case of very large networks, the Route Request can have a Time to Live (TTL) field set to different values:
  - For example, initially, the TTL can be just sent to 1, and then successfully increased

Route Discovery: Transmission of Route Request

Institute for the Wireless Internet of Things at Northeastern

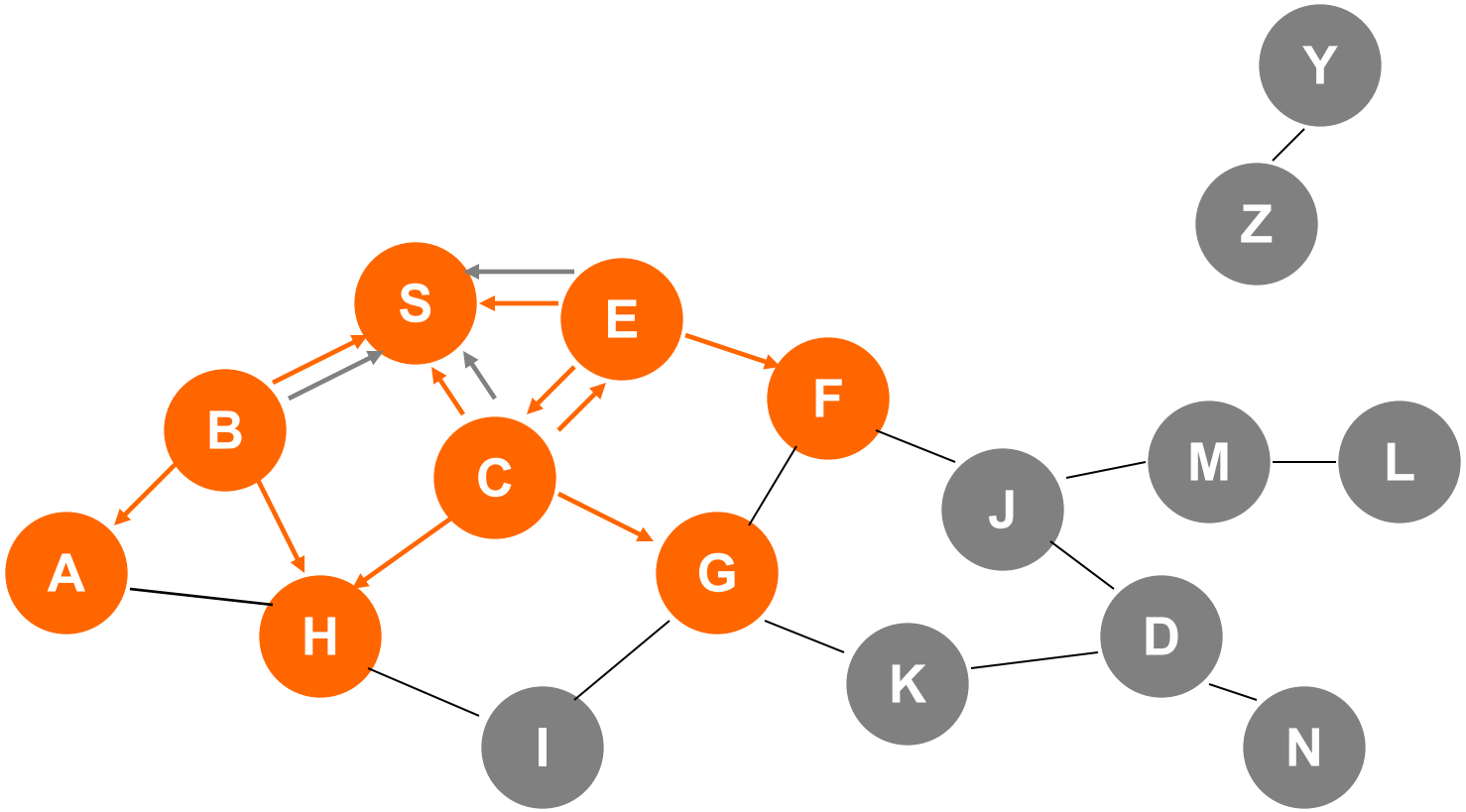


Route Discovery: Transmission of Route Request



Route Discovery: Transmission of Route Request

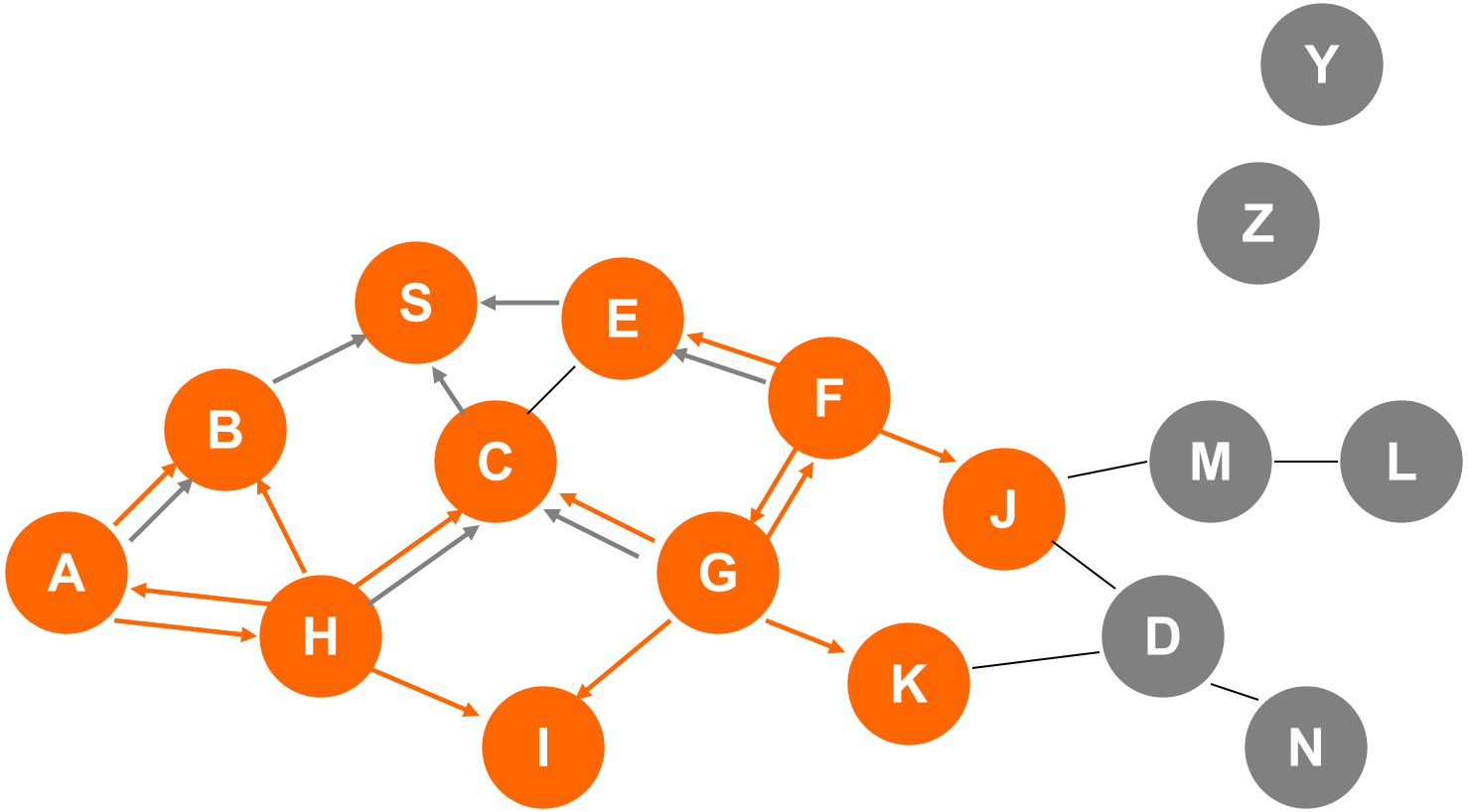
Institute for the Wireless Internet of Things at Northeastern



