Dynamic Routing Algorithms

CT3531 – Nets & Comms 2

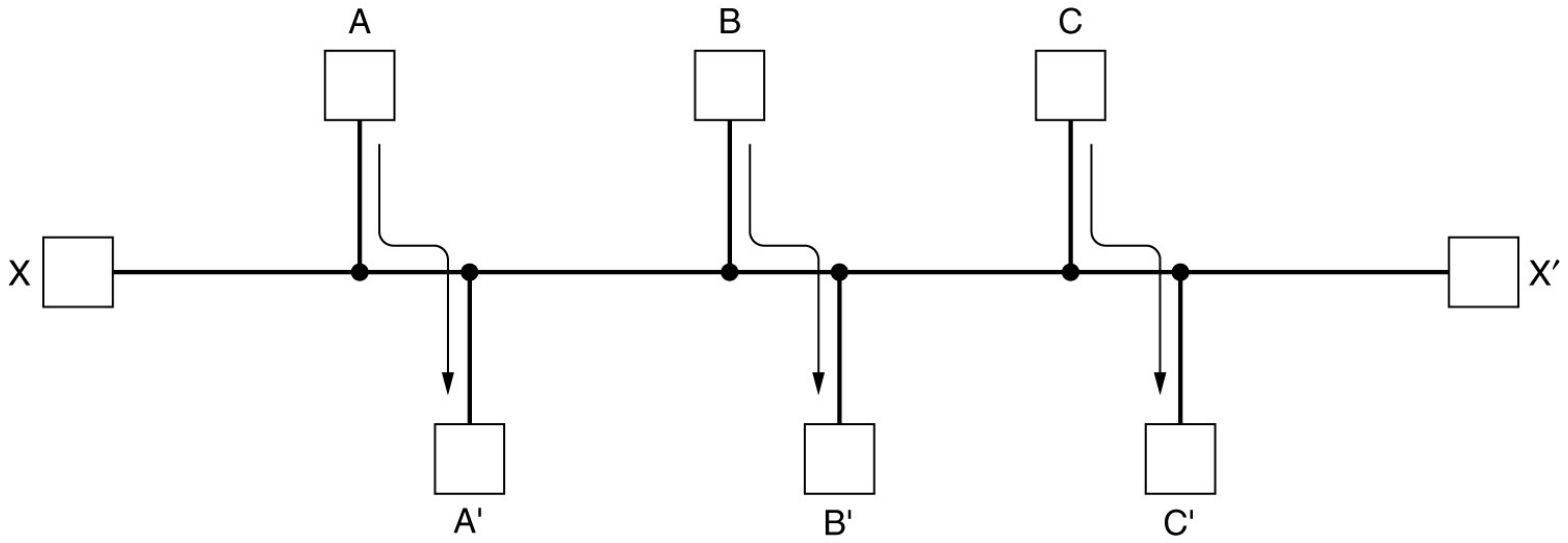
Content

- Routing Algorithms
  - Static routing
    - Shortest path routing
    - Flooding
  - Dynamic routing
    - Distance vector routing
    - Link state routing
    - Introduction to OSPF
  - Hierarchical routing

Routing Algorithms

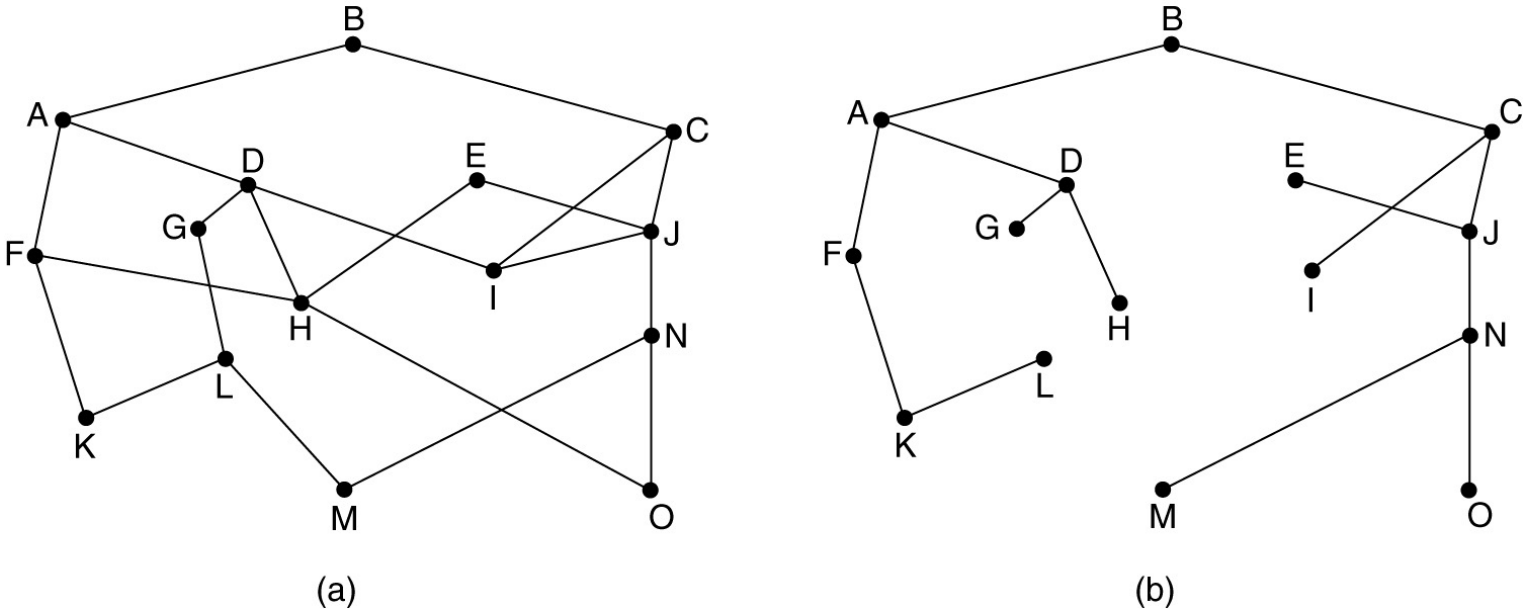
- A router can be seen as a device that has two processes inside:
  - Forwarding process – handles each packet as it arrives, looking up for the outgoing line to use for it.
  - The other process is responsible for filing in and updating the routing tables; this is where the **routing algorithm** comes into play
- Certain properties are desirable for a routing algorithm: correctness, simplicity, robustness, stability, fairness and optimality;

Conflict between fairness and optimality



- Fairness and optimality may sound obvious, but they are often contradictory goals
  - Suppose there is enough traffic between A and A’, B and B’, C and C’ to saturate the horizontal links. To maximize the total flow, the XX’ traffic should be shut down completely
  - Evidently, some sort of compromise between global efficiency and fairness to individual connections is needed

The Optimality Principle



- If router J is on the optimal path from router I to router K, then the optimal path from J to K follows the same route
- Direct consequence of the optimality principle is that we can see that all optimal routes from all sources to a given destination form a tree rooted at the destination. This tree is called **sink tree**
  - The tree in the figure is a sink tree for router B in the subnet, where the metric is the number of hops
- ***The goal of all routing algorithms is to discover and use the sink tree for all routers***