← Reverse Path (for the future)

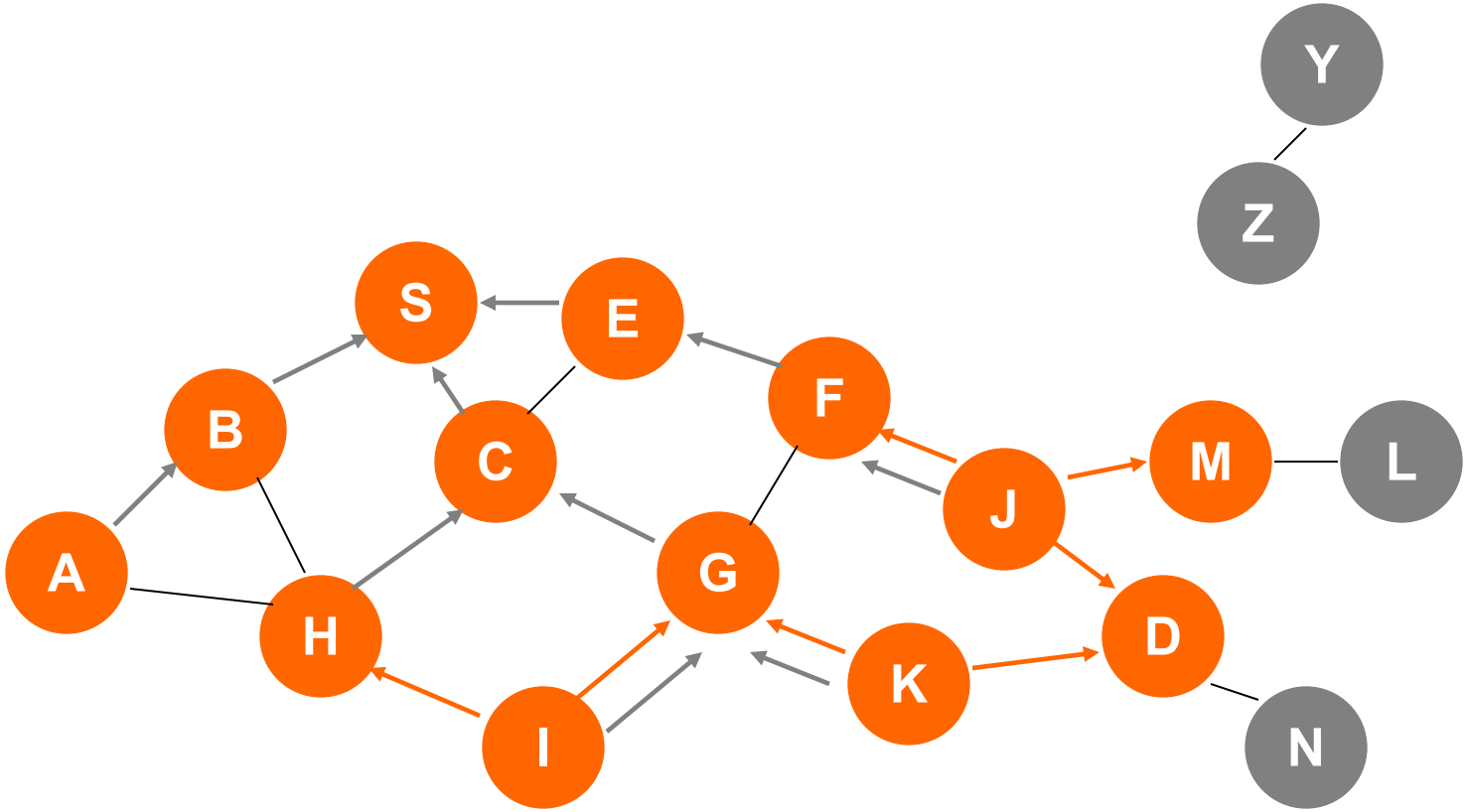
Route Discovery: Transmission of Route Request

Institute for the Wireless Internet of Things at Northeastern

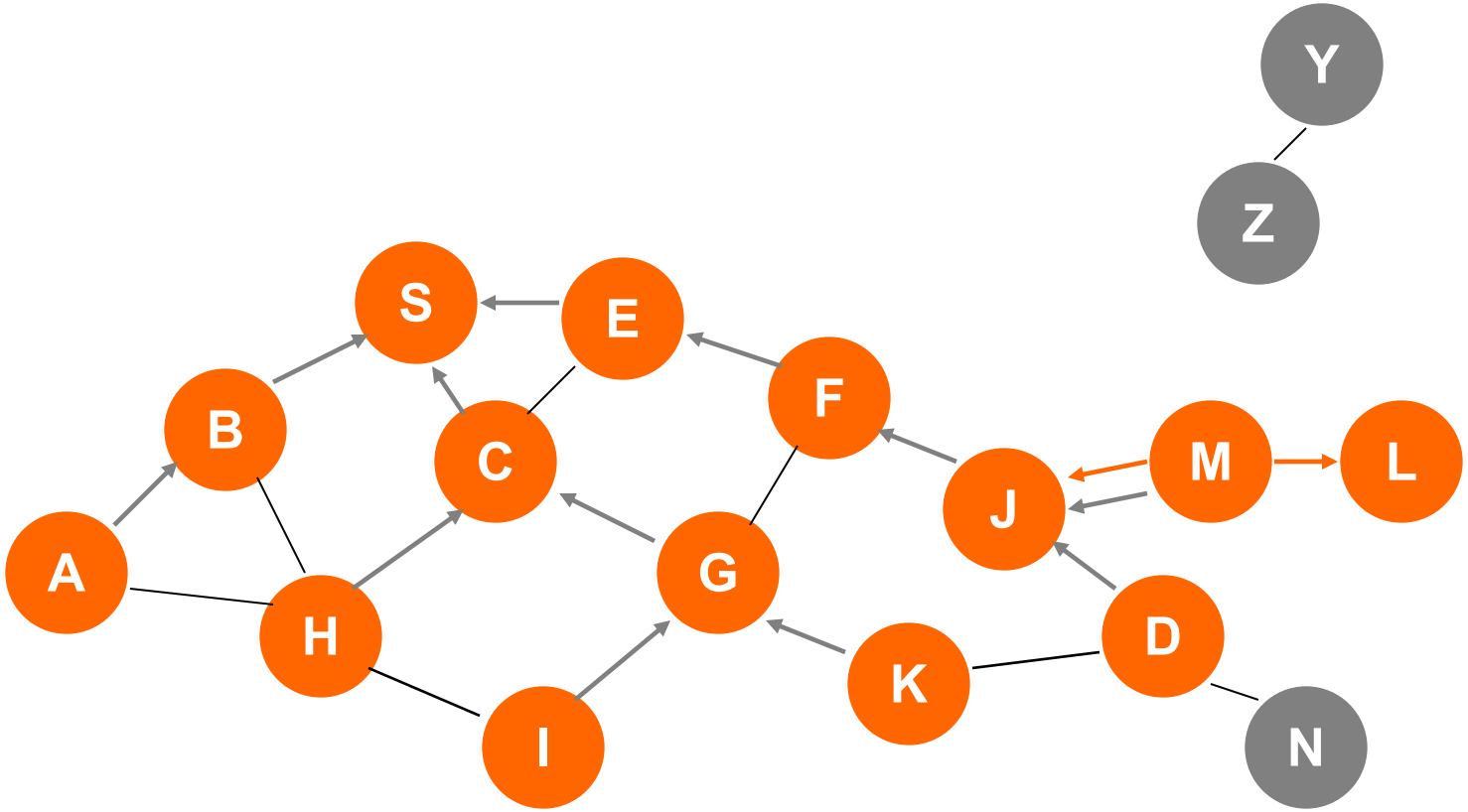


Route Discovery: Transmission of Route Request

Institute for the Wireless Internet of Things at Northeastern



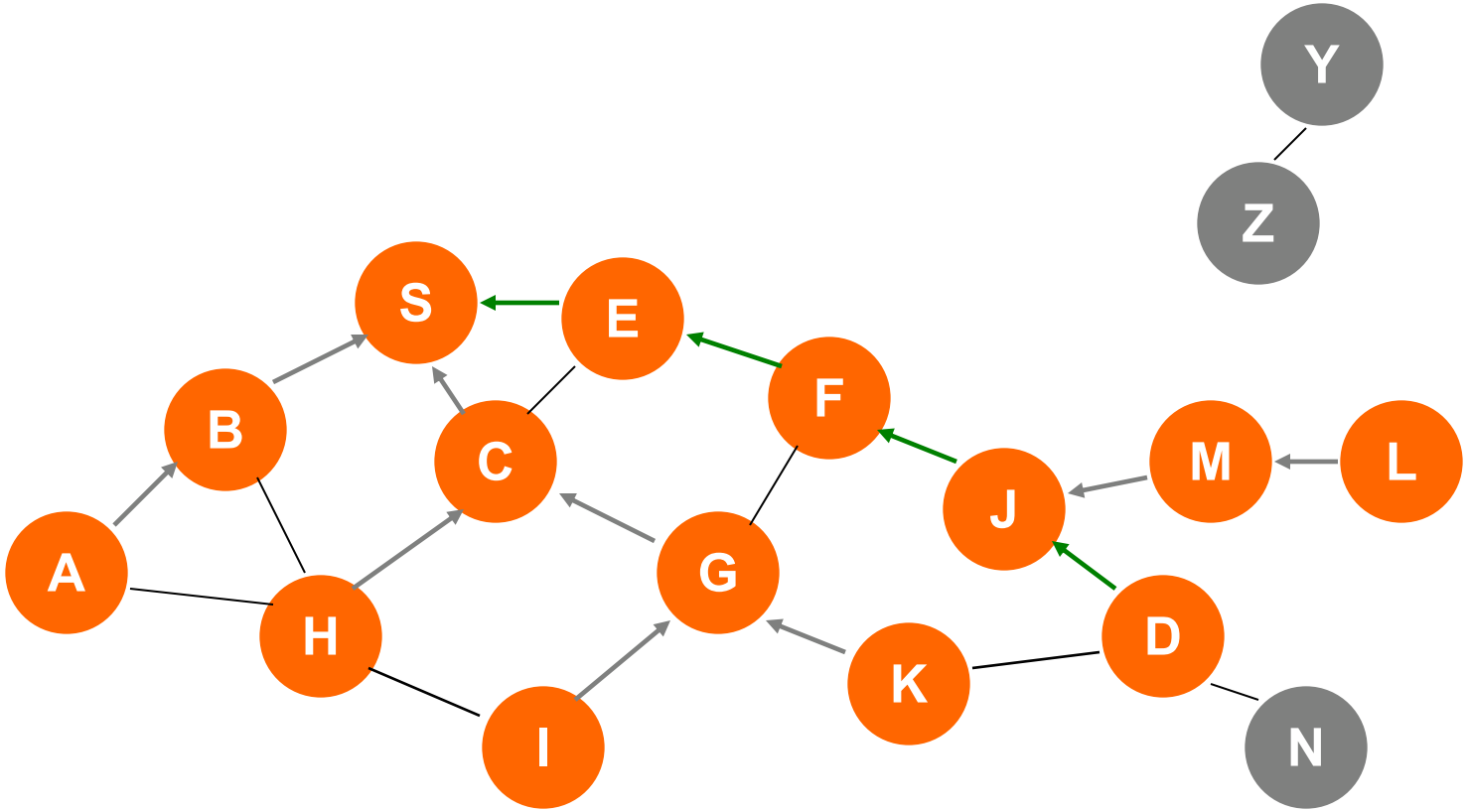
Route Discovery: Transmission of Route Request



- Node D does not forward the Route Request, because node D is the intended target of the packet

Route Discovery: Transmission of Route Reply

Institute for the Wireless Internet of Things at Northeastern



Represents links on path taken by the Route Reply

Route Maintenance

- The network topology can change spontaneously (e.g., nodes can be switched off, can move, etc.)
  - Routes might not be valid anymore!