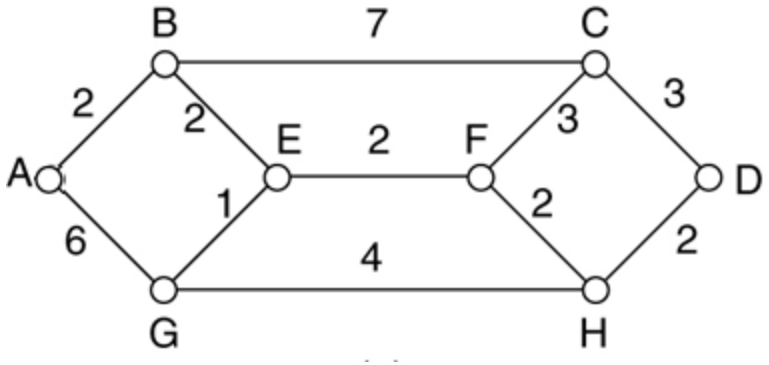
Routing Algorithms Types

- Static (non adaptive) Routing Algorithms
  - Don't base their routing decisions on measurements or estimates of the current traffic and/or topology
  - The choice of the route to use from I to J is computed in advance (offline) and downloaded into the routers at the network boot time.
- Dynamic (adaptive) Routing Algorithms
  - They change their routing decisions to reflect changes in topology and usually changes in traffic as well
  - They differ how they get their information (locally, from adjacent router, from all routers), when they change the routes or what metrics they use for optimization (distance, number of hops, estimated transit time, etc)

Flooding routing

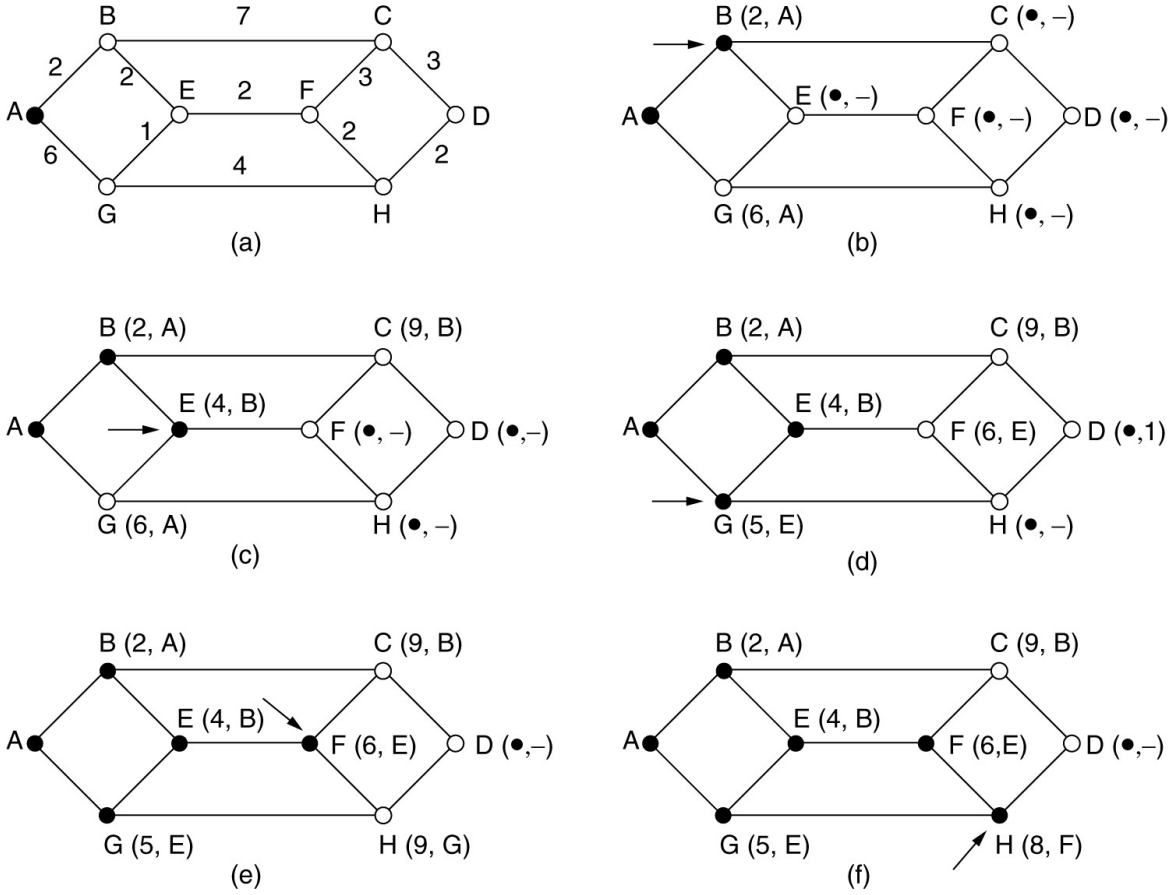
- Static algorithm, in which every incoming packet is sent out on every outgoing line except the one it arrived on
- It generates a vast number of duplicate packets; therefore, some measures must be taken to damp the process:
  - Have a hop counter contained in the header (decremented by each router), with the packet being discarded when counter reaches 0)
  - Keep tracking of which packets have been flooded, so flooding again will be avoided
- Selective flooding – the routers don't send every incoming packet out on every line, but only on those lines that are going in the approximate right direction
- Flooding is not highly practical in most applications, but it does have some use in applications where tremendous robustness is highly desirable (i.e. military applications, where routers are deployed at once, radio networks, etc..)
- It can be used as a metric against which other routing algorithms can be compared. Flooding always chooses the shortest path, because it chooses every possible path in parallel. Consequently, no other algorithm can produce a shorter delay (if we ignore the overhead…)

Shortest path routing



- The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line (link)
- To choose a route between a given pair of routers, the algorithm needs to find the shortest path between them on the graph
- Metric:
  - Number of hops: ABC and ABE are equally long
  - Geographic distance: ABC is clearly much longer than ABE, assuming we have the diagram at scale
  - Other metrics: i.e. each arc could be labeled with the mean queuing and transmission delay for some standard test packet as determined by hourly test runs; with this graph labeling, the shortest path is the fastest path rather than the path with the fewest hops or kilometers
- The labels on the arcs could be computed as a function of many factors: distance, average traffic, bandwidth, communication cost, mean queue length, measured delay and other factors. By changing the criteria, the algorithm will then compute the shortest path according to the measuring criteria or combination of criteria.

Dijkstra algorithm for computing shortest path

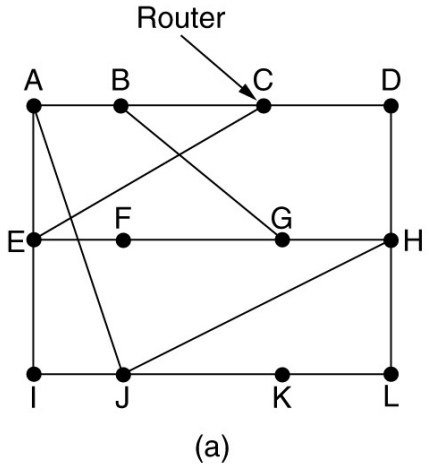


- Each node is labeled with its distance from the source node along the best-known path. Initially no paths are known, so all nodes are labeled with infinity
- As the algorithm proceeds, and paths are found, the label may change reflecting better paths
- A label may be either tentative or permanent, initially all labels are tentative; when it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed after that.