- To maintain the routes, each node periodically broadcasts a **Hello** message to its neighbors

- If a neighbor does not reply, it is assumed the node is no longer there and routes through it need to be **purged**:
  - The node checks which of its routes involved this particular neighbor that has disappeared
  - The node informs to its neighbors that specific route is no longer valid
    - This is done recursively, hop by hop

AODV: Convergence

Institute for the Wireless Internet of Things at Northeastern

- To ensure rapid convergence, routes include a sequence number that is controlled by the destination
  - The destination increments it every time that it sends a fresh Route Reply

- For example, after having purged a route:
  - Senders ask for a fresh route by including in the Route Request the destination sequence number of the last route they used,
    - Either the sequence number of the route that was just purged or 0 as an initial value
  - The request is broadcasted until a route with a higher sequence number is found
  - Intermediate nodes store the routes that have a higher sequence number, or the fewest hops for the current sequence number

- **Check additional reading material and observe how it works in ns-3!**

Optimized Link State Routing (OLSR)

Institute for the Wireless Internet of Things at Northeastern

- Very similar to Link State Routing, just with some minor modifications to make it more suitable for mobile networks;

- **Remember:** each node computes its own routing table by collecting information of the entire network topology, which is periodically flooded

- **Proactive Routing:** The information in the routing table is kept as accurate as possible during all the time:
  - This is done to minimize the time between a packet is generated and the time the packet reaches the destination (lab assignment…)

- **Check additional reading material and observe how it works in ns-3!**

Protocols for Wireless Sensor Networks

Institute for the Wireless Internet of Things at Northeastern

- There are literally thousands of routing protocols for wireless sensor networks
- A possible classification:
  - **Location-based (Geographic) Protocols**
    - MECN, SMECN (Small Minimum-Energy Communication Network), GAF (Geographic Adaptive Fidelity), GEAR, Distributed Topology/Geographic Routing Algorithm (PRADA)
  - **Data Centric Protocols**
    - Flooding, Gossiping, SPIN, Directed Diffusion, SAR (Sequential Assignment Routing) , Rumor Routing, Constrained Anisotropic Diffused Routing, COUGAR, ACQUIRE
  - **Hierarchical Protocols**
    - LEACH, PEGASIS, TEEN (Threshold Sensitive Energy Efficient Sensor Network Protocol), APTEEN,
- Once upon a time, this was a very popular research field… but really that's where it ended.

Technologies for the IoT

Institute for the Wireless Internet of Things at Northeastern

- There are many technologies to enable the connectivity of "Things" to the Internet:
  - **Network layer and above:**
    - Routing Protocol for Low-Power and Lossy Networks (RPL)
      - And many variations:
        - CORPL – RPL for Cognitive Radio Networks
        - CARP – Similar to RPL but tailored to underwater networks
    - IPv6 over…
      - IPv6 over Low-power Wireless Personal Area Networks (6LoWPAN)
      - IPv6 over Bluetooth Low Energy
      - IPv6 over ... you named it
    - (1000s of routing protocols for Wireless Sensor Networks)

Remember: Routing in the IoT

Institute for the Wireless Internet of Things at Northeastern

- Energy consumption is a major issue (for battery powered "things")
- Limited processing power
- Very dynamic topologies:
  - Link failure (due to wireless)
  - Node failures (due to low-cost)
  - Node mobility (in some environments)
- Data processing usually required on the node itself
- Sometimes deployed in harsh environments
- Potentially deployed at very large scale
- Must be self-managed (auto-discovery, self-organizing networks)

RPL ("ripple"): Routing Protocol for Low Power and Lossy Networks

- **IETF RFC 6550**
  - Short for: **Internet Engineering Task Force** Request for Comments 6550
  - Approved March 2012
- **Key features:**
  - Proactive distance-vector routing
  - Source routing
  - Does not rely on any particular feature of a specific link layer technology
  - Supports:
    - Multipoint-to-point traffic from devices inside the low-power and lossy network (LLN) towards a central control point
    - Point-to-multipoint traffic from the central control point to the devices inside the LLN are supported
    - Point-to-point traffic between devices

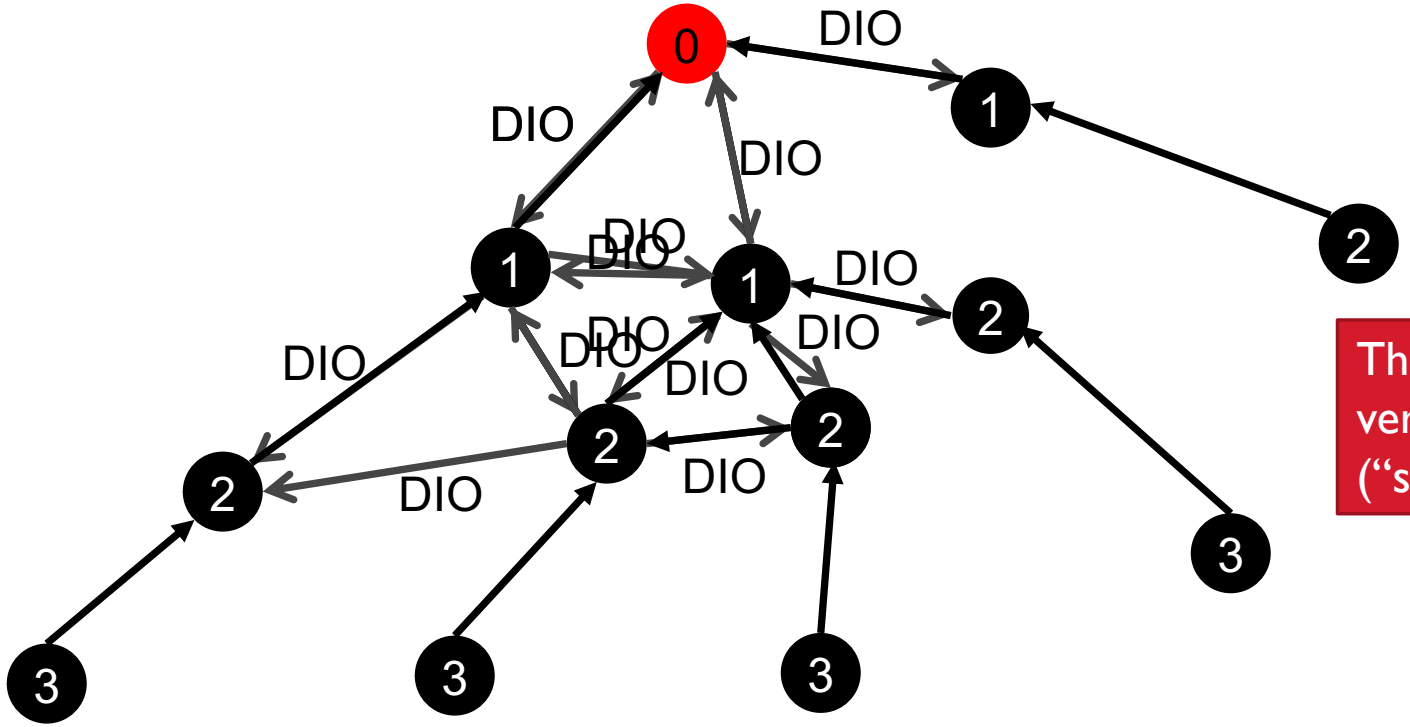
Basics of RPL

- Routing based on graph construction
  - **DAG:** Direct Acyclic Graph
  - **DODAG:** Destination Oriented DAG