Distance Vector Routing

- Computer networks usually use dynamic routing algorithms, since the static ones don't take in calculus the network load
- It was used in ARPANET and it is sometimes used in Internet under the name RIP
- Each router maintains a table (i.e. vector) giving the best known distance to each destination and which line to use to get there. These tables are updated by changing info with the neighbors. The routing table contains an entry for each router in the subnet. This entry contains two parts:
  - Preferred outgoing line to use for that destination
  - The estimation of the time or distance to that destination (the used metric can be number of hops, time delay in milliseconds, total no. of packets queued along that line, or something similar)
- The router is assumed to know the distance to each of its neighbors.
  - If metric is hops, the distance is just one hop…
  - If it is time, the router can measure it directly with special echo packets…
  - If it is queue length, the router examines each of its queues…

Distance vector routing



(a)

New estimated delay from J

To	A	I	H	K	Line
A	0	24	20	21	8 A
B	12	36	31	28	20 A
C	25	18	19	36	28 I
D	40	27	8	24	20 H
E	14	7	30	22	17 I
F	23	20	19	40	30 I
G	18	31	6	31	18 H
H	17	20	0	19	12 H
I	21	0	14	22	10 I
J	9	11	7	10	0 –
K	24	22	22	0	6 K
L	29	33	9	9	15 K

JA delay is 8 | JI delay is 10 | JH delay is 12 | JK delay is 6

Vectors received from J's four neighbors

New routing table for J

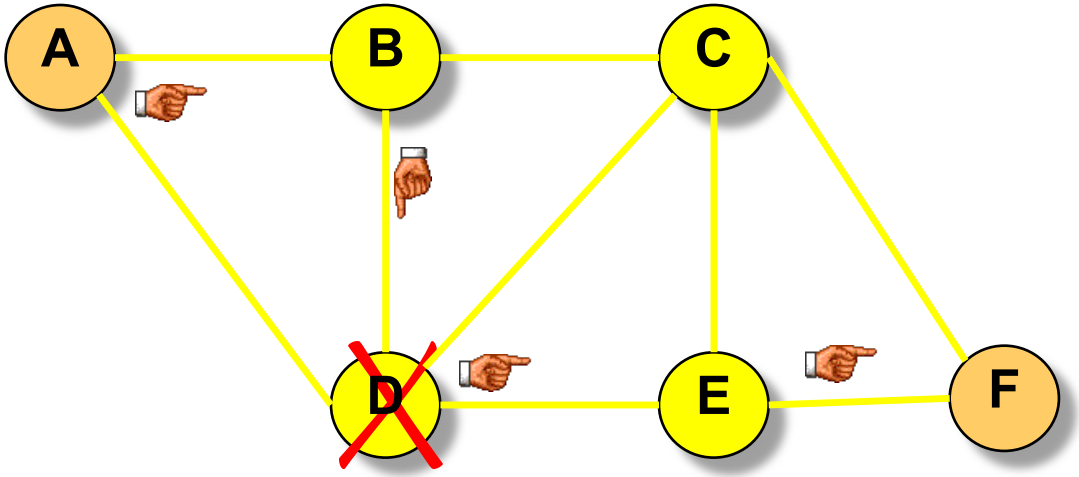
(b)

- J measures the delays to its neighbors
- J computes its new routes to router G:
  - It knows it can get to A in 8ms and A claims to get to G in 18ms, so J knows it can count on packets with 26ms to G if it routes the packets through A.
  - Similarly it computes delays to G via I (10 +31), via H (12 + 6) and via K (6 + 31). The best of these values is 18 so it makes an entry in its routing table that the delay to G is 18 ms and the route to use is via H.
- Same calculation is performed to all other destinations

Distance Vector vs. Link State Routing

- With distance vector routing, each node has information only about the next hop:
  - Node A: to reach F go to B
  - Node B: to reach F go to D
  - Node D: to reach F go to E
  - Node E: go directly to F

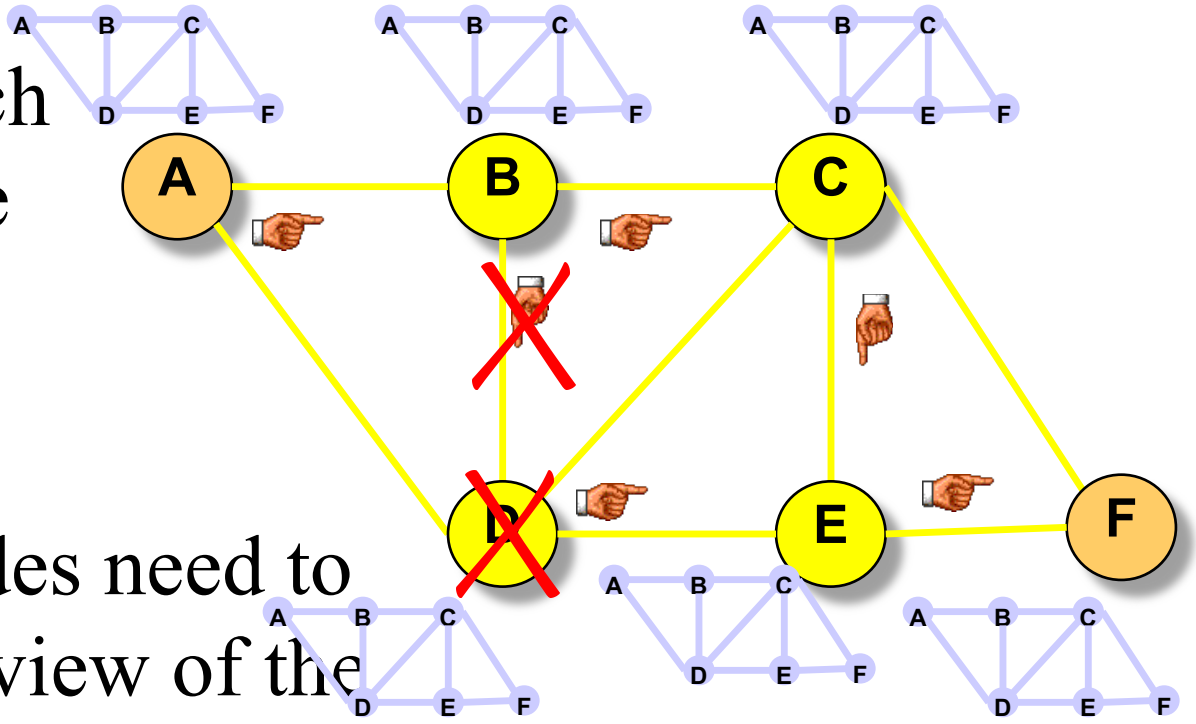
- Distance vector routing makes poor routing decisions if directions are not completely correct
(e.g., because a node is down).



- If parts of the directions incorrect, the routing may be incorrect until the routing algorithms has re-converged.

Distance Vector vs. Link State Routing

- In link state routing, each node has a complete map of the topology
- If a node fails, each node can calculate the new route
- Difficulty: All nodes need to have a consistent view of the network



Link state routing

- Distance vector routing was replaced by link state routing for two reasons:
  - The delay metric was queue length, it didn't take in calculus the line bandwidth into account
  - The algorithm took to long to converge
- Each router must do the following:
  - Discover their neighbors and learn their net addresses
  - Measure the delay or cost to each of their neighbor
  - Construct a packet telling all it has just learned
  - Send this packet to all other routers
  - Compute the shortest path to every other router

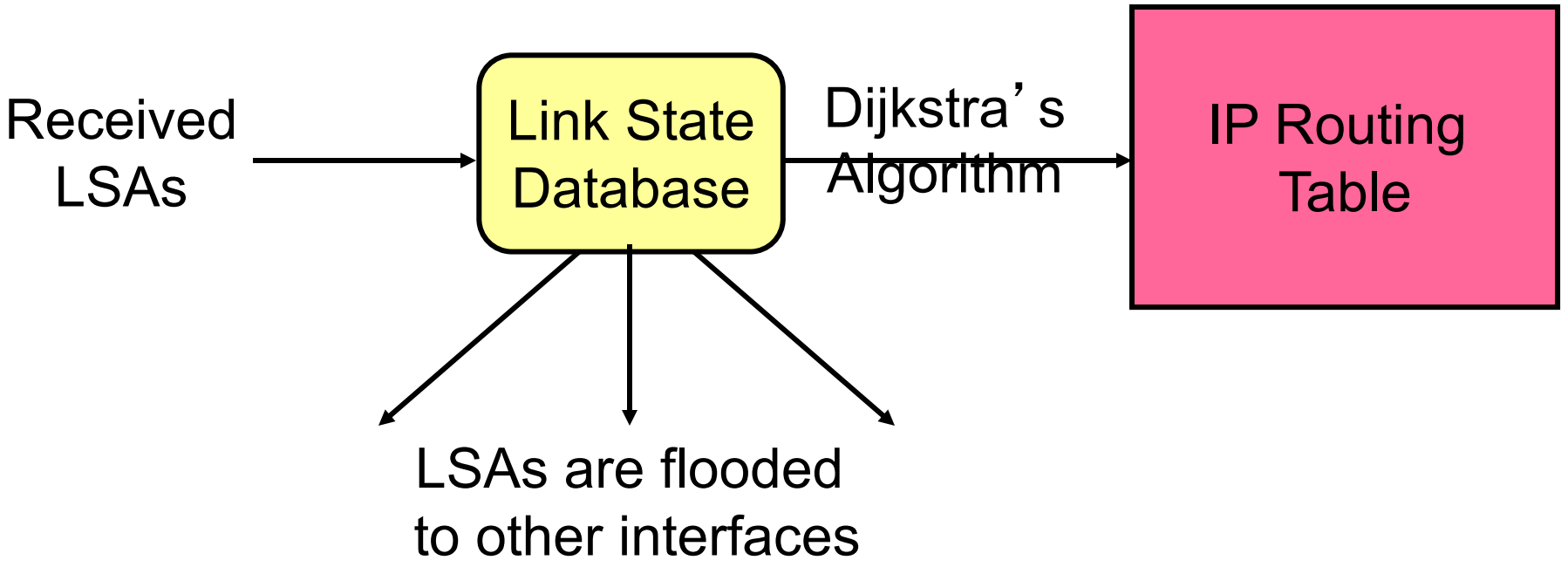
Link State Routing: Basic princples

1. Each router establishes a relationship *("adjacency")* with its neighbors
2. Each router generates *link state advertisements (LSAs)* which are distributed to all routers

   LSA = (link id, state of the link, cost, neighbors of the link)

3. Each router maintains a database of all received LSAs (*topological database* or *link state database*), which describes the network has a graph with weighted edges
4. Each router uses its link state database to run a shortest path algorithm (Dijikstra's algorithm) to produce the shortest path to each network

Operation of a Link State Routing protocol



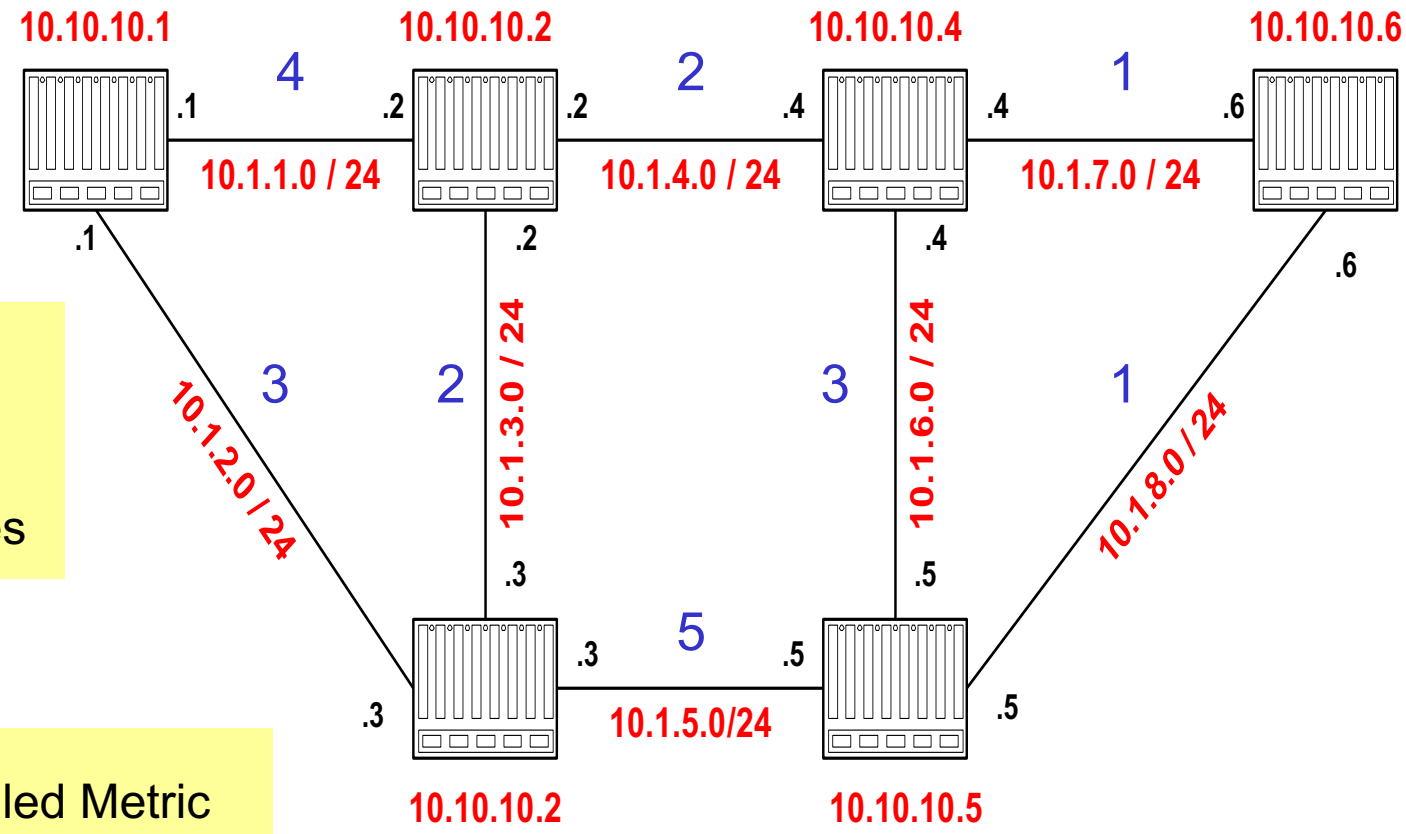
OSPF

- OSPF = Open Shortest Path First
- The OSPF routing protocol is the most important link state routing protocol on the Internet
- The complexity of OSPF is significant
- History:
  - 1989: RFC 1131 OSPF Version 1
  - 1991: RFC1247 OSPF Version 2
  - 1994: RFC 1583 OSPF Version 2 (revised)
  - 1997: RFC 2178 OSPF Version 2 (revised)
  - 1998: RFC 2328 OSPF Version 2 (current IPv4 version)
  - 2008: RFC5340 OSPF Version 3 (IPv6 version)

Features of OSPF

- Provides authentication of routing messages
- Enables load balancing by allowing traffic to be split evenly across routes with equal cost
- Type-of-Service routing allows to setup different routes dependent on the TOS field
- Supports subnetting
- Supports multicasting
- Allows hierarchical routing

Example Network



Router IDs are selected independent of interface addresses

Link costs are called Metric
Metric is in the range [0 , 2^{16}]
Metric can be asymmetric

Link State Advertisement (LSA)

Each router sends its LSA to all routers in the network (using a method called reliable flooding)