- New types of ICMP messages (Internet Control Message Protocol)
  - **DIS:** DODAG Information Solicitation
  - **DIO:** DODAG Information Object
  - **DAO:** Destination Advertisement Object

RPL Operation

Institute for the Wireless Internet of Things at Northeastern



The rest of the protocol is just very similar to AODV/OLSR ("salt and pepper"

- Directed Acyclic Graph (DAG) Information Option (DIO) messages are broadcasted to build the tree
  - Includes a node's rank (its level), expected transmission count (ETC), etc.
- ETX probe is sent periodically to probe neighboring ETX

The Internet Protocol (IP)

- The glue that holds the whole Internet together
  - Designed with internetworking in mind
- **Objective:** To provide a **best-effort** (i.e., not guaranteed) way to transport packets from source to destination…
  - …without regard to whether these machines are on the same network or whether there are other networks in between them
- **How does it work?**
  - The transport layer takes data streams and breaks them up so that they may be sent as IP packets
    - Packets can be up to 64 KB each, but in practice they are usually not more than 1500 bytes (so they fit in one Ethernet frame)
  - IP routers forward each IP packet through the Internet, along a path from one router to the next, until the destination is reached
    - These might need to be fragmented at some point
  - When a packet finally gets to the destination machine, the network layer reassembles the packet if it has been fragmented, and sends it "up" to the transport layer

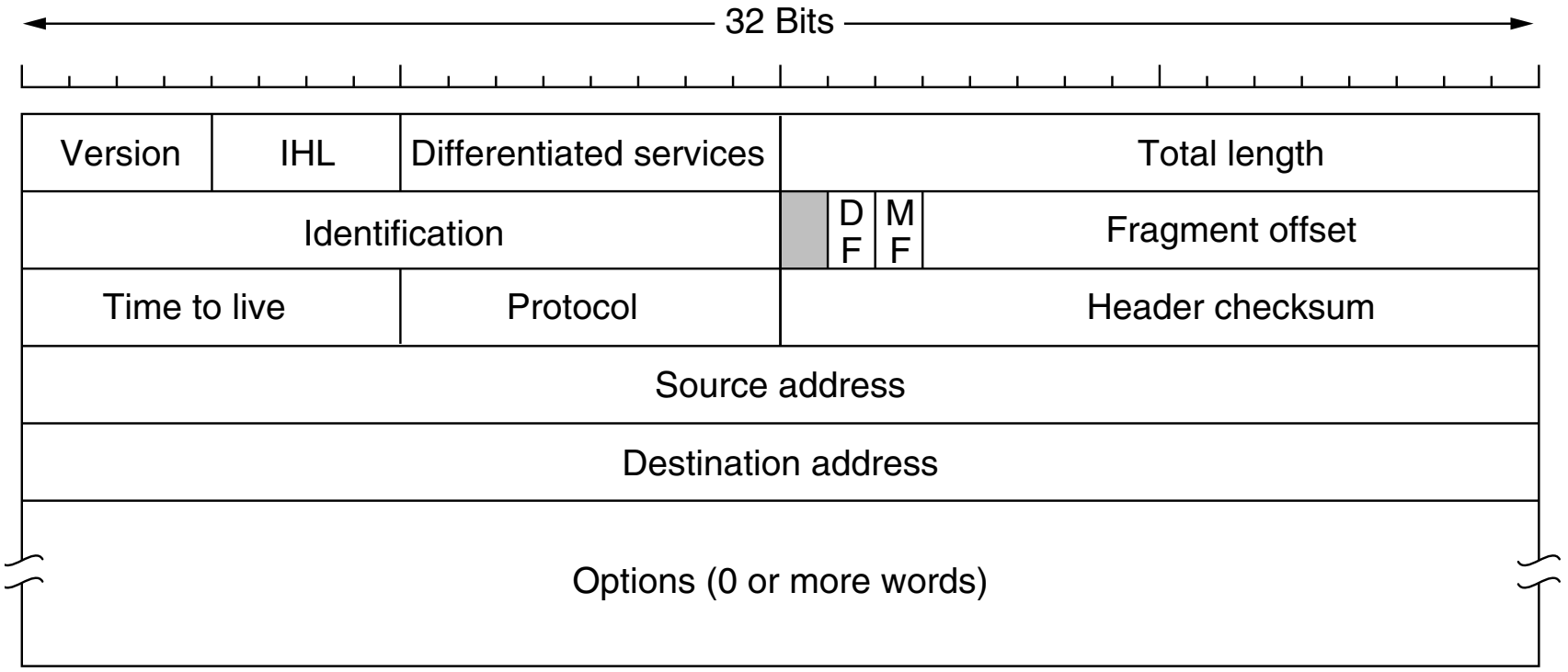
The IP Version 4 Protocol (IPv4)

Institute for the Wireless Internet of Things at Northeastern

- The first "major version" of IP and currently the dominant protocol of the Internet:
  - What happened to version 1, 2 and 3?
    - This is related to the history of TCP:
      - TCPv1 designed in 1973
      - TCPv2 designed in 1977
        - These two versions were doing two things:
        - 1. Host level end-to-end protocol
        - 2. Internet packing and routing protocol
        - SPLIT!
      - TCPv3 and IPv3 in 1978
      - **IPv4 in 1981 (RFC 791)**

IPv4: Packet Structure

- **Header:** 20-byte fixed-length part + variable-length part



IPv4: Packet Structure

- Header:
  - **Version:** 4, 5?, 6, …
  - **Internet Header Length (IHL):** Length of the header field in 32-bit words (up to 15*32/8=60 bytes)
  - **Differentiated Services (formerly called *type of service*):**
    - Originally, 3 bits specify whether a host cares more about delay, throughout or reliability + 3 bits to specify priority
    - Now, 4 bits describe type of service (e.g., expedited, assured, etc.) + 2 bits for congestion notification
  - **Total Length:** Includes both header and data, currently up to 65535 bytes (maybe more in the future)
  - **Identification:** Identifies which packet a fragment belongs to
  - 1 empty bit
  - **Don't Fragment (DF)**
  - **More Fragments (MF)**
  - **Fragment Offset:** To reassemble the packet

IPv4: Packet Structure

- **Time To Live (TTL):** Currently it just counts hops (used to count seconds)
- **Protocol:** Tells the network layer what to do with the packet (e.g., TCP, UDP, etc.)
- **Header Checksum:**
  - Adds all the 16-bit words in the header
  - Needs to be recomputed at every step, as the header changes
- **Source Address / Destination Address:** IP addresses
- **Options:**
  - **Security:** Information not to be routed through non-trustworthy countries
  - **Strict Source Routing:** Specifies the complete path from source to the destination, i.e., the list of IP addresses the packet should traverse, and not any other route
  - **Loose Source Routing:** List of mandatory IPs, but others, in addition to these, are fine too
  - **Record Route:** to be able to trace the route that the packet followed
  - **Timestamp:** 32-bit time stamp in addition to each address in the path
  - The truth is that many routers ignore all these options…

IPv4: Addresses

- 32-bit addresses
  - 2^{32} total addresses ~ 10^9 addresses in total…

- Every host and router on the Internet has an IP address that can be used in the *Source address* and *Destination address* fields of IP packets

- **Observation:** An IP address does not actually refer to a host but it really refers to a network interface:
  - If a host is on two networks, it must have two IP addresses
    - Most hosts are on one network and thus have one IP address
    - Routers have multiple interfaces and thus multiple IP addresses

IPv4: Prefixes

- IP addresses are hierarchical, unlike MAC addresses

- Each 32-bit address is comprised of two parts:
  - A variable-length **network portion** in the top bits
    - This is called **the prefix** and has the same value for all hosts on a single network, e.g., in the same Ethernet LAN or WLAN
  - A variable-length **host portion** in the bottom bits
    - This is different for the different users in the network

- IP addresses are written in **dotted decimal notation**
  - In this format, each of the 4 bytes is written in decimal, from 0 to 255
    - E.g., 192.168.0.1, 10.0.0.1, etc.

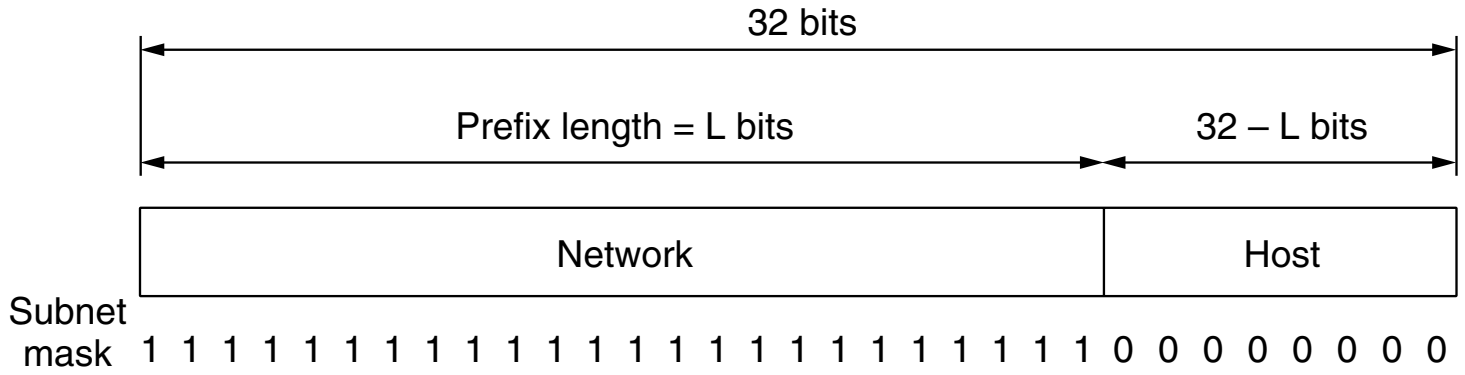
IPv4: Prefixes

- Prefixes are written by giving the lowest IP address in the block and the **size of the block**
  - The size is determined by the number of bits in the network portion
  - By convention, it is written after the prefix IP address as a slash followed by the length in bits of the network portion
    - E.g., if the prefix contains 2^8 addresses and so leaves 24 bits for the network portion, it is written as 192.168.1.0/24

- The prefix length cannot be inferred from the IP address alone
  - Routing protocols must carry the prefixes to routers

IPv4: Prefixes

- The length of the prefix corresponds to a binary mask of 1s in the network portion
  - This is why sometimes it is called the **subnet mask**
  - It can be ANDed with the IP address to extract only the network portion



- E.g., 192.168.1.107/24:
  - Subnet mask: 255.255.255.0 (11111111.11111111.11111111.00000000)
  - Network address: 192.168.1.0
  - Host address inside the network: 107

IPv4: Prefixes

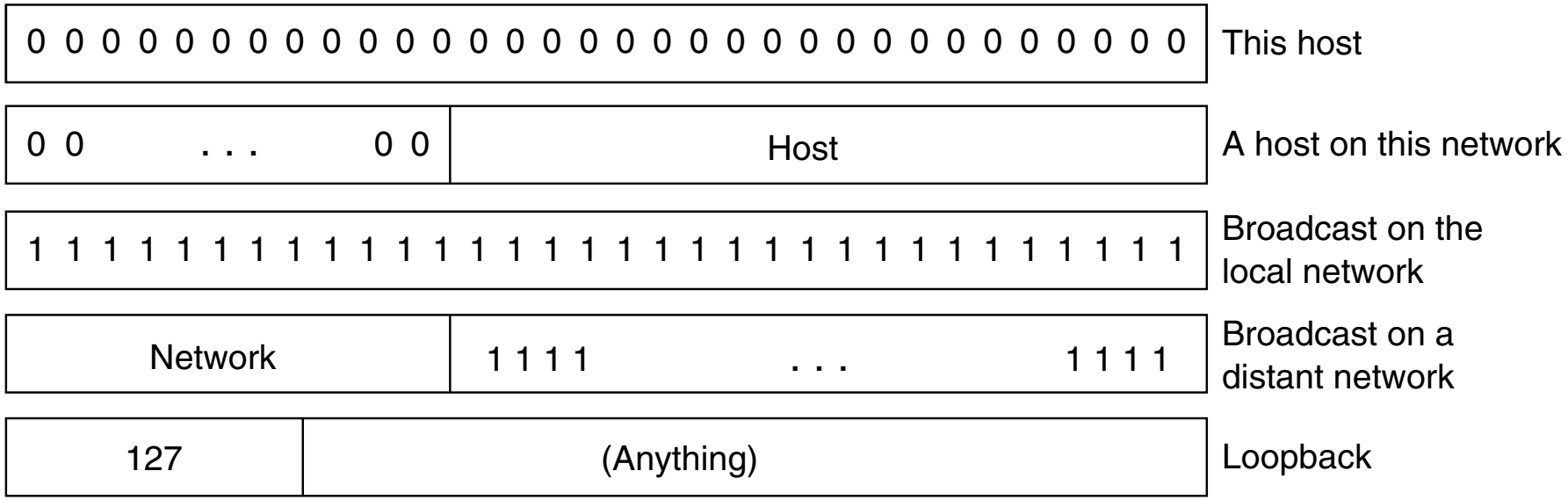
- **Advantages of prefixes:**
  - Routers can forward packets based on only the network portion of the address, as long as each of the networks has a unique address block
    - The host portion does not matter to the routers because all hosts on the same network will be sent in the same direction
    - It is only when the packets reach the network for which they are destined that they are forwarded to the correct host
    - This makes the routing tables much smaller than they would otherwise be
    - E.g., consider that the number of hosts on the Internet is above 6-7 billion
      - That would be a very large table for every router to keep!
      - However, by using a hierarchy, routers need to keep routes for only around 300,000 prefixes…