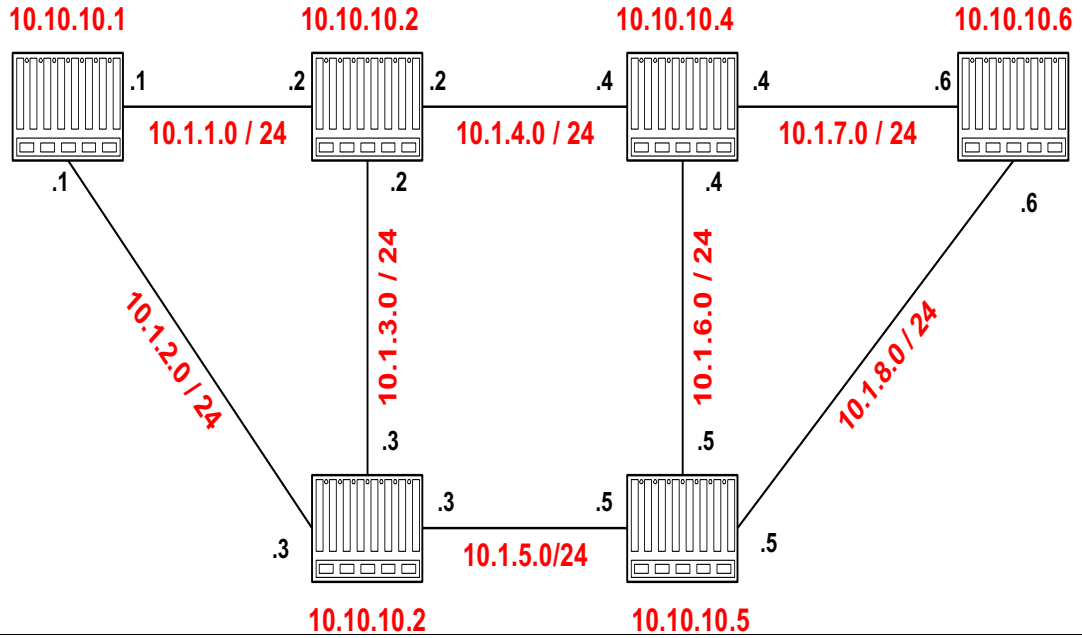
- **The LSA of router 10.10.10.1 in the previous diagram is as follows:**

- **Link State ID**: 10.10.10.1 = *Router ID*
- **Advertising Router:** 10.10.10.1 = *Router ID*
- **Number of links:** 3 = *2 links plus router itself*

- **Description of Link 1:** Link ID = 10.1.1.1, Metric = 4
- **Description of Link 2:** Link ID = 10.1.2.1, Metric = 3
- **Description of Link 3:** Link ID = 10.10.10.1, Metric = 0

Network and Link State Database

Each router has a database which contains the LSAs from all other routers

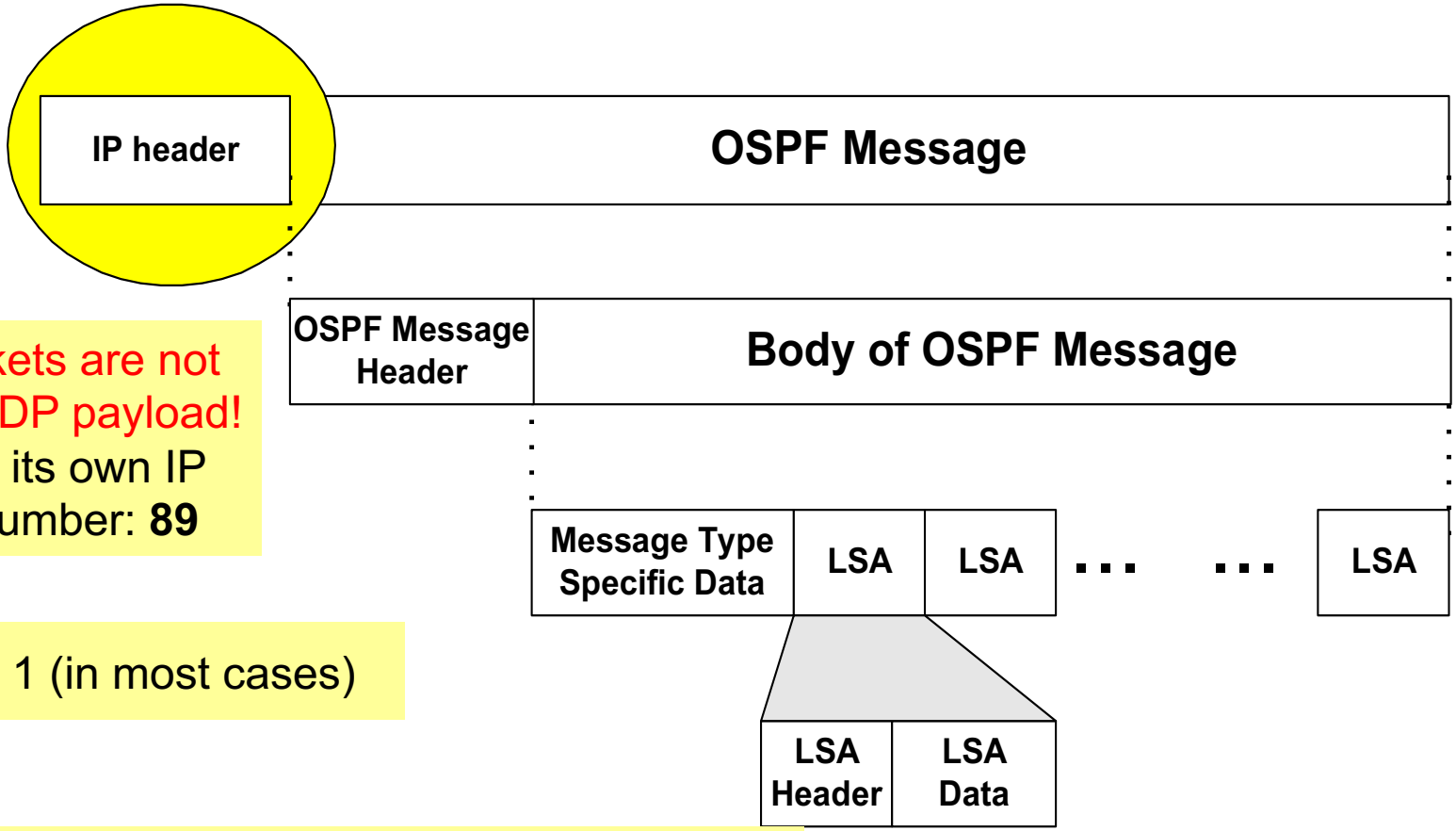


LS Type	Link StateID	Adv. Router	Checksum	LS SeqNo	LS Age
Router-LSA	10.1.10.1	10.1.10.1	0x9b47	0x80000006	0
Router-LSA	10.1.10.2	10.1.10.2	0x219e	0x80000007	1618
Router-LSA	10.1.10.3	10.1.10.3	0x6b53	0x80000003	1712
Router-LSA	10.1.10.4	10.1.10.4	0xe39a	0x8000003a	20
Router-LSA	10.1.10.5	10.1.10.5	0xd2a6	0x80000038	18
Router-LSA	10.1.10.6	10.1.10.6	0x05c3	0x80000005	1680

Link State Database

- The collection of all LSAs is called the link-state database
- Each router has an identical link-state database
  - Useful for debugging: Each router has a complete description of the network
- If neighboring routers discover each other for the first time, they will exchange their link-state databases
- The link-state databases are synchronized using reliable flooding