IPv4: Prefixes

- **Disadvantages of prefixes:**
  - The IP address of a host depends on where it is located in the network:
    - Every IP address belongs to a specific network, and routers will only be able to deliver packets destined to that address to the network
      - Remember that, on the contrary, an Ethernet address is unique in the world!
    - Designs such as **mobile IP** are needed to support hosts that move between networks but want to keep the same IP addresses
  - Hierarchical addressing can be a waste of addresses unless it is carefully managed
    - If addresses are assigned to networks in (too) large blocks, there will be (many) addresses that are allocated but not in use
    - This allocation would not matter much if there were plenty of addresses to go around
    - However, it was realized more than two decades ago that the tremendous growth of the Internet was rapidly depleting the free address space

IPv4: Special Addresses

- Some addresses have a special meaning:



IPv4: Address Scarcity Problem

Institute for the Wireless Internet of Things at Northeastern

- **Problem:** IP addresses are scarce → They need to be use sparingly

- **Solution 1: Dynamic IP assigned by the ISP:**
  - An IP address is assigned to a host when it is on and using the network
  - The IP address is taken back when the host becomes inactive
  - The IP address can then be assigned to another computer that becomes active
  - Disadvantage: many devices are active all the time and thus require an IP address all the time

- **Solution 2:** Move to IPv6
  - Uses 128-bit long addresses (we will talk about it later)

- **Solution 3:** Network Address Translation

IPv4: Network Address Translation (NAT)

Institute for the Wireless Internet of Things at Northeastern

- **Basic idea:** The ISP assigns each home/business a single IP address for Internet traffic:
  - Within the customer network, every computer gets a unique IP address, which is used for routing internal traffic
  - Just before a packet exits the customer network and goes to the ISP, an address translation from the unique internal IP address to the shared public IP address takes place
  - This translation makes use of three ranges of IP addresses that have been declared as private:
    - Networks may use them internally as they wish
    - The only rule is that no packets containing these addresses may appear on the Internet itself

10.0.0.0	10.255.255.255/8	(16,777,216 hosts)
172.16.0.0	172.31.255.255/12	(1,048,576 hosts)
192.168.0.0	192.168.255.255/16	(65,536 hosts)

The Internet Protocol Version 6 (IPv6)

- Approved in 1998 (RFC 2460)
  - But only used by 4.65% of the Internet traffic as of October 2014

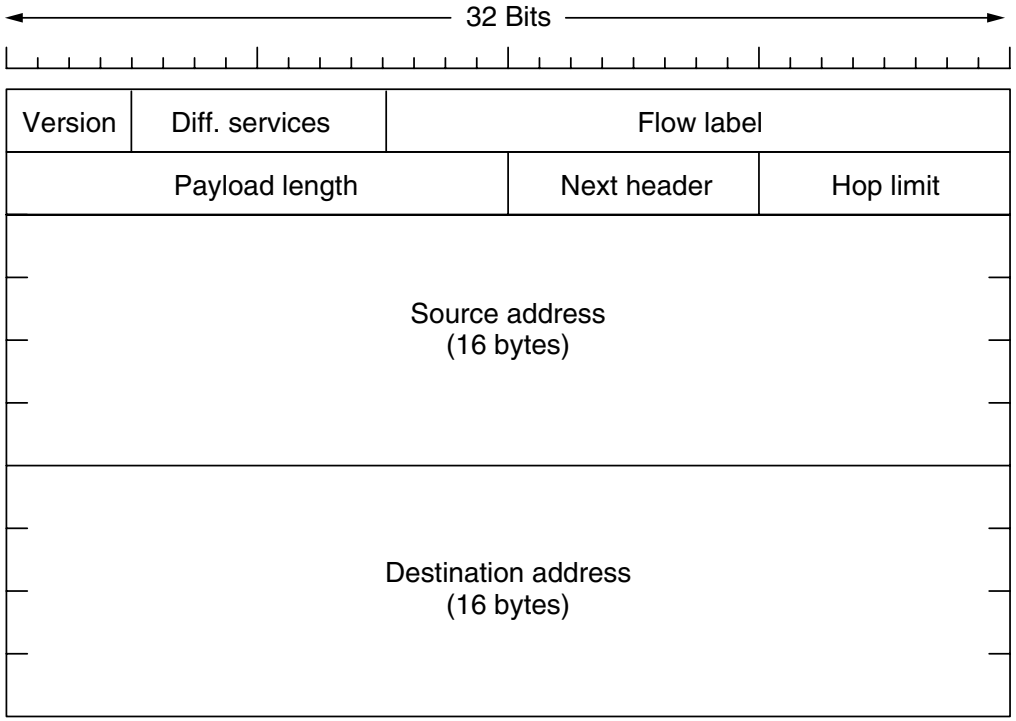
- **Where is IPv5?**
  - This version was skipped on purpose to avoid confusion with the Internet Stream Protocol (ST)
    - Originally envisioned to be a connection-oriented complement to IPv4
    - Marked its packets with an Internet Protocol version 5
      → Not really known as IPv5…

IPv6: Features

- Addresses are 128-bit long:
  - Can address up to 2^{128} devices, i.e., $+10^{38}$ hosts!
- Much shorter header with 7 fields instead of 13:
  - Allows routers to process packets faster and thus improves throughput and delay
- Better support for options:
  - Many of the IPv4 fields are now optional
  - The way options are represented is different → Simpler for routers to skip over options not intended for them
- Big advance is in security:
  - Authentication and privacy are now key features
- Enhanced Quality of Service (QoS)

IPv6: Packet Header

- **Required:** 40 bytes
- **Optional:** depends on the optional header



IPv6: Packet Header Fields

Institute for the Wireless Internet of Things at Northeastern

- **Differentiated Services:** Used to distinguish the class of service for packets with different real-time delivery requirements (for QoS)
- **Flow Label:** Allows a source and destination to mark groups of packets that have the same requirements and should be treated in the same way by the network, forming a pseudo-connection
- **Payload Length:** without counting the 40 header bytes (as it used to be)
- **Next Header:** tells which of the (currently) six extension headers, if any, follow this one
  - This is how the header could be simplified!
- **Hop Limit:** as the TTL in IPv4, decremented in every hop
- **Source and Destination Addresses:**
  - Written as eight groups of four hexadecimal digits with colons between the groups

IPv6: Missing Fields

Institute for the Wireless Internet of Things at Northeastern

- Fragmentation is handled with an optional header
  - Only a source router can fragment IPv6 packets
  - The procedure described in Slide 20/21 is followed for the other routers

- There is no checksum:
  - Not needed: link layer has error control link by link and transport layer has error control end-to-end…

- IHL: not needed, the header has a fixed length now

IPv6: Extension Headers

- IPv6 uses optional extension headers
  - Provide additional information, encoded in an efficient way

Extension header	Description
Hop-by-hop options	Miscellaneous information for routers
Destination options	Additional information for the destination
Routing	Loose list of routers to visit
Fragmentation	Management of datagram fragments
Authentication	Verification of the sender's identity
Encrypted security payload	Information about the encrypted contents

- Check additional reading materials…

Why IP for the IoT?

Institute for the Wireless Internet of Things at Northeastern

- **Extensive interoperability**
  - Other wireless embedded 802.15.4 network devices
  - Devices on any other IP network link (WiFi, Ethernet, GPRS, Serial lines, ...)
- **Established security**
  - Authentication, access control, and firewall mechanisms
  - Network design and policy determines access, not the technology
- **Established naming, addressing, translation, lookup, discovery**
- **Established proxy architectures for higher-level services**
  - NAT, load balancing, caching, mobility
- **Established application level data model and services**
  - HTTP/HTML/XML/SOAP/REST, Application profiles
- **Established network management tools**
  - Ping, Traceroute, ...
- **Transport protocols**
  - End-to-end reliability in addition to link reliability
- **Most "industrial" (wired and wireless) standards support an IP option**

Why IPv6 for the IoT?

Institute for the Wireless Internet of Things at Northeastern

- **Pros**
  - More suitable for higher density (futuristically 2 orders of magnitude larger than traditional networks)
    - IPv6 Addresses 128 bits
    - IPv4 not enough 32 bits
  - No NAT necessary
  - Possibility of adding innovative techniques such as location aware addressing
- **Cons**
  - Larger address width (having efficient address compression schemes may alleviate this con)
  - Complying to IPv6 node requirements (IPSec is mandated)

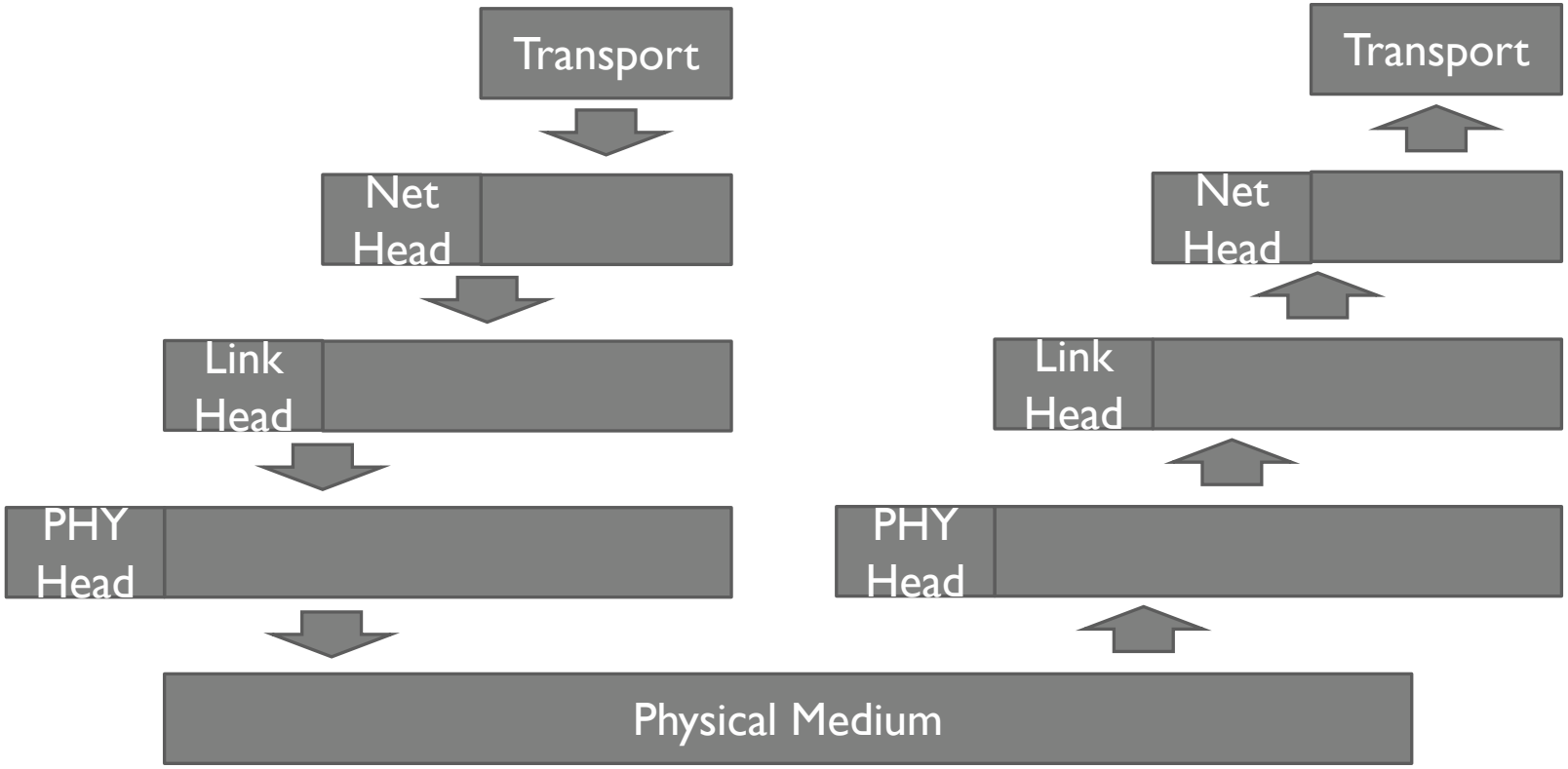
Need for Encapsulation Protocols

Institute for the Wireless Internet of Things at Northeastern

- IPv6 addresses are too long and cannot fit in most IoT datalink frames which are relatively much smaller
  - Standards to encapsulate IPv6 datagrams in different datalink layer frames for use in IoT applications are needed

- Example: **6LoWPAN Format**
  - Encapsulates IPv6 long headers in IEEE 802.15.4 (LoWPAN) small packets
    - Cannot exceed 128 bytes

Remember


Transport
Net Head
Link Head
PHY Head
Transport
Net Head
Link Head
PHY Head
Physical Medium

Problems with IP over LoWPAN

- No method exists to make IPv6 run over IEEE 802.15.4 networks
- Stacking IP and above layers "as is" may not fit within one 802.15.4 frame
  - IPv6 40 bytes, TCP 20 bytes, UDP 8 bytes + other layers (security, routing, etc) leaving few bytes for data
- Not all ad hoc routing protocols may be immediately suitable for LoWPAN
  - The routing vectors may not fit within a packet!
- Security for multi hop needs to be considered

6LoWPAN

Institute for the Wireless Internet of Things at Northeastern

- It was a working group, created different protocols/recommendations:
  - **RFC 4944:** packet format, header compression, addresses compression
  - **RFC 6282:** enhancements to header compression
  - **RFC 6775:** neighbor discovery

  - ***All available in Canvas!***

Next Steps

Institute for the Wireless Internet of Things at Northeastern

- Research paper:
  - Due on November 20
- Second computer laboratory assignment:
  - Due on November 15

- Coming soon:
  - Homework assignment (numerical)
  - Online (open-book) quiz (multiple choice questions)