OSPF Packet Format

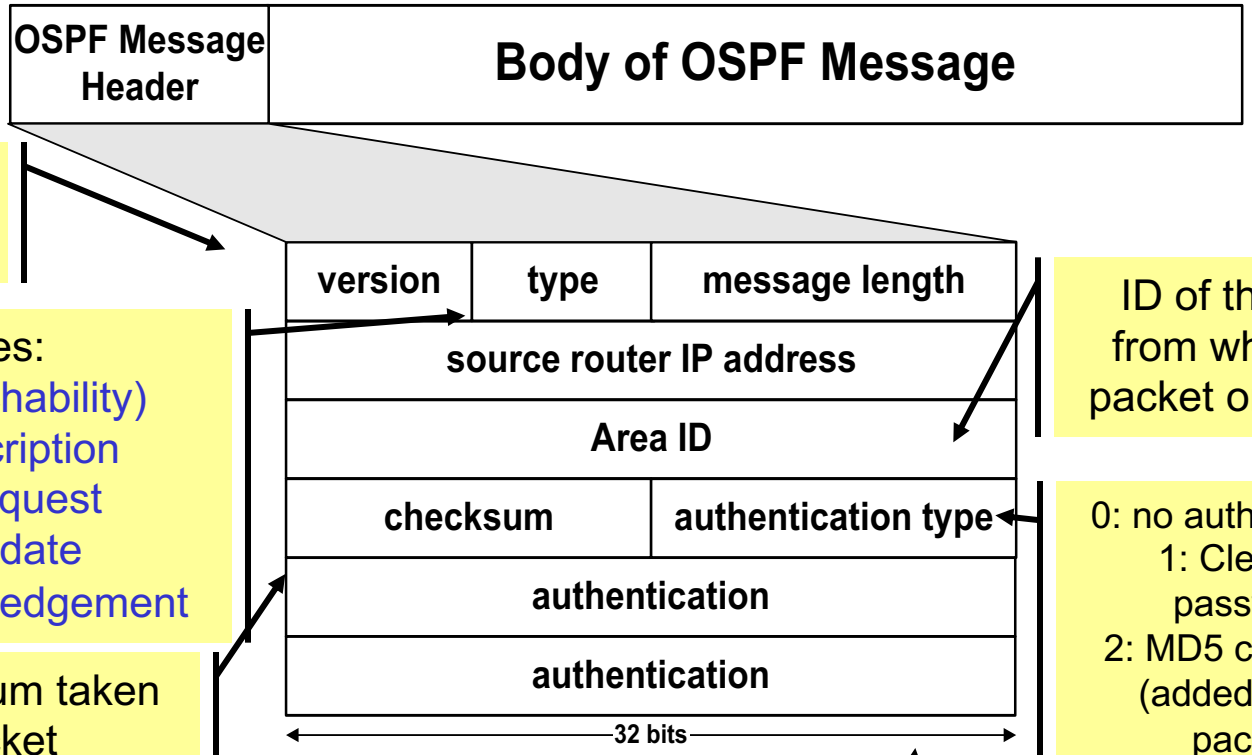


OSPF packets are not carried as UDP payload! OSPF has its own IP protocol number: **89**

TTL: set to 1 (in most cases)

Destination IP: neighbor's IP address or 224.0.0.5 (ALLSPFRouters) or 224.0.0.6 (AllDRouters)

OSPF Packet Format



2: current version is OSPF V2

Message types:
1: Hello (tests reachability)
2: Database description
3: Link Status request
4: Link state update
5: Link state acknowledgement

Standard IP checksum taken over entire packet

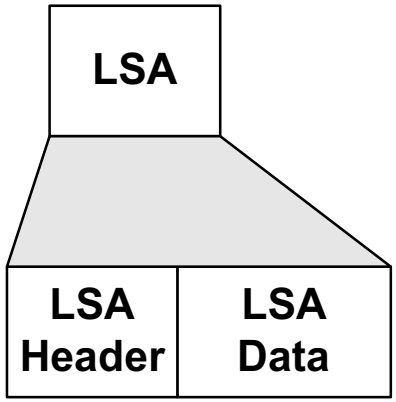
ID of the Area from which the packet originated

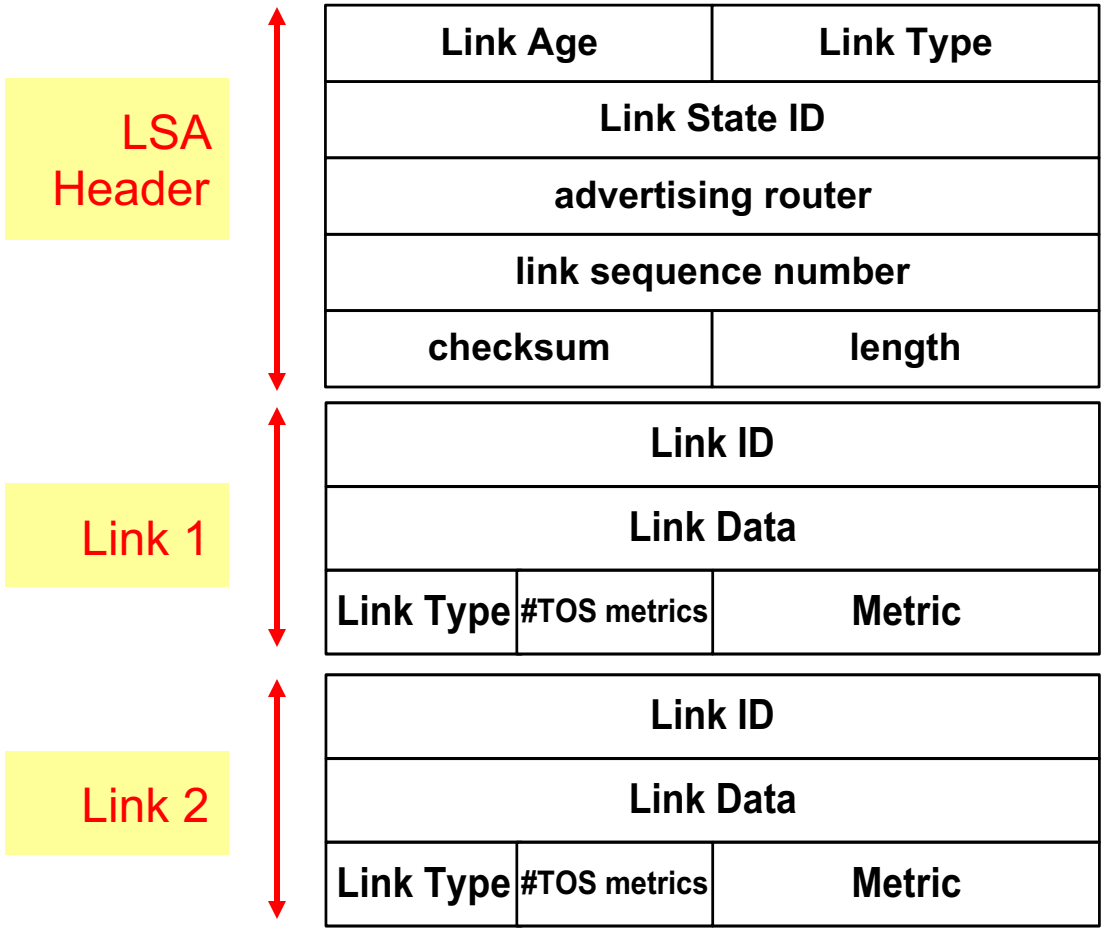
0: no authentication
1: Cleartext password
2: MD5 checksum (added to end packet)

Authentication passwd = 1: 64 cleartext password
Authentication passwd = 2: 0x0000 (16 bits)
KeyID (8 bits)
Length of MD5 checksum (8 bits)
Nondecreasing sequence number (32 bits)

Prevents replay attacks

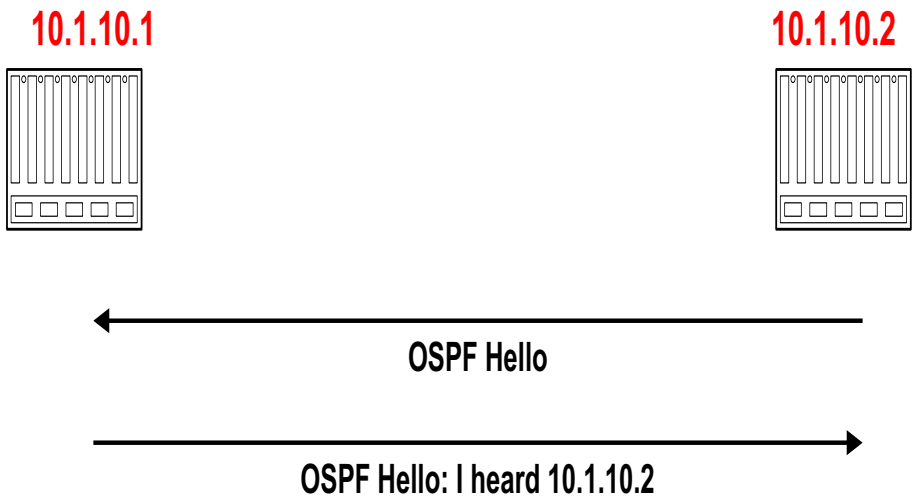
OSPF LSA Format





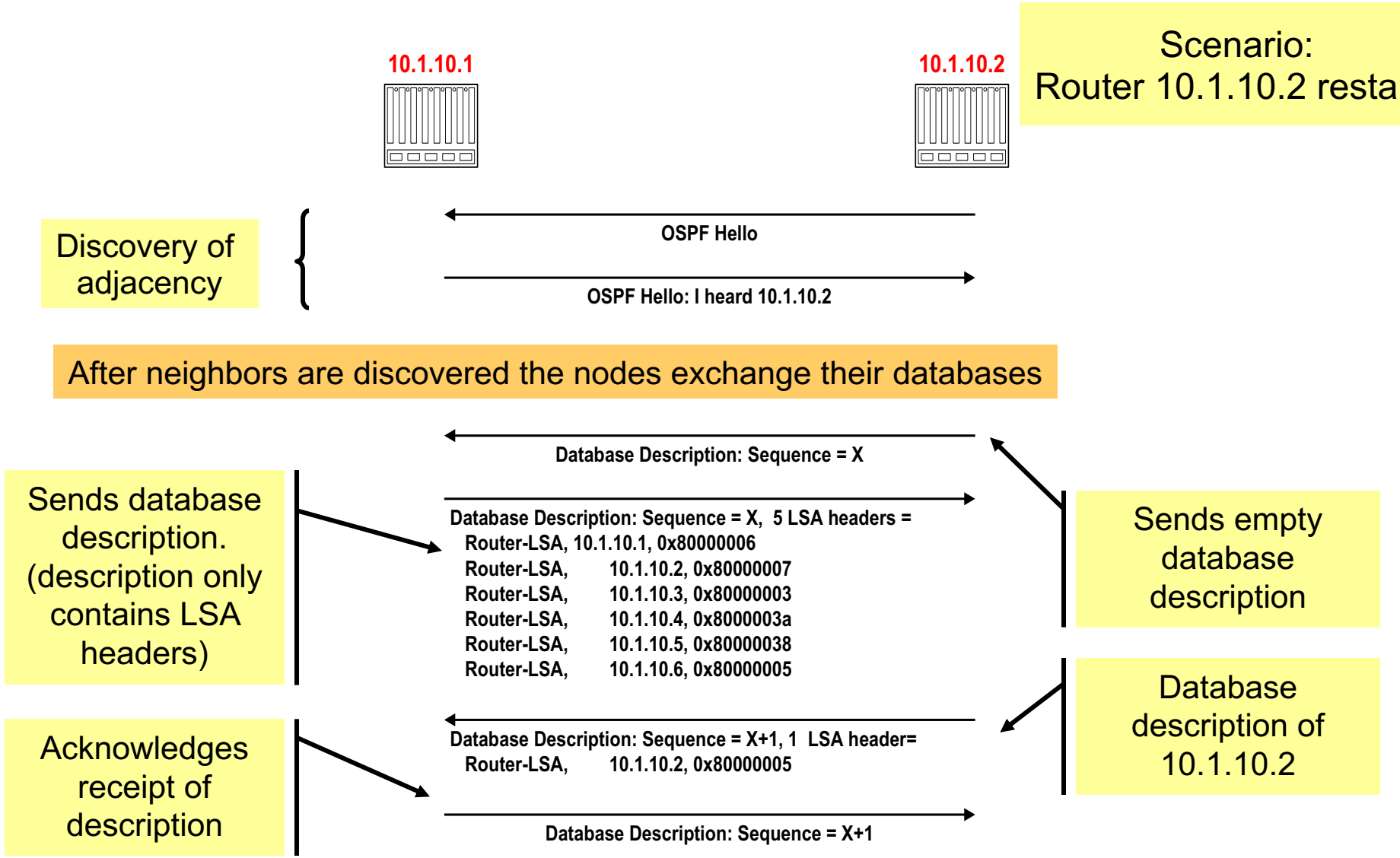
Discovery of Neighbors

- Routers multicasts OSPF Hello packets on all OSPF-enabled interfaces.
- If two routers share a link, they can become neighbors, and establish an adjacency

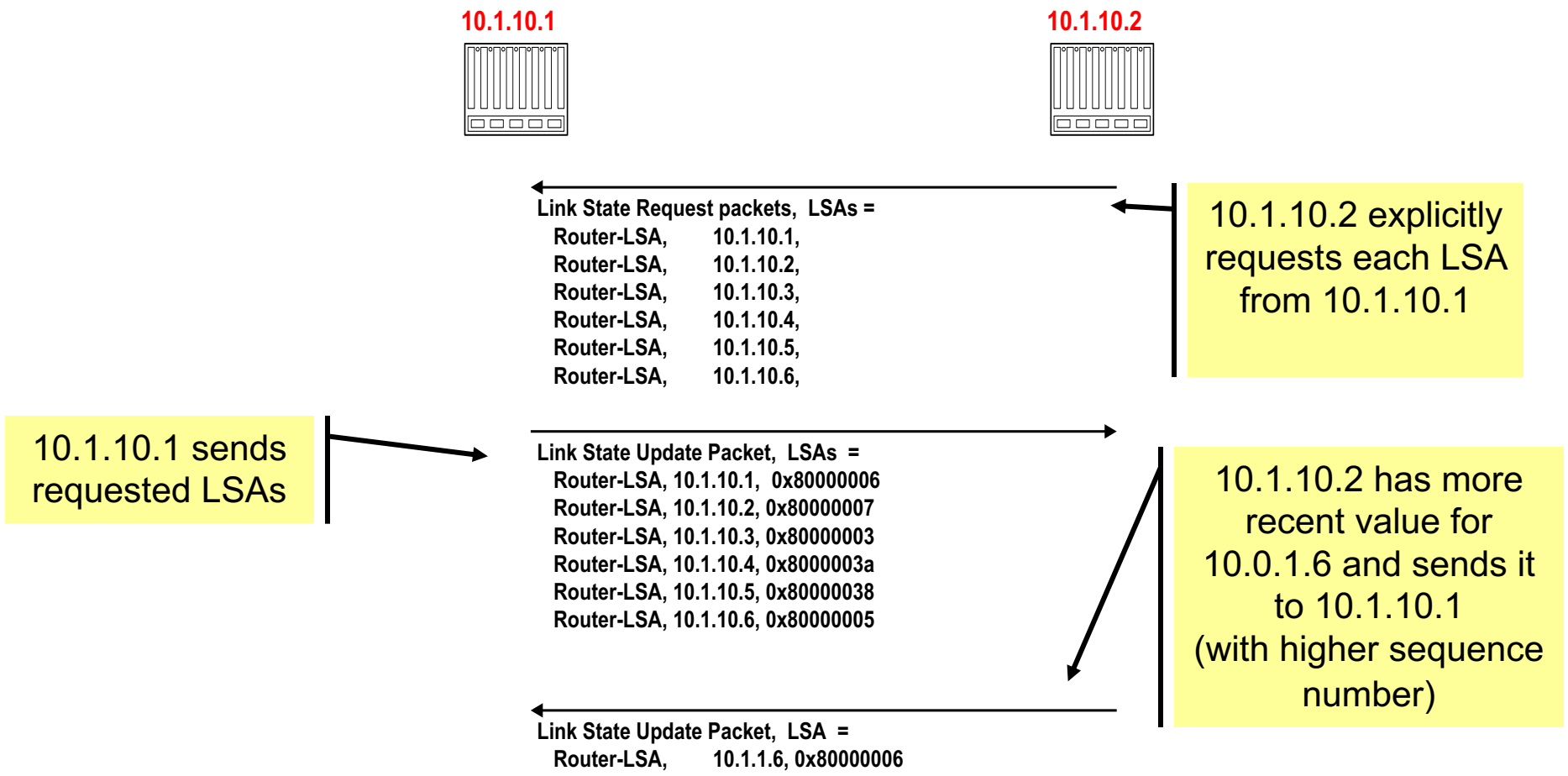


Scenario:
Router 10.1.10.2 restarts

Neighbor discovery and database synchronization

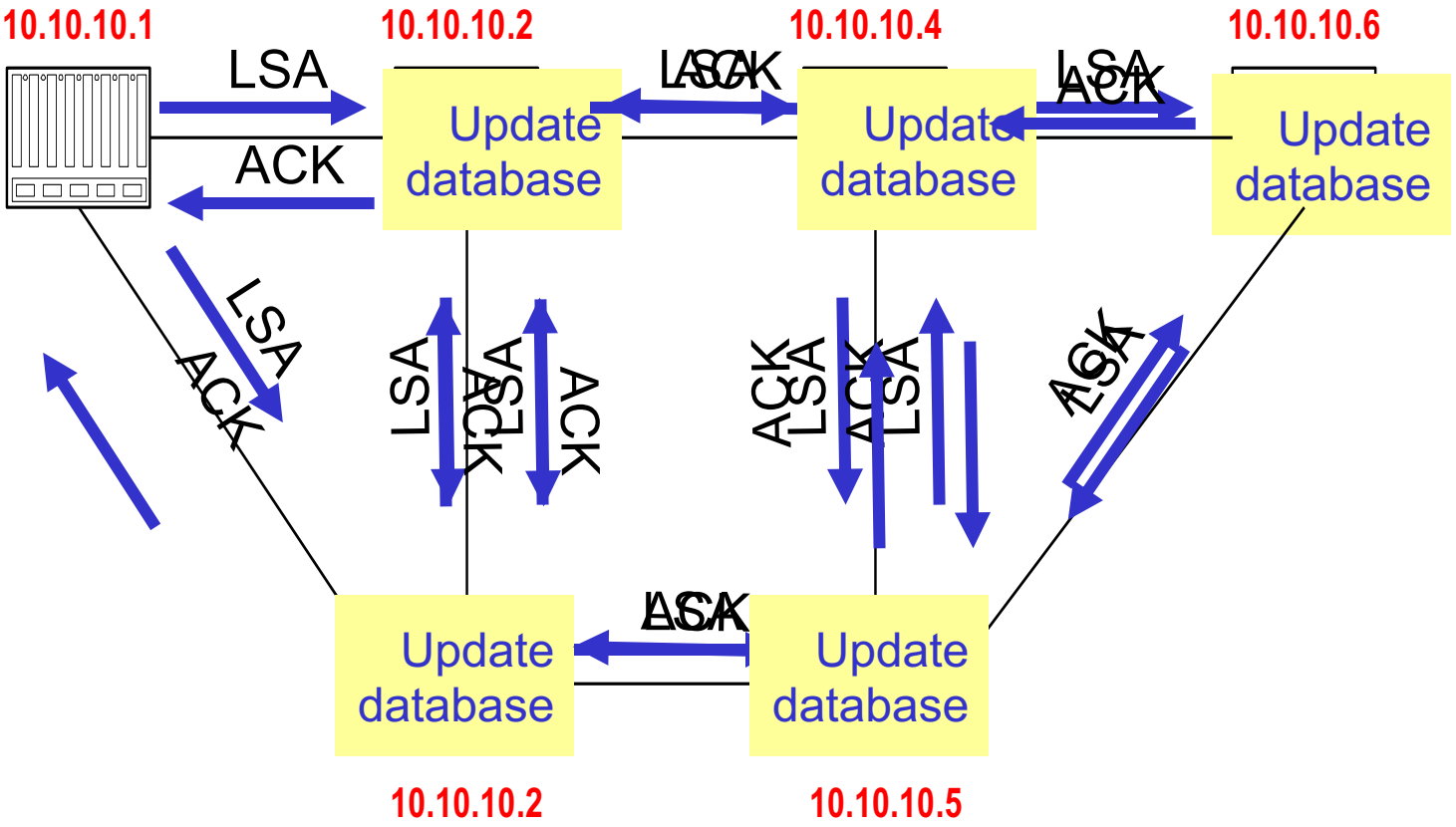


Regular LSA exchanges



Routing Data Distribution

- LSA-Updates are distributed to all other routers via **Reliable Flooding**
- **Example:** Flooding of LSA from 10.10.10.1



Dissemination of LSA-Update

- A router sends and refloods LSA-Updates, whenever the topology or link cost changes. (If a received LSA does not contain new information, the router will not flood the packet)
- Exception: Infrequently (every 30 minutes), a router will flood LSAs even if there are not new changes.
- Acknowledgements of LSA-updates:
  - explicit ACK, or
  - implicit via reception of an LSA-Update

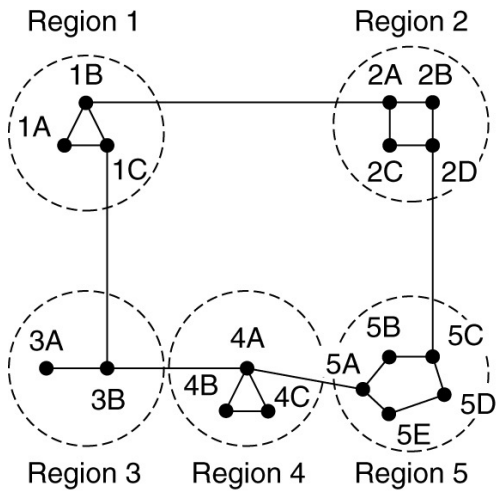
Computing the new routes

- Once a router has accumulated a full set of link state packets it can construct the entire subnet graph because every link is represented
- Dijkstra's algorithm can be run locally to construct the shortest path to all possible destinations
- The results of this algorithm will be installed in the routing tables and the normal operation resumed
- Practical consideration:
  - For a subnet having n routers, with k neighbors, the memory required to store the input data is proportional with kn. For large subnets, this may be a problem
  - Computation time can be also a problem
  - In many practical situations, the link state algorithm works well.

Hierarchical routing

- As networks grow in size, the routers routing tables grow proportionally; not only that, but also more CPU time is required to scan them and more bandwidth is required to send new status reports
- At a point, the network may grow beyond a point where it is not longer feasible for every router to have entry for every other router, so the routing has to be done hierarchically
- The routers are divided in so called regions, with each router knowing details about routers in its own region, but not knowing details about internal structure of other regions
- For huge networks, it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on, until we run off of names for aggregations.

Hierarchical routing



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

- Routing in two level hierarchy with five regions
  - Full routing table for 1A has 17 entries
  - For hierarchically routing, the routing table has 7 entries
- Penalty to be paid in the form of increase path length:
  - Best route from 1A to 5C is through region 2
  - With hierarchically routing, the traffic for region 5 goes through region 3, because that is better for most destinations in region 5

Autonomous Systems

- An **autonomous system** is a region of the Internet that is administered by a single entity.
- Examples of autonomous regions are:
  - Heanet's national network
  - Eircom's backbone network
  - Regional Internet Service Provider
- Routing is done differently within an autonomous system (**intradomain routing**) and between autonomous system **(interdomain routing**).

BGP

- BGP = Border Gateway Protocol
- Currently in version 4
- Note: In the context of BGP, a gateway is nothing else but an IP router that connects autonomous systems.
- Interdomain routing protocol for routing between autonomous systems
- Uses TCP to send routing messages
- BGP is neither a link state, nor a distance vector protocol. Routing messages in BGP contain complete routes.
- Network administrators can specify routing policies

BGP

- BGP's goal is to find any path (not an optimal one). Since the internals of the AS are never revealed, finding an optimal path is not feasible.
- For each autonomous system (AS), BGP distinguishes:
  - **local traffic** = traffic with source or destination in AS
  - **transit traffic** = traffic that passes through the AS
  - **Stub AS** = has connection to only one AS - carry local traffic
  - **Multihomed AS** = has connection to >1 AS, but does not transit traffic
  - **Transit AS** = has connection to >1 AS and carries transit traffic